2017 International Conference on Data Science and Advanced Analytics

Ensemble-based Location Tracking Using Passive RFID

Hao-Ying Liang*, Yun-Tung Shieh*, Addicam Sanjay†, Shao-Wen Yang†, Shou-De Lin*
*National Taiwan University
{b02505012, b02902024}@ntu.edu.tw, sdlin@csie.ntu.edu.tw
†Intel Corporation
{addicam.v.sanjay, shao-wen.yang}@intel.com

***Abstract*—Location tracking of passive RFID tags is useful for its ultra-low cost, but is very challenging for its passive nature. A passive RFID tag relies on no internal power source and draws power from the field created by the reader to power the microchip's circuits. This has made passive RFID tags highly sensitive to surrounding materials, as well as any disturbance. Therefore, conventional machine learning models may not perform well. In this paper, we propose an ensemble-based machine learning model, together with novel feature engineering techniques, for location tracking of passive RFID, which can work seamlessly with any supervised learning methods. The reader-based sub-models training method used in our model significantly reduces the training time by splitting our model into smaller sub-models and training them in parallel, which is desirable in real-world application.**

***Keywords*—RFID, RSSI, Indoor Location, Positioning Algorithm, Ensemble.**

I. Introduction

Radio frequency identification (RFID) is a ubiquitous technique applied in various areas for over a decade [1]. While we are in the midst of a major shift from Internet to the Internet of Things (IoT) – the heart of this disruption is making sense of the significant amount of data being generated – location and time are the key to understanding many things. This paper addresses the problem of location tracking using RFID technology.

RFID, by a broad definition, is a blanket term that covers anything that can be identified using radio frequencies. RFID is available at multiple frequencies with diverse sets of technologies, and, therefore, comes with performance and cost characteristics. RFID technologies can be categorized by usage, frequency, physical property, and data [1]. A popular taxonomy is defined by the physical property of internal power source. Specifically, active RFID and passive RFID are fundamentally different technologies; however, they are often compared and evaluated together because internal power source is usually a deciding factor for designing low-power small form factor devices, especially in the context of the Internet of Things (IoT).

There are three types of RFIDs, differentiated by whether or not an internal power source is used and the way the internal power source is used.

- Active RFID has internal power source for powering the microcontroller and transmitting signals to a reader.
- Passive RFID does not require any internal power source and relies solely on power drawn from the signal from a reader.
- Semi-passive RFID uses an internal power source to run the microcontroller but draws the power from a reader just like passive RFID.

Passive RFID is particularly useful for deploying onto high-volume, low-cost assets, e.g., clothes, because no internal power source implies smaller form factor and lower costs. For example, Walmart has been trying to adopt RFID technology for inventory management so that they can cut the volume of excess inventory in its massive supply chain.

In this paper, we propose feature generation techniques for location tracking of passive RFID that can work seamlessly with any supervised learning methods. We also propose a reader-based sub-models training method, together with an ensemble solution based on multi-label classification, that makes our approach very efficient in terms of training time. In our case, we use a retail dataset as an example, but our method shall be applicable to the tracking of any high-volume, low-cost assets, e.g., warehouse logistics.

We collected around 60M real-world indoor RFID tracking data instances to conduct experiments. To our knowledge, this is the first-ever empirical study on RFID location tracking that involves this huge amount of ground truth labeled data instances.

II. Background

Location tracking has been well studied in the literature and the industry in the past decade, which has enabled a variety of location based services. Parametric (or model-based) methods include received signal strength (RSS) lateration [2], time of arrival (TOA) lateration [3], time difference of arrival (TDOA) weighted mean squares [4], received signal phase (RSP) [5], angle of arrival (AOA) [6], etc. These methods are based on certain assumptions of predetermined models and parameters, e.g., path loss propagation modeling, and can be vulnerable to noise. Fingerprinting, on the other hand, is a data driven model[2], and has been the most promising in providing accurate indoor location estimates. It is based on k-nearest neighbor search on a pre-recorded RSS-to-location database.

[1] http://www.rfidjournal.com/

978-1-5090-5004-8/17 $31.00 © 2017 IEEE
DOI 10.1109/DSAA.2017.36

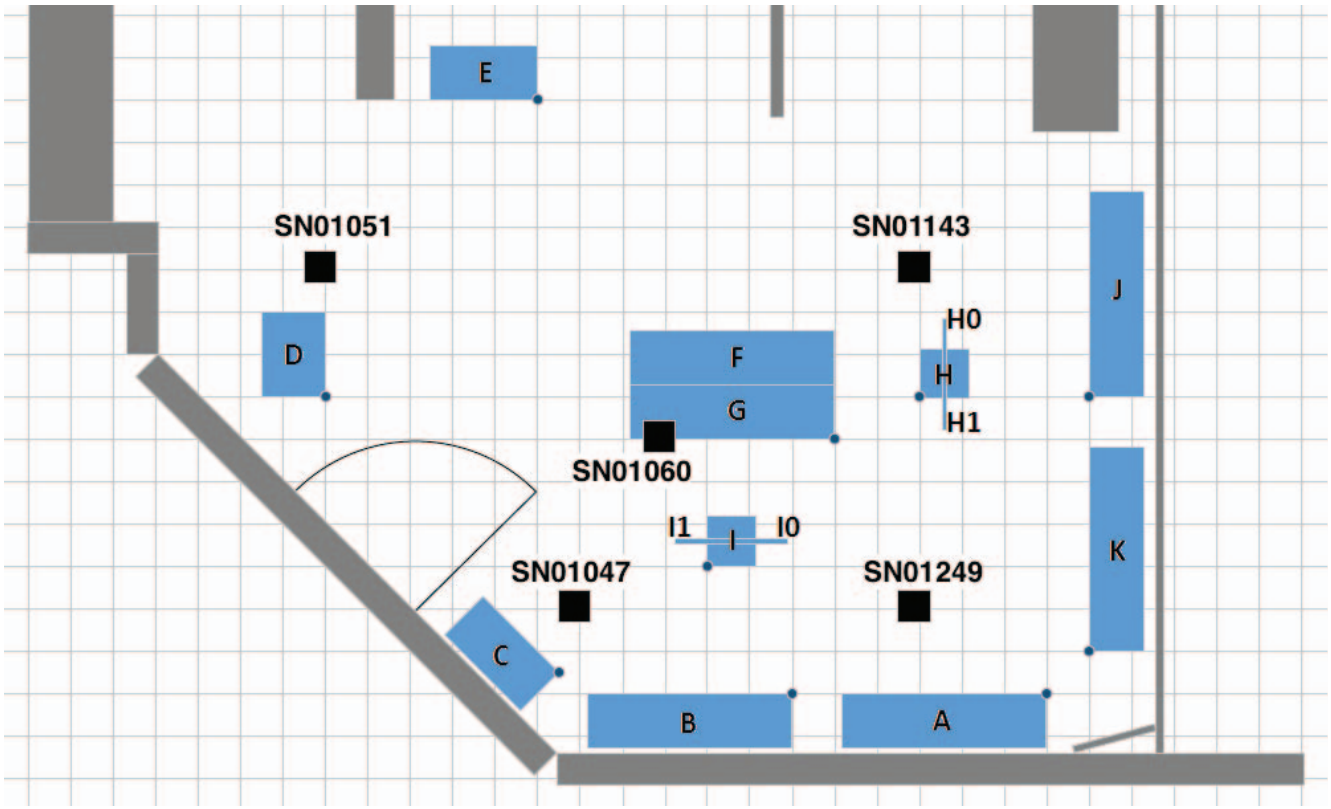

Fig. 1: The interior layout of the experimental clothes shop. The black squares are readers, and the blue rectangles are shelves, where clothes with tags are placed. Each grid has a dimension of one foot by one foot.

Data driven methods can be much more robust against noises, in particular when the noise models are unknown. This work falls into the category of data driven methods powered by state of the art machine learning techniques.

The major concerns of deploying any existing location tracking solutions are either in their performance inadequacy or their requirements for substantial infrastructure deployment efforts. The use of passive RFID addresses the latter concern to some extent because the battery-free nature makes a location tracking solution easy to deploy; in combination with its low cost, the location tracking solution can be almost maintenance-free. However, the battery-free nature also makes location tracking tricky because the measurements, i.e., received signal strength indicator (RSSI), can be particularly unreliable. This paper uses an ensemble-based machine learning model, together with novel feature engineering techniques to address the former concern of accuracy inadequacy.

III. Problem Definition

This paper addresses the location tracking problem in the context of asset tracking. Just as any data driven methods like fingerprinting, the location is discrete by discretizing the space. In the context of indoor localization where people can maneuver continuously across the space, location is defined by densely-sampled grids over the space; in the context of asset tracking, location is defined by certain predetermined landmarks. For example, in a retail store, clothes are either on top of a table, in a rack or on a shelf.

In our scenario, we use RFID in a retail clothes store, where there are hundreds of clothes with tags, ten shelves, and five readers. Our goal is to tell which shelf a cloth is located in, given the received data instance of a tag, which consist of its RSSI, phase, frequency, scanned multiple times in a period of time. This can be classified into the category of multi-class classification problems.

IV. Dataset

The dataset was collected in a laboratory in Intel facility in Arizona replicating a real brick-and-mortar apparel shop setting. In the following, we describe in detail the infrastructure used for data collection, the way the ground truth labeled data were collected, and how the ground truth labels are defined. Finally, we conduct several statistical analyses on the dataset.

A. Hardware and Software Settings

The data were collected using Intel®Responsive Retail Platform (RRP) [2]. It is an end-to-end infrastructure including Responsive Retail Sensor (or sensor), which features an integrated RFID reader and antenna. In our setting, there are

[2]https://newsroom.intel.com/newsroom/wp-content/uploads/sites/11/2017/01/IntelResponsiveRetailPlatformFactSheet.pdf

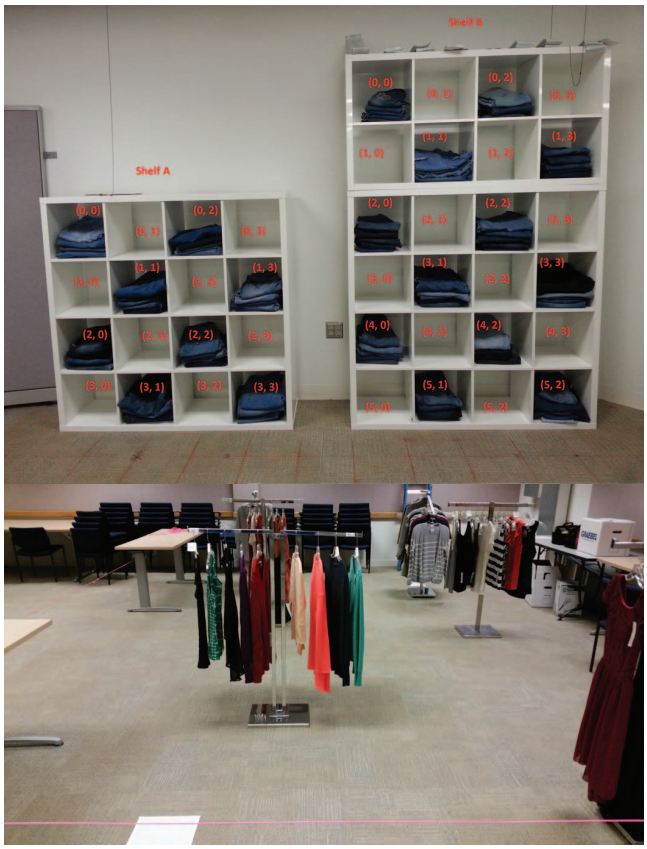

Fig. 2: Shelf setting

five sensors, as depicted in Figure 1 in 1-by-1 foot grids, where dark squares are sensors and light (blue) rectangles are shelves or tables.

Figure 2 shows the setting of the two shelves on the right of Figure 1. As can be seen, this is a fine replicate of an apparel shop where clothes are organized just like the way it is in a real store.

In order to elaborate on the software settings, we first start with a brief description about the way RFID tags are designed. RFID tag has two states – A and B, and alternates between the two states. Tags in state A are considered not-yet-inventoried and hence readable by a sensor; on the other hand, tags in state B are considered already-inventoried or of-no-interests and hence invisible to any sensor. Based on the definition of states, RFID sensor has four exclusive sessions – 0, 1, 2 and 3. In a nutshell, when an RFID tag is read by a reader, its state is switched by the reader from state A to state B; a session defines how long (in seconds) the tag persists in state B. For example, in session 0, a tag switches back to state A immediately after being read, whereas in session 2, a tag persists in state B for at least two seconds before switching back to state A.

In addition to states and sessions, RRP also provides customization of reading behaviors – deepscan and mobility. Deepscan behavior is intended to read tags (inventory) deeply; mobility behavior is intended to allow fast scanning of easily read tags.

Fig. 3: RFID tag

The data were collected continuously, where sensors are in session 2 with mobility mode in the daytime, and in session 0 with deepscan mode in the nighttime. This is a common practice so that it can read a large tag population quickly in the daytime by minimizing duplicate reads, and read tags deeply in the nighttime when tags are almost surely stationary.

B. Data Collection

While data were collected continuously by sensors mounted in the ceiling, the ground truth labels were collected separately with a TSL 1128 Bluetooth UHF RFID Reader. Basically, we manually scanned every tag on the shelves and racks and labeled their true locations. There are in total five sensors, each of which comes with one antenna and one reader, and approximately 300 RFID tags. The data were collected continuously day and night for five consecutive days, with a total number of entries of around 60M. Tab. I illustrates a snippet of the data, where EPC and antenna identity (ID) are unique strings identifying tags and antennas, respectively, and labels are the enumeration of discrete locations, which in our case are shelves, along with numeric fields including RSSI, phase, frequency and timestamp.

C. Ground Truth Labels

Given Figure 1, we define the ground truth labels as shelves A, BC, E, FG, H, I, JK. Notice that we have combined some nearby shelves into a single label for simplicity, as in BC, FG, and JK.

D. Statistical Analyses

Analyses of different aspects are conducted here to help understand the data in a more detailed way, and the results are presented as charts alongside.

In particular, we calculate the raw data distribution from the perspectives of each shelf. That is, we see how many raw data instances were collected from each shelf. The result is shown in Figure 4.

Further analyses are done on distinct clothes, rather than raw data instances. Figure 5 shows how many clothes are located in each shelf, while Figures 6 to 10 present the distributions of clothes read by each reader

V. Methodology

Figure 11 shows the overall process of our solution. We first perform data preprocessing and feature engineering, which include aggregation of the data over time, and extracting

TABLE I: A snippet of the data. Note that all EPCs have a prefix of 3014032F440F0E0540.

Raw Data						
EPC	RSSI	AntennaID	Phase	Frequency	Timestamp	Label
7CD02C	-50.0	SN01051	0.8344855486097889	906.25	1449730800021	E
7AE21B	-71.0	SN01047	2.6016314162540475	909.75	1449730800039	B
...	...	...	...	...	...	...

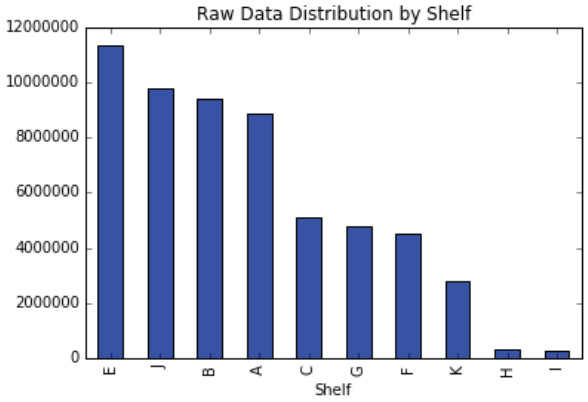

Fig. 4: Raw data distribution from the perspective of each shelf.

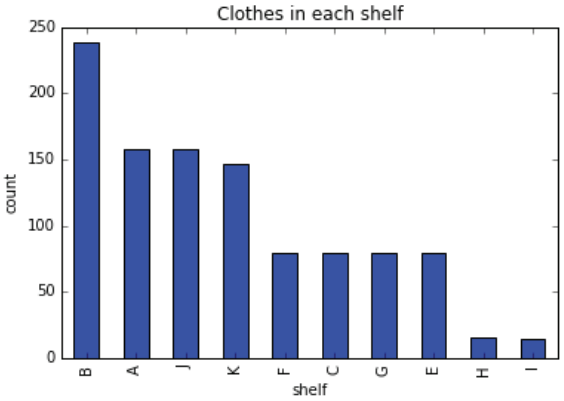

Fig. 5: Number of distinct clothes in each shelf.

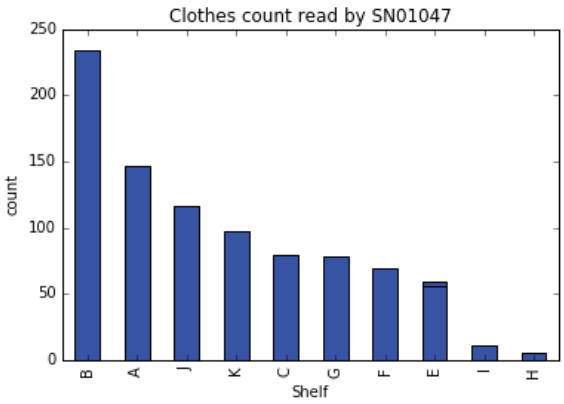

Fig. 6: Distinct counts of clothes in each shelf scanned by SN01047.

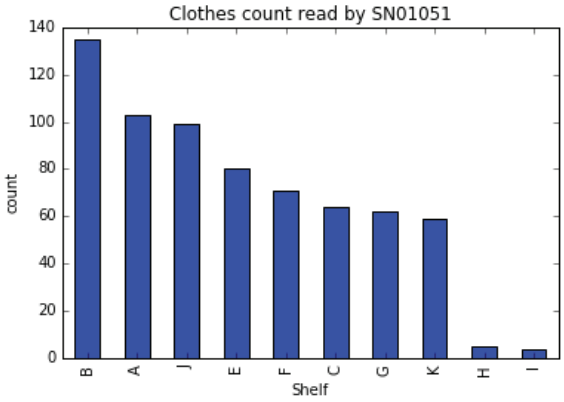

Fig. 7: Distinct counts of clothes in each shelf scanned by SN01051.

additional aggregate features. For each reader, we then construct an individual predictor for location tracking. Finally, the probabilistic outputs of each predictor are integrated using multi-label classification techniques to produce the final result. Below, we will elaborate on each of these components.

A. Aggregation

RFID signals can be scanned in a relatively frequent manner (e.g. once per second). However, typically we do not need to make a decision about a tag's location that frequently. Thus, whenever a decision of location has to be made, normally there have already been multiple instances received. We consider it an advantage from the prediction point of view, since the model can gather more information before making a decision. Thus, in this stage we propose an aggregation method that aggregates all the data of a tag scanned in a short period of time to jointly make a decision about its location.

The way we carry out this aggregation is intuitive. Since each record has a timestamp associated with it, we first partition the time into periods of equal lengths, say five minutes, which we call time spans, and then group together the data that were scanned in the same time span. Then, for each tag, we combine all the data in the same group into one by averaging the feature values.

One thing to note here is that we have multiple readers at different locations, and each of them can be scanning the same tag. Because the distributions of data from different readers are likely to vary significantly, aggregating data that were not

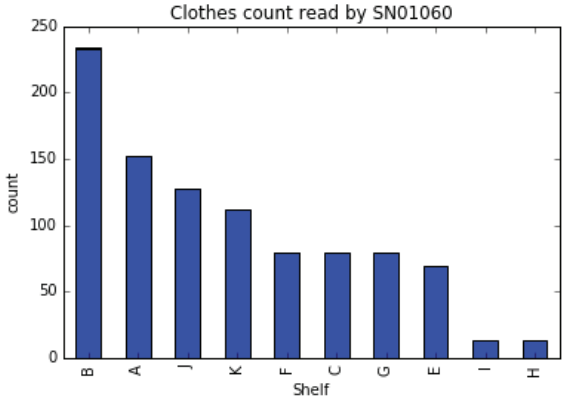

Fig. 8: Distinct counts of clothes in each shelf scanned by SN01060.

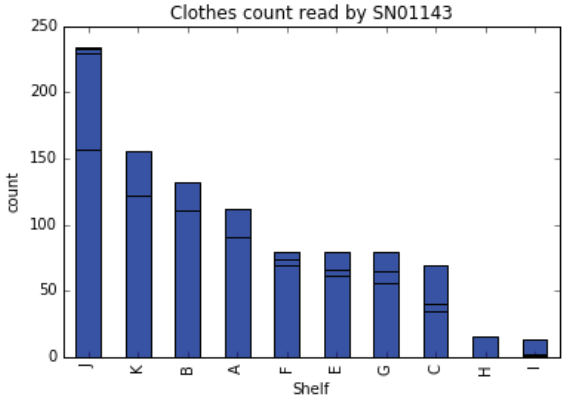

Fig. 9: Distinct counts of clothes in each shelf scanned by SN01143.

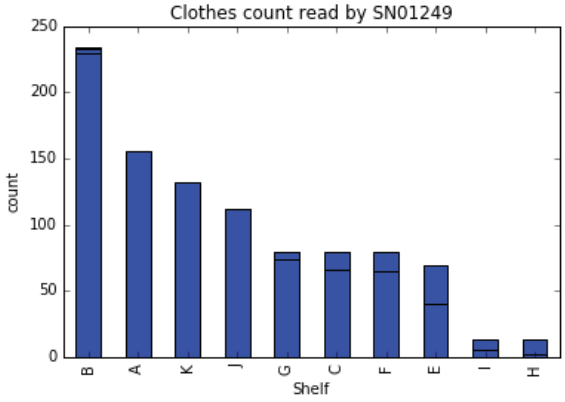

Fig. 10: Distinct counts of clothes in each shelf scanned by SN01249.

scanned by the same reader is not really meaningful. Thus, we aggregate only the data that came from the same reader. For example, in a certain time span, if a tag was scanned by, say one reader for 20 times, another for 15 times, but none of the others, then there will be only two resulting entries in that time span, which are the average of the 20 entries scanned by the first reader, and that of the 15 entries scanned by the second reader, respectively.

In addition to calculating the average of the data, we also incorporate two sets of aggregate features at this stage. The first is the numbers of times that a tag was scanned by each reader in a time span. For example, if a tag was scanned by the first reader 90 times, by the second reader 110 times, but not by any other readers, in a given time span, then the first set of aggregate features for this tag would be 90, 110, 0, 0, 0. The second set, derived from the first, is the proportions of numbers of times that a tag was scanned by each reader, respectively. In the case of the previous example, the second set of aggregate features for this tag would be 0.45, 0.55, 0, 0, 0.

We call these two sets of aggregate features together the count features. We expect these count features to be useful, as we consider that tags located in different shelves must have quite different distributions of numbers of times being scanned by each reader, which means they must differentiate themselves in terms of these count features.

Table I shows the raw data we have obtained and Table II shows the count features.

B. Reader-based Sub-models Partitioning

Given that the data were collected by multiple different readers, it is a natural choice to construct sub-models based on readers, because the raw data collected from a tag may differ significantly if scanned by different readers, due to the differences in the distance to the reader, the obstacles in between, and the magnitude of interference, just to name a few. However, most machine learning algorithms assume and depend on the fact that data are coming from the same distribution. By partitioning our model into reader-based sub-models, we can be certain that the data in each sub-model come from the same distribution.

Some may say this complicated partitioning scheme can be replaced by turning reader ID into a one-hot encoded categorical feature, which can also solve the problems we propose above. Yes, but in our case where there are millions of data, training the whole big model takes prohibitively long time. With reader-based sub-models partitioning, the training time of each sub-model is several times less than that of the whole model. Moreover, since the training of each sub-model is independent, they can be trained in parallel, which further reduces the training time.

In the experiment section, we will show that the reader-based sub-models training method we propose yields comparable accuracy rate to that of the whole model with reader ID encoded as categorical feature, while remaining time-efficient, making our model much preferable in real-world applications.

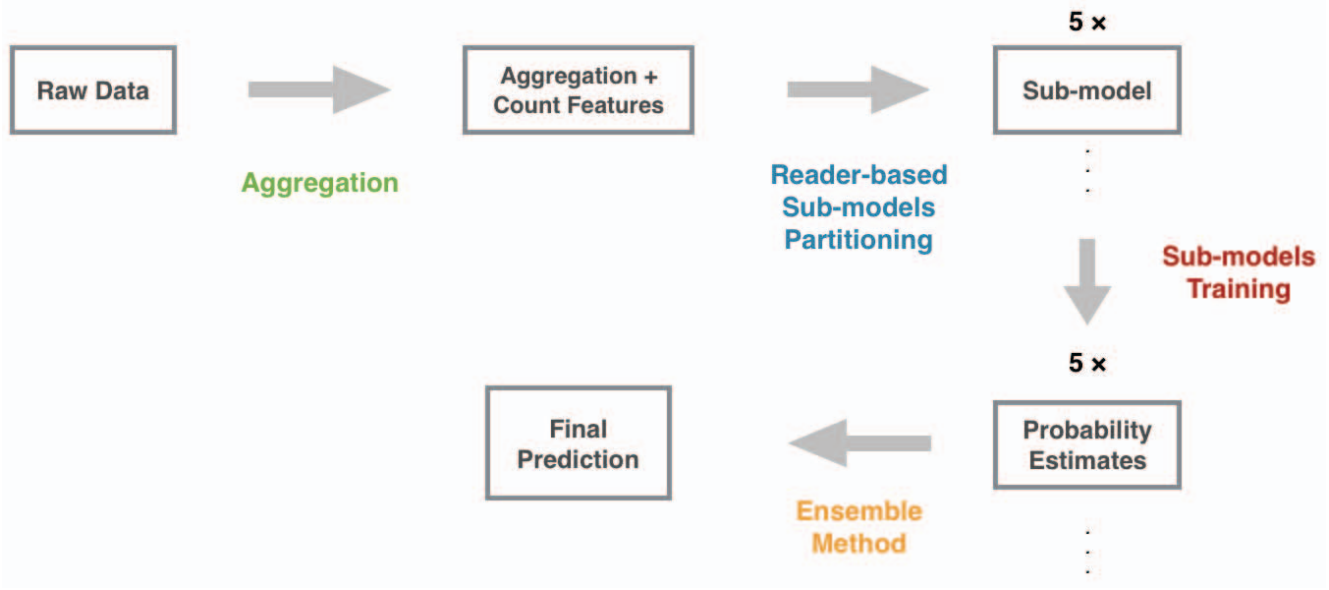

Fig. 11: The big picture of our methodology.

TABLE II: The count features incorporated during aggregation. Here, the time span is set to 600 seconds.

Count Features										
EPC	SN01051	SN01047	SN01060	SN01249	SN01143	n_SN01051	n_SN01047	n_SN01060	n_SN01249	n_SN01143
7AC0BA	0	63	1	8	0	0.000	0.875	0.014	0.111	0.000
7AC0BB	0	32	8	416	0	0.000	0.070	0.018	0.912	0.000
...	...	...	...	...	...	...	...	...	...	...

C. Sub-models Training

Having gone through all the pre-processing of the data, we begin to apply various machine learning algorithms to solve these multi-class classification problems in each of the sub-models.

The models we have tried primarily include logistic regression (LR), deep neural network (DNN), and gradient boosting decision tree (GBDT). LR is a basic and efficient machine learning solution, while DNN and GBDT are two state-of-the-art solutions that have achieved decent performance in various competitions. Our experiments indicate that LR and DNN achieved comparable accuracy rates in our case, while GBDT significantly outperformed both of them.

Note that at this stage, a tag may be predicted multiple times, as there is more than one sub-model. Hence, there shall be a way to combine all the predictions of a tag in a time span into one, after taking each of them into consideration, and this is where our ensemble methods will come into play.

D. Ensemble Methods

As mentioned above, currently, a tag may have multiple predictions in a time span by different readers, but only one final prediction is expected. Thus, we employ an ensemble solution to somehow combine different sub-models' predictions.

As their name suggests, ensemble methods make use of multiple models to achieve a better performance than could be otherwise achieved by any of these models alone. Despite the fact that we are using the name ensemble methods here, the scenario we are facing is very different from a typical ensemble task, as normally an ensemble task assumes the models to be combined are trained on the same set of data. However, in our case, the sub-models are trained on different sets of data, each collected by a different readers. Thus, a typical ensemble method may not be applicable in our scenario.

To make use of the results from the sub-models, we should first grab the predictions from each of them. Previously, we had our sub-models directly predict which shelf each tag belongs to. In fact, these predictions were generated in two steps. First, the sub-models produce the probability estimates of each tag belonging to each shelf. Then, for each tag, the sub-models chose the shelf with the highest probability estimate to be the prediction. Here, instead of using the predicted labels from the second step, we use the probability estimates from the first step as input to our ensemble methods, as they contain much more information than a single label output. Table III gives a snippet of the resulting features.

First, we try a typical ensemble model. For each tag, given all of its probability estimates predicted by different sub-models, we simply average them, and then choose the shelf with the highest resulting probability estimate to be the final prediction. In simpler words, this can be thought of as we are trusting all sub-models equally, so we let them vote, and see which shelf is the winner.

Then, we propose a more sophisticated ensemble method powered by multi-label classification. Instead of trusting all

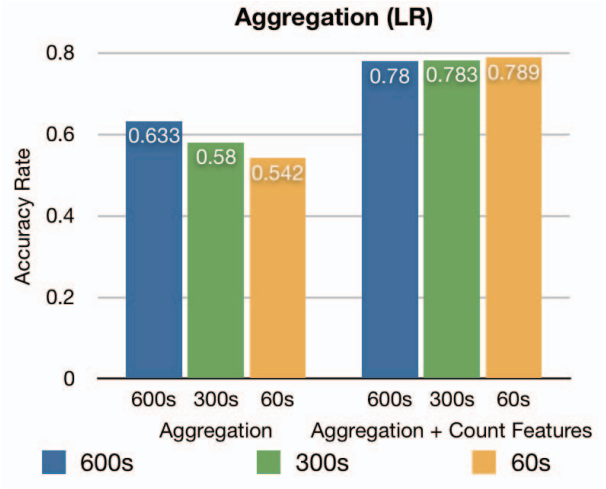

Fig. 12: Comparison of accuracy rates between aggregation with and without count features, and with different length of time span. Here, LR is used as the training algorithm.

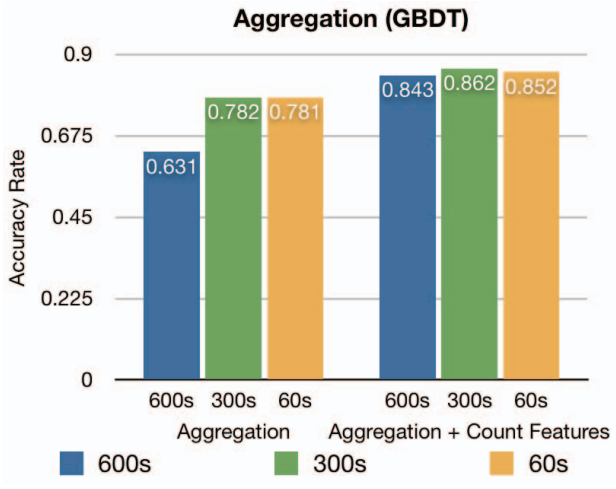

Fig. 13: Comparison of accuracy rates between aggregation with and without count features, and with different lengths of time span. Here, GBDT is used as the training algorithm.

sub-models equally, we learn from the data whether we should trust a sub-model. In details, for each tag, given all of its probability estimates predicted by different sub-models, along with some other features of it including its count features, we should tell which sub-models to trust. We pose this as a multi-label supervised learning problem, and the ground truths here are whether each sub-model has correctly predicted the belonging shelf of each tag. For example, if a tag has been correctly predicted by the first two sub-models, but either not predicted or wrongly predicted by the other three sub-models, then the ground truth for this tag would be $[1, 1, 0, 0, 0]$, where 1 means to trust that sub-model, and 0 means the opposite. After learning which sub-models we should trust for a tag, we average only the probability estimates that were predicted by the trusted sub-models, after which we choose the shelf with the highest resulting probability estimate to be the final prediction. In simpler words, this can be thought of as we first elect some of the sub-models as representatives, who will then vote to decide which shelf should be the final prediction. In practice, we have tried Binary Relevance (BR) and Random k-Labelsets (RAkEL), two remarkable multi-label classification algorithms, for our purpose.

VI. Experiments

During our work, we have made several hypotheses and observations to verify our methodology. Here, we describe them in details and show their results.

In the following experiments, we split our data into 80% of training data, and 20% of testing data based on timestamp (data that were collected later are taken as testing data). The following results are all based on the testing data.

A. Hypothesis: Count Features Must Carry Important Information

In this experiment we compare the accuracy rates of aggregation with and without count features, and with different lengths of time span given different algorithms.

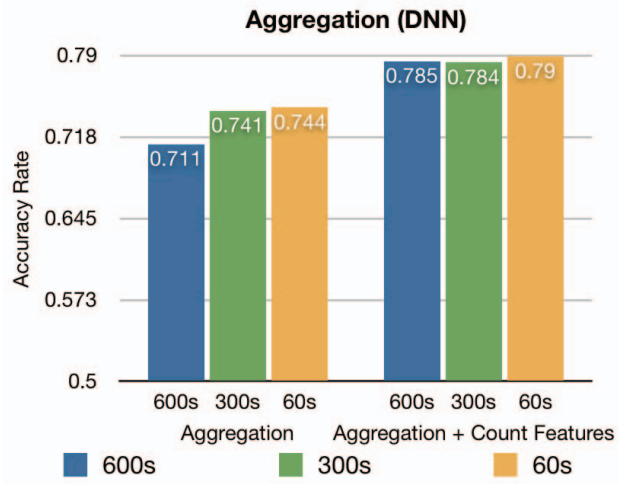

Fig. 14: Comparison of accuracy rates between aggregation with and without count features, and with different lengths of time span. Here, DNN is used as the training algorithm.

The results are shown in Figure 12, Figure 13, and Figure 14. As can be seen clearly, count features contribute significant improvement.

Also, we can notice that the length of time span does not seem to have any huge influence on the models with count features incorporated. Thus, in this case, the length of time span can be set to any reasonable value regarding the application needs. As for the models without count features, the longer the time span, the lower the accuracy rates. This is in our expectation because the major advantage of having a longer time span is that more count features can be collected within a time span, but since we exclude the count features here, we would only suffer more from data shrinkage due to aggregation if we use a longer time span.

TABLE III: Data that serve as input to the ensemble methods, where is_correct_(AntennaIDs) are the labels, p_(AntennaIDs)_(Shelves) are the probability estimates given by each reader on each shelf, and rssi_(AntennaIDs) are the RSSIs collected by each reader. Note that count features are also incorporated as input.

Data Prepared for Ensemble Methods						
EPC	is_correct_SN01051	...	p_SN01051_A	...	rssi_SN01051	...
7AC0CC	1	...	7.31e-06	...	-75.6	...
7AC0FC	1	...	1.88e-06	...	-74.0	...

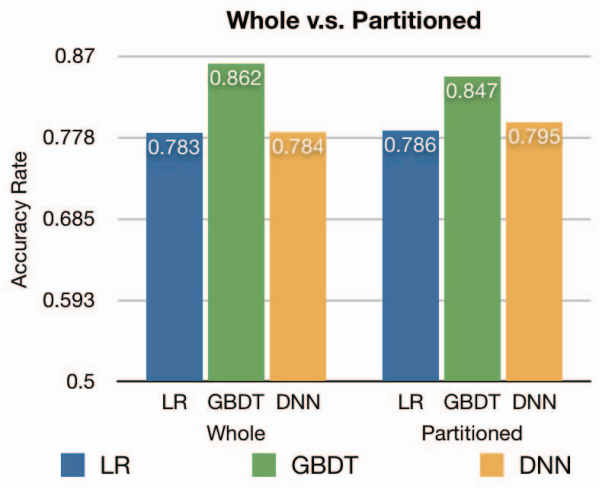

Fig. 15: Comparison of accuracy rates before and after sub-models partitioning.

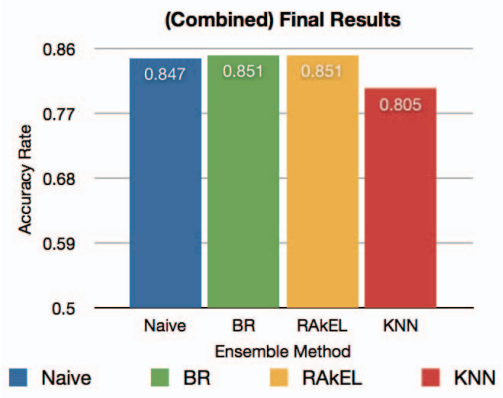

Fig. 16: Comparison of final accuracy rates between different multi-label ensemble algorithms and state-of-the-art KNN algorithm.

In the following experiments, we arbitrarily choose five minutes to be the length of the time span.

B. Hypothesis: Reader-based Sub-models and Whole Model Must Yield Comparable Final Accuracy Rates

In this experiment we compare the accuracy rate of the reader-based sub-models together and that of the whole model. The naive ensemble method is employed here to get the final accuracy rates. The result is shown in Figure 15. As expected, they yield comparable accuracy rates.

Under the condition that partitioned and whole models have similar accuracy rates, training time is the key factor of choosing which model to use. With reader-based sub-models training, each reader-based sub-model trains more than ten times faster than the whole model in the case of GBDT, for example, which makes our model particularly valuable. Furthermore, nothing stops us from parallelizing the training of each sub-model. In this way, the training time can be further reduced by a factor of the number of readers.

C. Observation: GBDT Outperforms LR and DNN

In order to decide which algorithm we should use to train our reader-based sub-models, we compare the results of LR, GBDT, and DNN as shown in Figure 15. Apparently, GBDT performs the best among three algorithms. Therefore, we apply GBDT as our reader-based sub-models classifier for the following experiments.

D. Observation: Our Model Outperforms State-of-the-art KNN Model

Here, we show the final accuracy rates of our model under different ensemble methods, along with the accuracy rate of the state-of-the-art KNN model, which is yielded by naive ensemble method.

As can be seen clearly in Figure 16, our model outperforms the KNN model significantly. While multi-label ensemble method does not seem to have a big influence on our model, it still improves the accuracy rate by a slight amount compared to naive ensemble method.

VII. Conclusions

In this paper, we present a machine-learning-based model that facilitates RFID location tracking. In our case, we use it to locate clothes in a clothes shop, which is a very common usage, but it is certainly not limited to this usage only. Instead, we expect it to generalize to any scenario that utilizes RFID location tracking.

Our study shows that the count features are essential to our model. Following this fact, we reckon that the number of readers and how they are arranged must have a significant influence on the performance of our model, because with more readers and a better arrangement of them, we can generate more detailed count features, which should make it easier for us to track the tags.

The model we propose beats the state-of-the-art KNN model significantly, meanwhile remaining time-efficient by employing reader-based sub-models training. This makes our

model readily scalable when more readers are deployed, which is highly desirable and valuable in real-world applications.

Finally, as our next step, we are going to further complicate our scenario by constantly moving clothes between shelves, allowing people to walk around freely, and adding some other interfering signal sources. It is foreseeable that in order to handle such complicated situation, not only should we deploy more readers to collect data, but we should also come up with a more sophisticated model.

ACKNOWLEDGMENT

This work is, in part, supported by Intel Corporation. We thank our colleagues from Intel Corporation who collected the data that greatly assisted the research.

In addition, this work is based upon work supported by Taiwan Ministry of Science and Technology (MOST) under grant number 104-2628-E-002 -015 -MY3 & 106-2218-E-002 -014 -MY4.

REFERENCES

[1] T. Hassan and S. Chatterjee, "A taxonomy for rfid," in *The 39th Hawaii International Conference on System Sciences*, 2006.

[2] J. Yang and Y. Chen, "Indoor localization using improved rss-based lateration methods," in *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*. IEEE, 2009, pp. 1–6.

[3] Z. Farid, R. Nordin, and M. Ismail, "Recent advances in wireless indoor localization techniques and system," *Journal of Computer Networks and Communications*, vol. 2013, 2013.

[4] M. Bouet and A. L. Dos Santos, "Rfid tags: Positioning principles and localization techniques," in *Wireless Days, 2008. WD'08. 1st IFIP*. IEEE, 2008, pp. 1–5.

[5] N. Chang, R. Rashidzadeh, and M. Ahmadi, "Robust indoor positioning using differential wi-fi access points," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 3, 2010.

[6] A. Bose and C. H. Foh, "A practical path loss model for indoor wifi positioning enhancement," in *Information, Communications & Signal Processing, 2007 6th International Conference on*. IEEE, 2007, pp. 1–5.

[7] A. Montaser and O. Moselhi, "{RFID} indoor location identification for construction projects," *Automation in Construction*, vol. 39, pp. 167 – 179, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S092658051300109X

[8] N. Li and B. Becerik-Gerber, "Performance-based evaluation of rfid-based indoor location sensing solutions for the built environment," *Advanced Engineering Informatics*, vol. 25, no. 3, pp. 535 – 546, 2011, special Section: Engineering informatics in port operations and logistics. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474034611000097

[9] C.-N. Huang and C.-T. Chan, "Zigbee-based indoor location system by k-nearest neighbor algorithm with weighted rssi," *Procedia Computer Science*, vol. 5, pp. 58 – 65, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050911003358

[10] N. Sihar and N. Balasbramaniam, "An indoor location sensing for wlan and zigbee integrated environment," *Electrical and Electronic Engineering*, vol. 5, no. 1A, pp. 23–27, 2015.

[11] X. Jiang, Y. Liu, and X. Wang, "An enhanced location estimation approach based on fingerprinting technique," in *Communications and Mobile Computing (CMC), 2010 International Conference on*, vol. 3. IEEE, 2010, pp. 424–427.

[12] S. C. G. Periaswamy, D. R. Thompson, and J. Di, "Fingerprinting rfid tags," *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 6, pp. 938–943, 2011.

[13] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil, "Landmarc: indoor location sensing using active rfid," *Wireless networks*, vol. 10, no. 6, pp. 701–710, 2004.

[14] 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*, vol. 3, p. 8, 2011.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[16] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[17] J. Read, P. Reutemann, B. Pfahringer, and G. Holmes, "MEKA: A multi-label/multi-target extension to Weka," *Journal of Machine Learning Research*, vol. 17, no. 21, pp. 1–5, 2016. [Online]. Available: http://jmlr.org/papers/v17/12-164.html

[18] 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*, vol. 11, no. 1, pp. 10–18, 2009.