Sensor Selection for Activity Classification at Smart Home Environments

Nithish Bolleddula[1], Geoffrey Yau Chun Hung[1], Daren Ma[1],
Hoda Noorian[1] and Diane Myung-kyung Woodbridge[2]

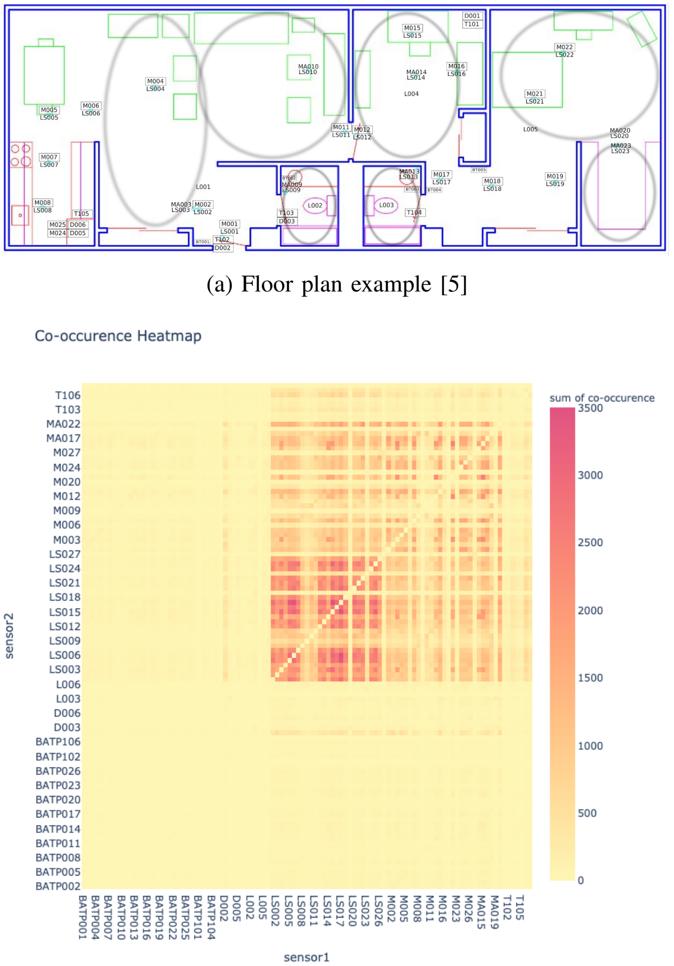
***Abstract*— As the world's older population grows dramatically, the needs of continuing care retirement communities increases. Studies show that privacy can be a major concern for adopting technologies, while the older population prefers smart homes [1]. In order to minimize the number of sensors to be installed in each house, we performed Principal Component Analysis (PCA) to filter out the relatively unimportant sensors. We applied a machine learning model to classify residents' activity types, using a different set of sensors chosen by PCA. Then, we validated the trade-off between the classification model accuracy and the number of sensors used in classification. Our experiment shows that feature engineering helps reduce accuracy degradation for activity type classification when using fewer sensors in smart homes.**

***Index Terms*— Smart homes, Privacy, Machine learning, Distributed computing**

I. Introduction

With the ongoing growth of the older population, smart home technologies became an emerging area [2]. Installing sensors in residences to monitor the behaviors of residents without limiting or disturbing their daily routine is considered as a functional environment to assist the well-being of residents unobtrusively and ubiquitously [3]. With a various number of sensors installed per home, rich real-life data are collected and analyzed to provide a detailed view of residents' daily behavior. Even though the sensor providers may encrypt all the data that are collected by the sensors, people still may feel reluctant to participate if sensors are placed in private places like bathrooms or too many sensors are being installed [4]. Therefore, minimizing the number of sensors while retaining maximum information gain is a critical problem for assisting residents promptly and minimizing privacy concerns.

In this research, we used the Human Activity Recognition from Continuous Ambient Sensor Data Set, collected and cleaned by Diane J. Cook et al. from Washington State University [5]. Each house has a floor plan similar to Figure 1a. The dataset consists of sensor data from 29 houses where volunteer residents stay. Based on different sensor types, each sensor will be triggered and send signals to the server. The dataset includes a timestamp, sensor id, room type and location, sensors type such as motion, light, door, temperature, etc. and activity class. The data schema follows the format below where the last field, 'activity', is a prediction label (Table I). We explored the co-occurence of the signals from different sensors within the same 5-min time window (Figure 1b). We could observe that there are certain sensors signals often occur together, which suggests us to reduce the number of sensors without losing too much information.



(a) Floor plan example [5]

(b) Co-occurrence plot demonstrating how sensors signal together within the same time window. Both axes are ordered by sensors and the values are the co-occurrence counts of two sensors within the time window.

Fig. 1: Residence floor plan and sensor co-occurrence

Each house has different data collection periods, which

[1] Nithish Bolleddula, Geoffrey Hung Daren Ma and Hoda Noorian are with the MS in Data Science Program, University of San Francisco. These authors contributed equally. {nbolleddula, yhung9, dma14, hnoorian}@dons.usfca.edu

[2] Diane Myung-kyung Woodbridge, Ph.D. is an assistant professor at the MS in Data Science Program, University of San Francisco. dwoodbridge@usfca.edu

TABLE I: Schema of collected data

timestamp	sensor_id	room-level location	detail	message	sensor type	activity
2011-06-15 00:06:32.834414	M021	Bedroom	Bed	On	Motion	Sleep

range from months to almost two years, and hence the data volume. The data size ranges from 100MB to 1GB, which requires efficient data preprocessing and machine learning algorithms. Given the nature of high frequency data from smart homes, the large data size makes it harder to develop and fine-tune machine learning algorithms on a single local machine. Distributed computing and cloud computing are used to process the data and develop machine learning models for large datasets. We applied distributed computing to process the data, Principal Component Analysis for selecting an optimal set of sensors and trained a machine learning model using Apache Spark [6] on Amazon Web Service (AWS) [7].

II. Related Work

The researchers who published the data set have mostly focused on the activity type classification itself so far, although they also discussed user privacy and technology usability concerns including sensor costs and acceptability by the general public. The research team relied on smart home sensors to monitor the behavioral states of the residents [8]. In another research, the researchers implemented a time series analysis for the study of inpatient rehabilitation [9]. The researchers drew the conclusion that the greatest amount of movement changes occur at the end of inpatient rehabilitation.

Human Activity Recognition (HAR) can provide useful solutions to many problems in elder-care and healthcare [10]. Extensive research [11] has focused on activity recognition using various sensor data for different scenarios. The steps can be broken down into preprocessing, feature extraction, and machine learning at a high level. Different studies have varying focuses, including learning models, near-real-time inference, and change-time point prediction [12].

III. Work Flow

A. Infrastructure

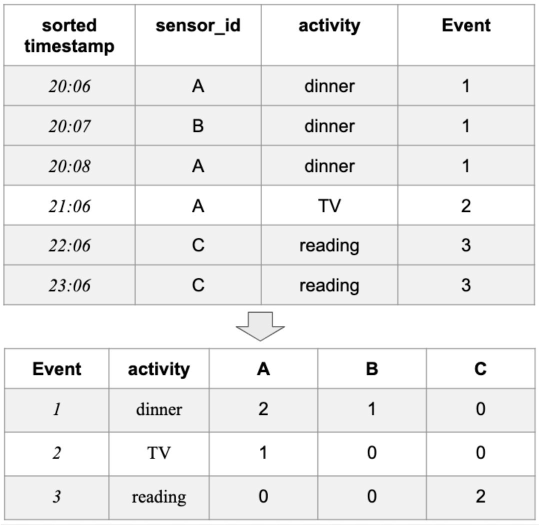
Amazon Web Services (AWS) is a platform that provides cost-effective storage and computing frameworks [7]. Since we are working with high-frequency data from sensors with high volume, the benefits of a scalable cloud service which is accessible become more critical [13]. We used this service for storing and processing the data.

For storing the raw sensor data, we utilized AWS Simple Storage Service (S3). S3 offers secure data transfer through access policy options that allows only authorized users to access the data. S3 also allows storing any type and size of an object with an option to replicate in case of data loss [14].

For applying preprocessing and machine learning algorithms, we utilized Apache Spark that distributes data across the network and processes it in parallel. We used AWS Elastic MapReduce (EMR) service as the distributed computing environment [15].

B. Data Pipeline

1) Distributed Data Preprocessing: The sensor log captures a resident's activities - any activities by residents may trigger multiple sensors and hence log data.



sorted timestamp	sensor_id	activity	Event
20:06	A	dinner	1
20:07	B	dinner	1
20:08	A	dinner	1
21:06	A	TV	2
22:06	C	reading	3
23:06	C	reading	3

Event	activity	A	B	C
1	dinner	2	1	0
2	TV	1	0	0
3	reading	0	0	2

Fig. 2: Pre-processing logic

Our first step is to group the log data that share the same activity along with the time interval as an event, and sort by timestamp. Figure 2 shows an example of preprocessed data. We have generated a number of features based on the work of Aminikhanghahi et al. [16]. We have only kept the features that are unique per window, in order to keep the data easy to handle after preprocessing. Since we have not implemented the transition point and change detection, we don't expect our classification performance to be comparable to their work, as the goal of our research is comparing the performance of models with all versus fewer sensors. For each window, we have extracted: minimum time of timestamp, maximum time of timestamp, duration of that activity in the given window, number of sensor events in the window, dominant location of the window, dominant location of the previous window, dominant sensor of the window, dominant sensor of the previous window and weekday as a number.

2) Principal Component Analysis: To minimize the number of sensors while retaining majority of information, we applied a dimension reduction technique, Principal Component Analysis (PCA) [17]. Resulting Principal Components (PCs) and its variance can help identify important sensors. We handle it by taking the absolute of PCs, multiply by the ratio of the variance, and then sum them up.

For each house, the data matrix $D_{n \times m}$ has n records and m sensors. With the application of Singular Value Decomposition (SVD), we are able to factorize the data matrix and perform Principal Component Analysis on top of that.

$$D_{n\times m} = U_{n\times n} \cdot S_{n\times m} \cdot PC_{m\times m}$$

where $PC_{m\times m} = (PC^{(1)}, PC^{(2)}, ..., PC^{(m)})$ is Right Singular Matrix of S having principal components from 1 to m, $S_{n\times m}$ = Rectangular matrix with singular values of D on diagonal, and $U_{n\times n}$ = Left Singular Matrix of S.

We denote $w^{(i)}$ as the variance explained by the i-th Principal Component, then we have a new vector w:

$$w = (w^{(1)}, w^{(2)}, ..., w^{(m)})$$

We define a novel Sensor Importance vector $I_{1\times m}$ as follows:

We consider first K principal components whose cumulative variance explained exceeds a constant value β. β is a hyper parameter which controls the number of principal components selected (K).

Once we select K principal components, we calculate Sensor importance vector as:

$$I_{1\times m} = \frac{\sum_{i=1}^{K} w^{(i)} \ |PC^{(i)}|}{\sum_{i=1}^{K} w^{(i)}}$$

i^{th} element of Sensor Importance Vector $I_{1\times m}$ represents importance of $sensor_i$. The intuition behind constructing the Sensor Importance Vector is to identify the sensors which contribute to the top-K principal components the most. This step yields a vector of size m where each entity approximates the importance of a specific sensor toward the PCs set. The vector is then sorted by their magnitude for sensor selection.

We select 25%, 50%, and 75% of the sensors based on the importances from Sensor Importance Vector $I_{1\times m}$, then rerun to the preprocessing pipeline to generate a new dataset only using selected sensors. This processed data is further used for building machine learning model. Cross-validating values of β against the accuracy of this model, $\beta = 0.8$ was chosen.

3) Distributed Machine Learning: We developed and fine-tuned machine learning algorithms of Random Forest [18], Naive Baye [19], Logistic Regression [20], a multi-layer feedforward Neural Networks [21] and XGBoost [22] to compare and choose the best model. The experiment output shows that tree-models including XGboost and Random Forest tend to have higher performance (Figure 3).

Random forest is an ensemble-based supervised learning algorithm that aggregates multiple decision trees [18]. The algorithm uses random sampling of training data when building trees and a random subset of features when splitting the nodes. This inherent randomness within the trees avoids overfitting issues complicit with deterministic decision trees, which allows the random forest to perform well without much of hyperparameter tuning. Additionally, the Random Forest algorithm is highly parallelizable, which is an excellent advantage in a distributed computing setting. Considering these advantages of Random Forest, we choose it for further purposes.

IV. Experiment Output

TABLE II: Runtime comparison on different cluster configurations

Total instances	Memory (GB) per instance	vCPU	# of data partitions	Runtime (secs)
i-5 processor	8	2	4	320
1	16	4	8	134
3	16	4	12	52
evening 5	16	4	20	44
3	32	8	24	42
5	32	8	40	34

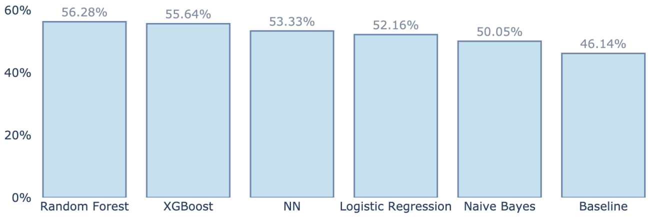
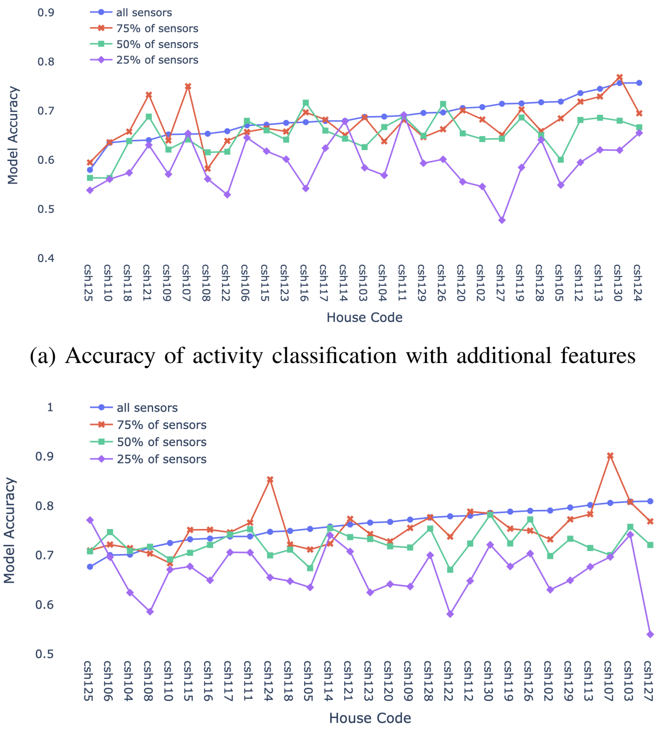


Fig. 3: Accuracy of different classification algorithms using 25% of sensor returned from PCA for house csh102 (without preprocessing). The baseline algorithm returns the most frequent class.

We deployed the developed preprocessing pipeline to several AWS EMR clusters with different configurations. The experiment results in Table II shows that an Apache Spark cluster with more worker nodes, memory, CPU, and data partitions yields the shortest execution time.



(a) Accuracy of activity classification with additional features

(b) Accuracy of activity classification with PCA with no additional features

Fig. 4: Accuracy of activity classification

After preprocessing and applying machine learning algorithms, the experiment results (Table III and Figure 4) show that the extracted features help minimize accuracy degradation. we observed a drop of around 3-4% in accuracy when using 50% of sensors instead of the full sensor set. In addition, eliminating 75% of sensors using the developed preprocessing with extracted features only causes less than 10% accuracy degradation.

The results also showed that the extracted features improve accuracy by around 7% consistently using the same set of data (Table III). Depending on the trade-off between accuracy and privacy/cost, fewer sensors option may be favorable in many use cases.

TABLE III: Average accuracy after data preprocessing and feature selection

% of sensors	Average accuracy with extracted features	Average accuracy without extracted features
25%	66.7%	59.4%
50%	72.4%	65.1%
75%	75.6%	67.4%
100%	76.1%	68.6%

V. Conclusion

In this paper, we demonstrated PCA as a sensor selection tool to identify important sensors from sensor log data without manual involvement. The same method can be scaled to different scenarios and data sizes. It is a trivial relationship that fewer sensors result in accuracy degradation, but the empirical experiment suggests even with 25% of original sensors, it only causes less than a 10% drop. With feature engineering, we can improve the accuracy by 7%. This suggests fewer sensors approach is possible to give reasonable accuracy. For future work, we plan to use advanced sensor data segmentation, transition detection and time series analysis to add more temporal features to the current model while minimizing the number of sensors installed.

At last, we demonstrated how distributed computing can significantly reduce the computation time needed. Our whole pipelines including preprocessing, feature extraction, PCA and machine learning prediction are all developed in Apache Spark which can be scaled easily with AWS EMR. Even the incoming data is streaming sensor data, we can extend the current pipeline to cater for the needs using Spark Streaming.

References

[1] K. L. Courtney, G. Demeris, M. Rantz, and M. Skubic, "Needing smart home technologies: the perspectives of older adults in continuing care retirement communities." 2008.

[2] G. Demiris, B. K. Hensel, M. Skubic, and M. Rantz, "Senior residents' perceived need of and preferences for "smart home" sensor technologies," *International journal of technology assessment in health care*, vol. 24, no. 1, pp. 120–124, 2008.

[3] M. Chan, D. Estève, C. Escriba, and E. Campo, "A review of smart homes—present state and future challenges," *Computer methods and programs in biomedicine*, vol. 91, no. 1, pp. 55–81, 2008.

[4] M. Pol, F. van Nes, M. van Hartingsveldt, B. Buurman, S. de Rooij, and B. Kröse, "P315: Older people's perspectives regarding the use of sensor monitoring in their home," *European Geriatric Medicine*, no. 5, pp. S180–S181, 2014.

[5] D. J. Cook, "Learning setting-generalized activity models for smart spaces," *IEEE intelligent systems*, vol. 2010, no. 99, p. 1, 2010.

[6] Apache Spark, "Apache spark: Lightning-fast cluster computing," 2020. [Online]. Available: http://spark.apache.org

[7] Amazon Web Service. (2020) Amazon. [Online]. Available: https://aws.amazon.com

[8] D. J. Cook, M. Schmitter-Edgecombe, L. Jönsson, and A. V. Morant, "Technology-enabled assessment of functional health," *IEEE reviews in biomedical engineering*, vol. 12, pp. 319–332, 2018.

[9] G. Sprint, D. Cook, D. Weeks, J. Dahmen, and A. La Fleur, "Analyzing sensor-based time series data to track changes in physical activity during inpatient rehabilitation," *Sensors*, vol. 17, no. 10, p. 2219, 2017.

[10] E. Kim, S. Helal, and D. Cook, "Human activity recognition and pattern discovery," *IEEE pervasive computing*, vol. 9, no. 1, pp. 48–53, 2009.

[11] E. De-La-Hoz-Franco, P. Ariza-Colpas, J. M. Quero, and M. Espinilla, "Sensor-based datasets for human activity recognition–a systematic review of literature," *IEEE Access*, vol. 6, pp. 59 192–59 210, 2018.

[12] C. A. Ronao and S.-B. Cho, "Recognizing human activities from smartphone sensors using hierarchical continuous hidden markov models," *International Journal of Distributed Sensor Networks*, vol. 13, no. 1, p. 1550147716683687, 2017.

[13] D. Fozoonmayeh, H. V. Le, E. Wittfoth, C. Geng, N. Ha, J. Wang, M. Vasilenko, Y. Ahn, and D. M.-k. Woodbridge, "A scalable smartwatch-based medication intake detection system using distributed machine learning," *Journal of Medical Systems*, vol. 44, no. 4, pp. 1–14, 2020.

[14] Amazon Web Services. (2020) Amazon s3. [Online]. Available: https://aws.amazon.com/s3/

[15] Amazon Web Services m. (2020) Amazon emr. [Online]. Available: https://aws.amazon.com/emr/

[16] S. Aminikhanghahi and D. J. Cook, "Enhancing activity recognition using cpd-based activity segmentation," *Pervasive and Mobile Computing*, vol. 53, pp. 75–89, 2019.

[17] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[18] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE transactions on systems, man, and cybernetics*, vol. 21, no. 3, pp. 660–674, 1991.

[19] I. Rish *et al.*, "An empirical study of the naive bayes classifier," in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, no. 22, 2001, pp. 41–46.

[20] J. S. Cramer, "The origins and development of the logit model," *Logit models from economics and other fields*, vol. 2003, pp. 1–19, 2003.

[21] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[22] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.