

The 35th ACM/SIGAPP Symposium on Applied Computing (SAC), Brno,
Czech Republic: Internet of Things Track

Embedded Machine Learning-Based Data Reduction In Application-Specific Constrained IoT Networks

Adarsh Pal Singh

Signal Processing and Communication Research Center
IIIT Hyderabad

March 31, 2020



INTERNATIONAL INSTITUTE OF
INFORMATION TECHNOLOGY

HYDERABAD

Paper Authors

Adarsh Pal Singh



5th Year, B.Tech. (Hons.) + MS by Research in
Electronics and Communication, IIIT Hyderabad
adarshpal.singh@research.iiit.ac.in

Dr. Sachin Chaudhari



Assistant Professor, SPCRC and IoT Lab, IIIT
Hyderabad
sachin.c@iiit.ac.in

Section I

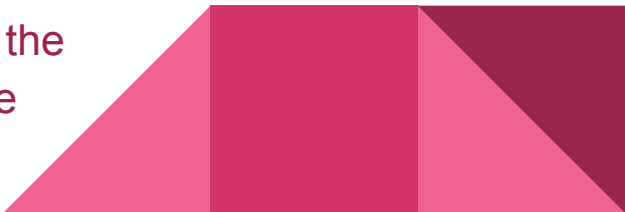
Basics, Motivation and Contributions

In a Nutshell...

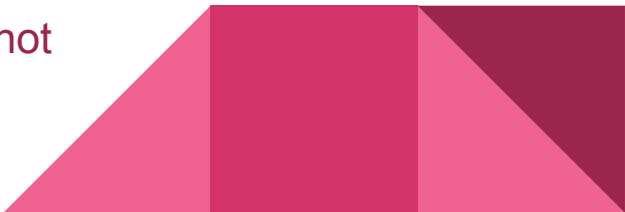
This research work introduces an embedded machine learning-based scheme to intelligently reduce the number of data transmissions from constrained wireless sensor nodes to the edge for application-specific IoT networks.



Motivation for Data Transmission Reduction

- Numerous IoT applications require WSNs to be deployed in constrained regions devoid of regular wired power supply.
 - The sensor nodes in such WSNs are battery-powered and generally comprise of a low-power microcontroller, sensor(s) and an RF transceiver.
 - These nodes typically follow a *sleep* → *wake-up* → *sense* → *transmit* → *sleep* pattern.
 - The radio transmission is known to be the biggest energy consuming activity!
- ∴ Increasing the lifetime of such sensor nodes by decreasing the amount of transmissions without compromising much on the data quality is an important area of research.
- 

So, What All Has Been Done Till Now?

- Low complexity data compression techniques to shorten the data packets.
 - Dual prediction models that run on both the sensor node and the edge and initiate transmission only when the predicted value exceeds the new sensed value by a predefined threshold.
 - Filter-based: Dual Kalman Filter, LMS Filter.
 - Time-series: Linear Regression, MA, EWMA, ARIMA, *Shewhart*.
 - For application-specific IoT networks, these techniques do not take data importance or the application into account when deciding the initiation of transmission!
- 

Shewhart Change Detection (Naive Algorithm)

- **Sensor Node:** Transmit the new data point x_t if it differs from the previously transmitted point by a predefined threshold ϵ .

if $|x_t - x_{prev_tx}| > \epsilon$, transmit x_t , $x_{prev_tx} = x_t$
else, discard x_t

- **Edge:** If no data point is received from the sensor node for a particular interval, consider the previously received data point to be true for that interval as well. Else, if new data point is received, consider that.

$$\hat{x}_t = \begin{cases} x_t & \text{if } x_t \text{ is received} \\ x_{prev_rx} & \text{otherwise} \end{cases}$$



Our Contributions

- A new embedded machine learning-based data transmission reduction scheme for application-specific IoT networks is introduced.
- An in-depth analysis of five supervised machine learning algorithms in the context of memory and computationally constrained microcontrollers is provided.
- The above scheme is validated on an actual occupancy estimation testbed and compared to Shewhart in terms of accuracy and the number of transmissions.



Section II

Machine Learning on Constrained Devices

Can Microcontrollers Run ML Algorithms?

- Supervised ML algorithms require substantial memory and computing resources in the training phase.
 - Many of these algorithms boil down to simple parameters that need standard arithmetic and logical operations for inference (testing) => can be run on microcontrollers!
 - LDA, QDA, GNB, SVM and DT exhibit low inference complexity.
- ∴ Train the ML model at the edge/cloud and offload the trained parameters to the microcontroller to run inference in real time.



Linear Discriminant Analysis (LDA)

- Linear classifier that assumes the class conditional distribution of data $P(X = x|Y = k)$ as a multivariate Gaussian. Predictions are made using Bayes' rule.

$$f_k(x) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T \Sigma^{-1}(x-\mu_k)} \quad P(Y = k|X = x_s) = \frac{f_k(x_s)P(y=k)}{\sum_{i=1}^K f_i(x_s)P(y=i)}$$

- For binary classification, inference boils down to: $x_s^T \cdot w \stackrel{?}{\geq} c$
where, $c = \ln \frac{P(y=0)}{P(y=1)} - \frac{1}{2}(\mu_0^T \Sigma^{-1} \mu_0 - \mu_1^T \Sigma^{-1} \mu_1)$ is a 4-byte floating-point constant and $w = \Sigma^{-1}[\mu_1 - \mu_0]$ is a $d \times 1$ floating-point vector.
- Multiplications: d
Additions/Subtractions: $d - 1$

Quadratic Discriminant Analysis (QDA)

- Quadratic decision boundary classifier has the same formulation as LDA except that Σ is calculated separately for each class. Predictions are made using Bayes' rule.

$$f_k(x) = \frac{1}{(2\pi)^{d/2} |\Sigma_k|^{1/2}} e^{-\frac{1}{2} (x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k)} \quad P(Y = k | X = x_s) = \frac{f_k(x_s) P(y=k)}{\sum_{i=1}^K f_i(x_s) P(y=i)}$$

- For binary classification, inference boils down to: $x_s^T \Sigma_{diff}^{-1} x_s + x_s^T w \geq_0^1 c$
 where, $c = 2 \ln \frac{P(y=0) |\Sigma_1|^{\frac{1}{2}}}{P(y=1) |\Sigma_0|^{\frac{1}{2}}} + \mu_1^T \Sigma_1^{-1} \mu_1 - \mu_0^T \Sigma_0^{-1} \mu_0$ is a 4-byte floating-point constant, $\Sigma_{diff}^{-1} = \Sigma_0^{-1} - \Sigma_1^{-1}$ is a $d \times d$ constant floating-point matrix and $w = 2 \Sigma_1^{-1} \mu_1 - 2 \Sigma_0^{-1} \mu_0$ is a constant $d \times 1$ floating-point vector.
- Multiplications: $d^2 + 2d$
 Additions/Subtractions: $d^2 + d - 1$

Gaussian Naive Bayes (GNB)

- Based on the Bayes theorem coupled with the “naive” assumption that all the d features of the dataset given the class label are mutually independent. Classifier formulation-

$$P(Y = y | X = x_s = (x_1, \dots, x_d)) = \frac{P(Y=y)P(x_1, \dots, x_d | y)}{P(x_1, \dots, x_d)} \quad \hat{Y} = \arg \max_y P(Y = y) \prod_{i=1}^d P(x_i | y)$$

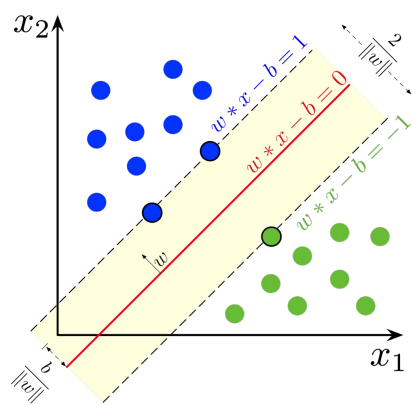
- For binary classification, inference boils down to: $\sum_{j=1}^d \rho_{0,j}^2 (x_j - \mu_{0,j})^2 - \sum_{i=1}^d \rho_{1,i}^2 (x_i - \mu_{1,i})^2 \stackrel{1}{\geq} c$
where, $c = 2 \ln \frac{P(y=0) \prod_{j=1}^d \frac{1}{\sqrt{\sigma_{0,j}^2}}}{P(y=1) \prod_{i=1}^d \frac{1}{\sqrt{\sigma_{1,i}^2}}}$ is a 4 byte floating-point constant, $\rho_0^2 = \frac{1}{\sigma_0^2}$ and $\rho_1^2 = \frac{1}{\sigma_1^2}$ are d -length floating-point arrays.

- Multiplications: $4d$
Additions/Subtractions: $4d - 1$

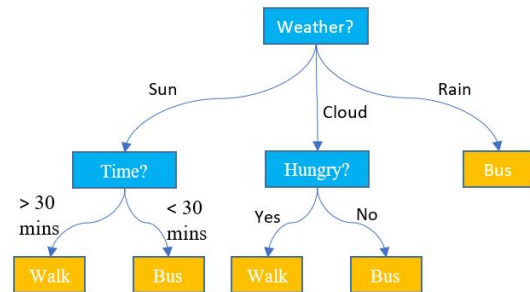


Support Vector Machine (SVM)

- SVM makes no assumptions about the data. It simply attempts to fit an optimal hyperplane that separates one class from the other.
- x_s first undergoes standard scaling: $\bar{x}_s[i] = p[i](x_s[i] - u[i])$ where u 's and p 's are floating-point arrays having means and inverse of standard deviations.
- Inference: $w^T \bar{x}_s \stackrel{1}{\geq}_0 b$ where b is a 4 byte floating-point constant (negative of hyperplane intercept) and w is a $d \times 1$ floating-point weight vector (coefficients of the hyperplane).
- Multiplications: $2d$
Additions/Subtractions: $2d - 1$



Decision Tree (DT)

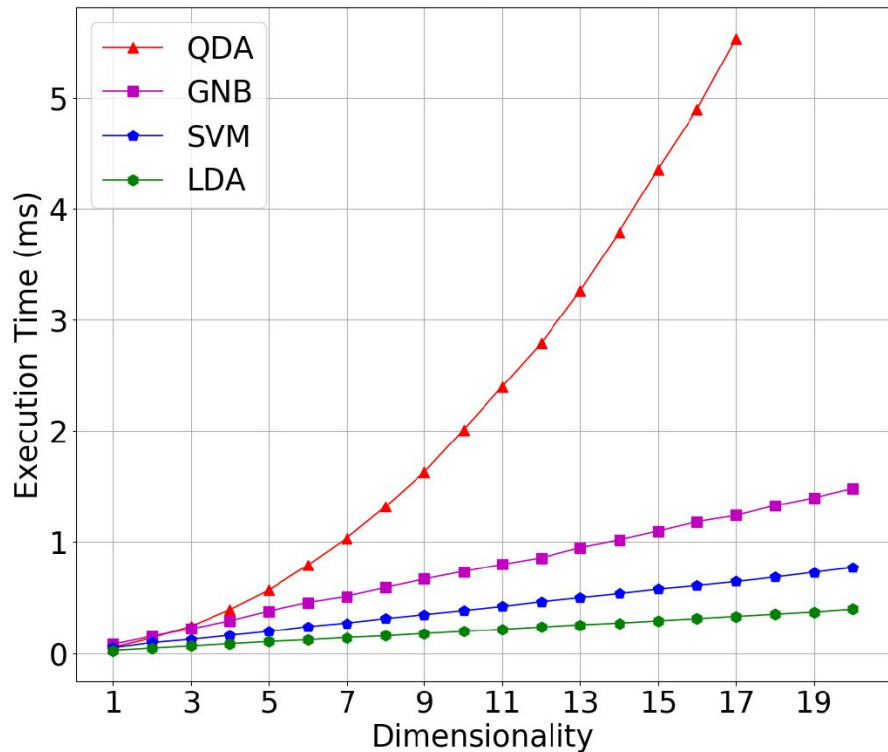


- DT learns optimum decision rules from the training set and discriminates between different classes based on these rules.
- A DT is an *if-else* ladder in the shape of a tree with branches feeding to logical test nodes and leaves corresponding to one of the classes.
- Each root to leaf path: *if condition₁ and condition₂ ... and condition_k then label*
- Memory complexity is dependent of the average depth of the tree and the number of leaf nodes: $4D_{avg}N_{leaf}$
- Time complexity dependent on the average number of logical operations needed for inference.

Complexities at a Glance

Algorithm	Multiplications	Additions	Bytes Offloaded
LDA	d	$d - 1$	$4d + 4$
QDA	$d^2 + 2d$	$d^2 + d - 1$	$4d^2 + 4d + 4$
GNB	$4d$	$4d - 1$	$16d + 4$
SVM	$2d$	$2d - 1$	$12d + 4$
DT	Nil	Nil	$4D_{\text{avg}} N_{\text{leaf}}$

Experimentation on ATmega328P

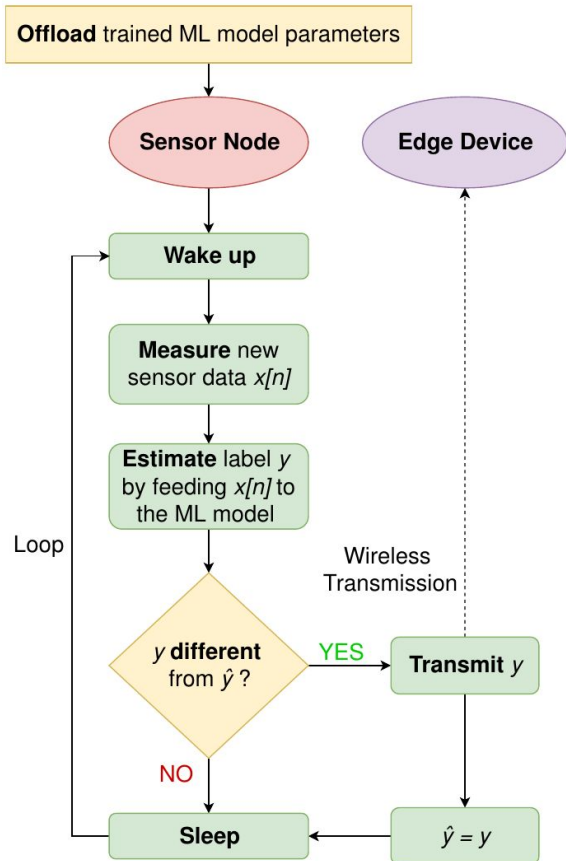


- Synthetic datasets of different dimensions ranging from 1 - 20 were created. Each was trained and then offloaded for testing.
- Execution times calculated by leveraging the internal $4\mu\text{s}$ resolution timer on ATmega328P.
- LDA is faster than SVM since it doesn't require standard scaling of data.
- Quadratic dependence of QDA on data dimension is clearly visible.
- QDA inference for $d > 17$ doesn't exist since ATmega328P hits its memory bottleneck.

Section III

Proposed Data Reduction Scheme

Algorithmic Flowchart



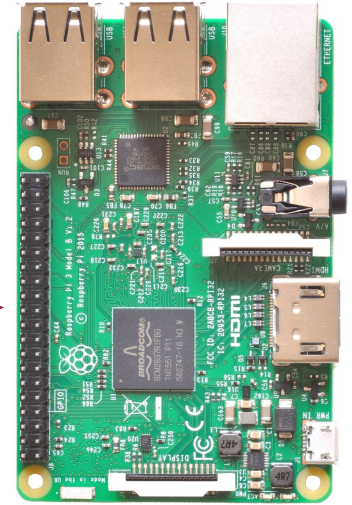
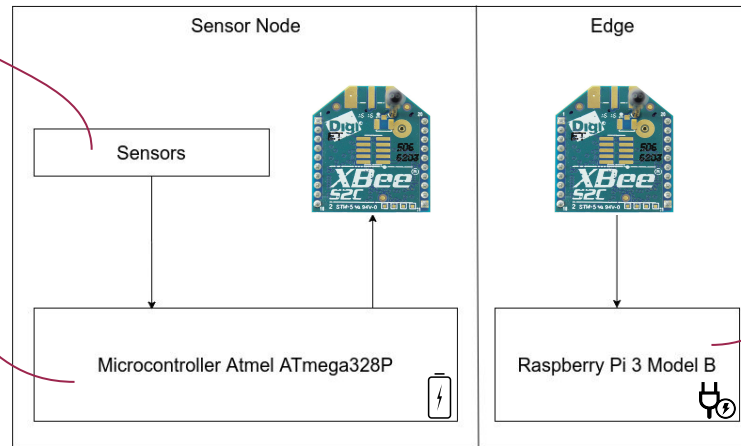
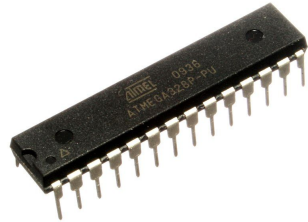
- Nodes typically follow a *sleep* → *wake-up* → *sense* → *transmit* → *sleep* pattern.
- First, acquire a training dataset and train one of the five ML algorithms at the edge/cloud.
- Offload the trained parameters and the inference algorithm to the nodes.
- Each time the node wakes up and collects new data, estimate the class label.
- If the class label is different from the previously transmitted label, transmit. Else, don't!

Section IV

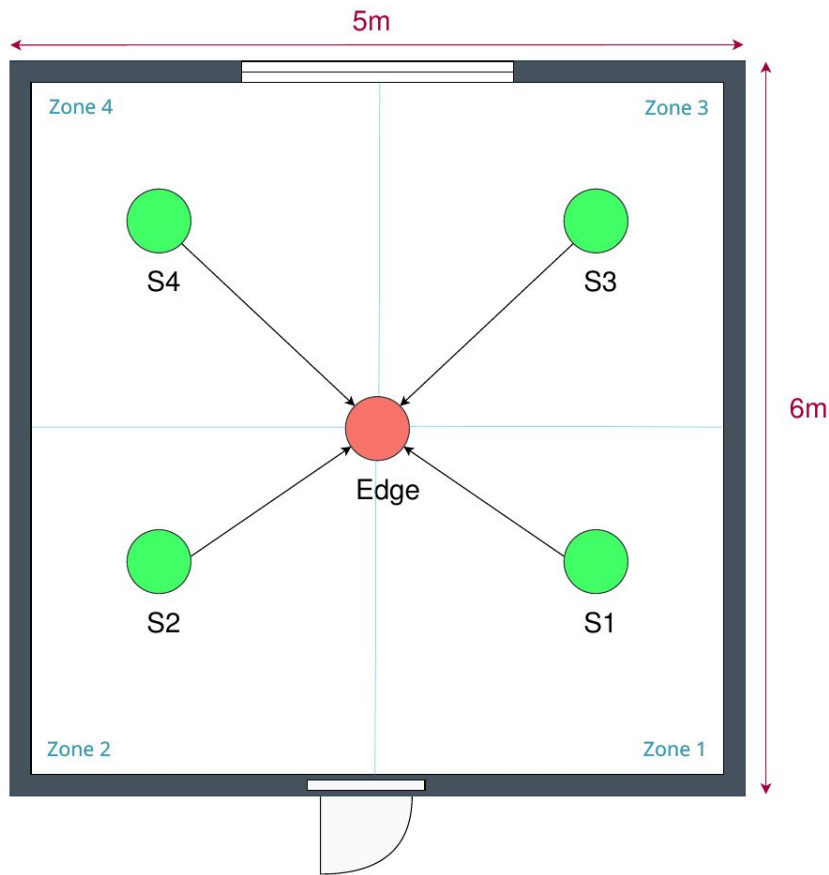
Occupancy Estimation Experimental Setup

Sensor Node and Edge Architecture

Sensor	Parameter	Resolution	Accuracy
DS18B20	Temperature	0.0625 °C	± 0.5 °C
DHT22	Humidity	0.1 %	2 %
BH1750	Light	1 Lux	1 Lux

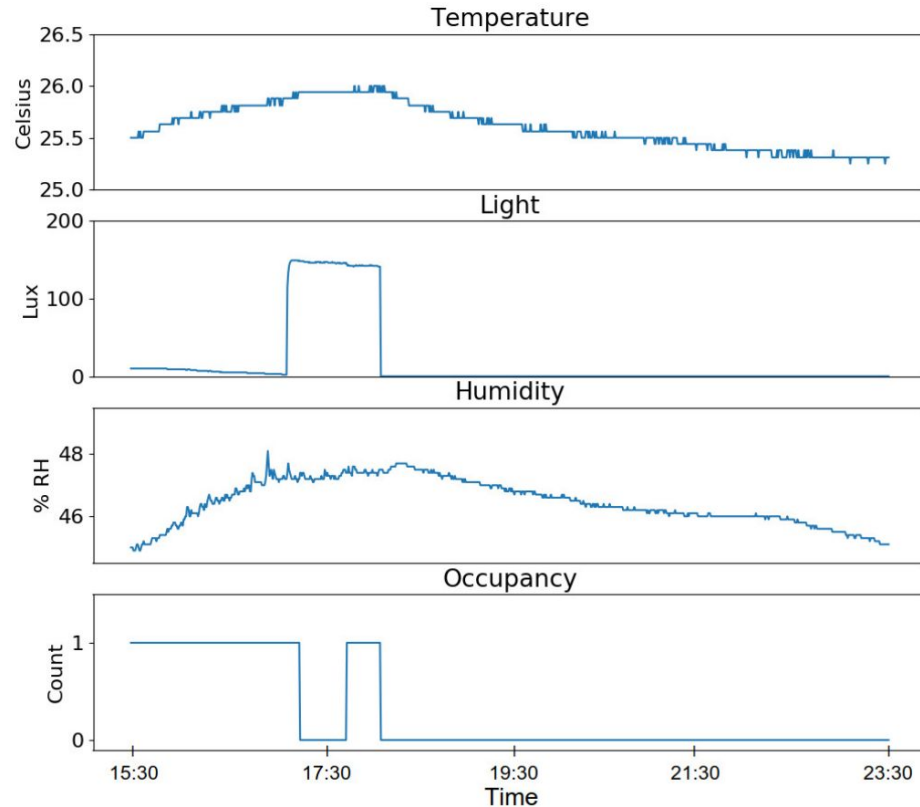


Experimental Setup



- A star WSN having 4 sensor nodes and one edge deployed in our lab.
- Sensor nodes programmed to transmit data every 30s.
- Edge programmed to timestamp and store the received data points.
- Occupancy in the room as well as in each zone recorded manually. Occupancy ranged from 0-3 with at most one person in a zone.
- Dataset has 10,000+ points pertaining to four days.

Dataset Visualization



- An 8-hour time-series plot of S1 data juxtaposed with the zone 1 occupancy count is shown.
- The correlations among different sensor and the occupancy can be observed.

Section V

Data Transmission Reduction for Occupancy Estimation Application

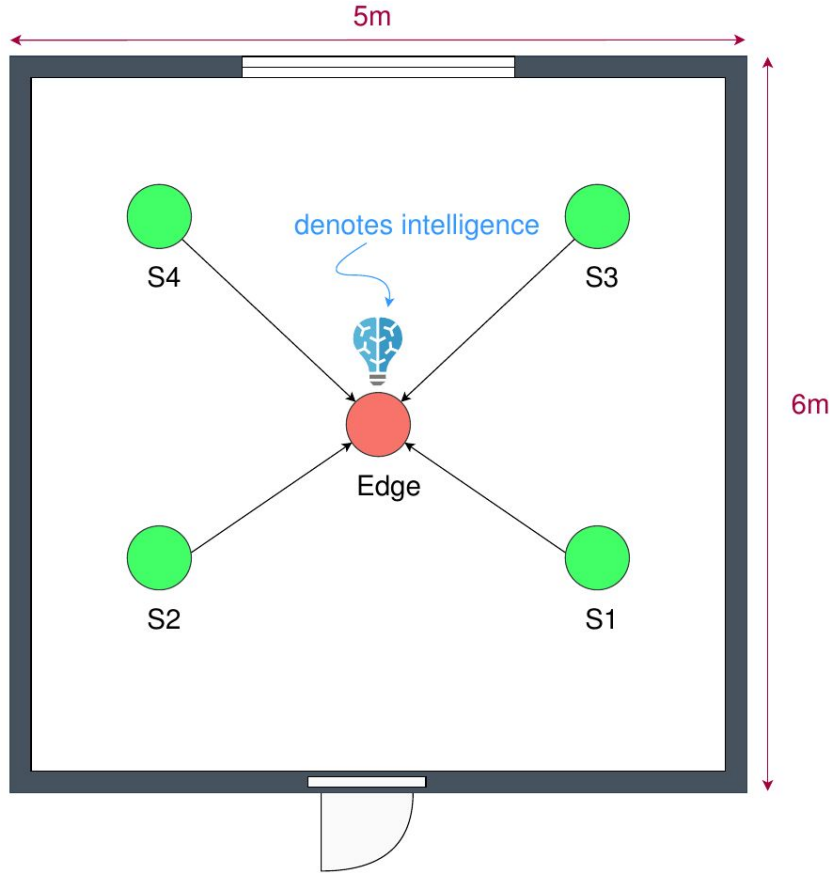
Data Reduction Experiments

Three different experiments were simulated on the collected dataset-

- **No Data Reduction:** ML inference for occupancy done at the edge using the complete dataset (**CD**).
- **Shewhart-Based Data Reduction:** Sensor nodes run Shewhart. ML inference for occupancy done at the edge using the Shewhart-reduced dataset (**SRD**).
- **ML-Based Data Reduction:** Each sensor node runs ML inference for occupancy on its own in a distributed detection (**DD**) fashion.

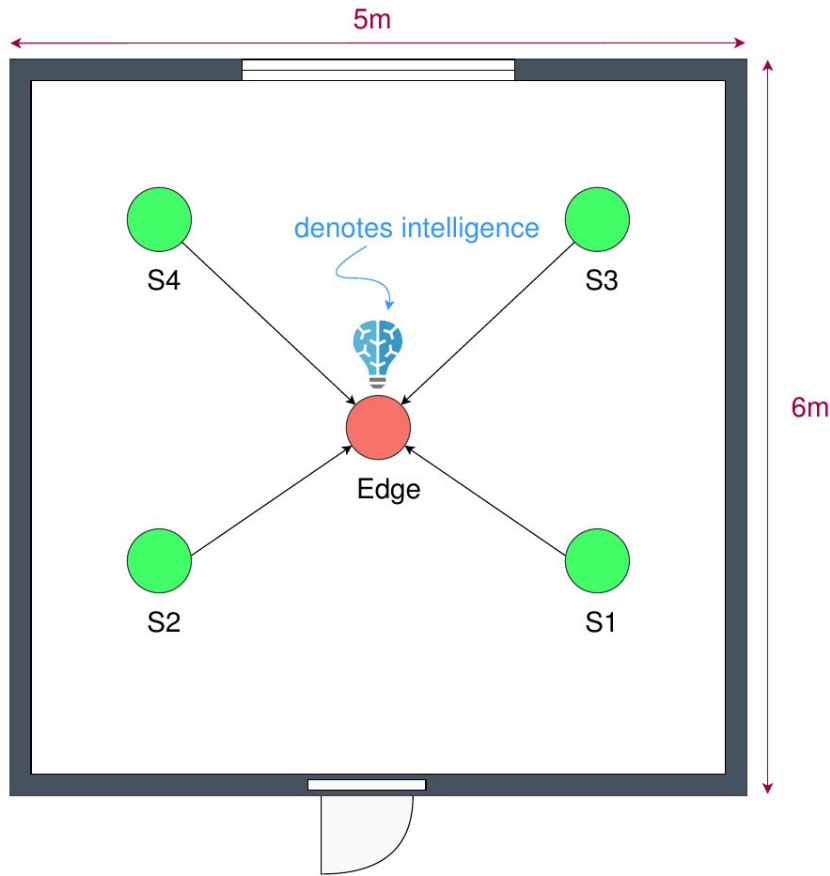


Experiment 1: Complete Dataset (CD)



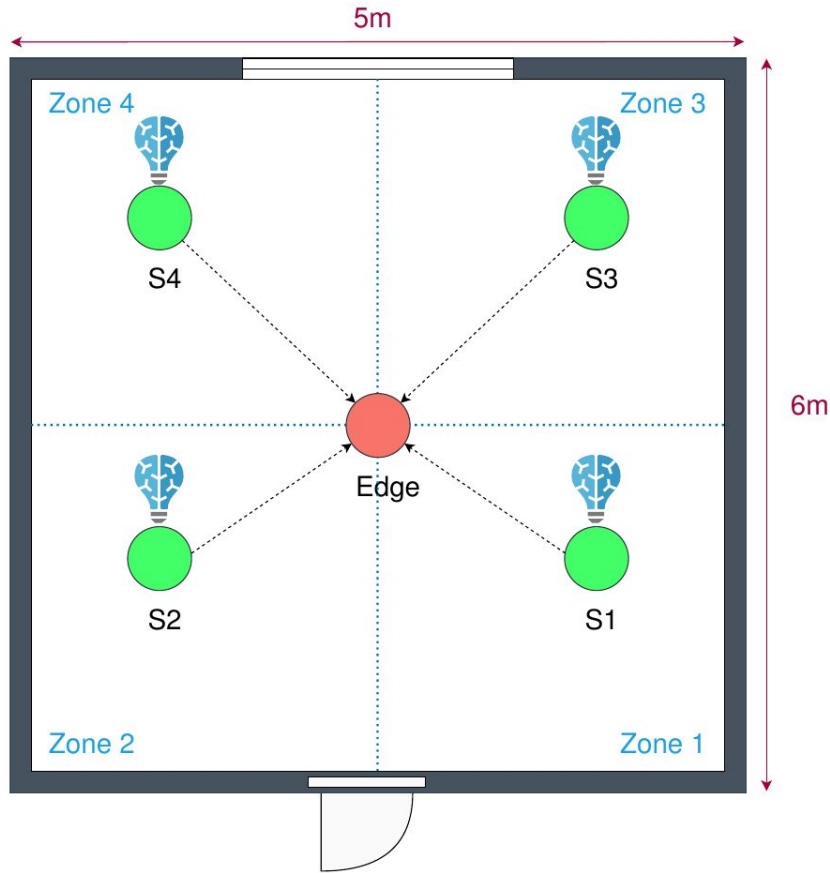
- Sensor nodes transmit data periodically without any data reduction scheme and the edge does the inference.
- Apart from the five ML algorithms, SVM (RBF) and RF were also applied.
- Accuracy and F1 score metrics were evaluated by doing a 10-fold stratified cross validation on the complete dataset.

Experiment 2: Shewhart Reduced Dataset (SRD)



- Sensor nodes run the Shewhart data reduction scheme and the edge does the inference.
- Shewhart thresholds are derived from sensor accuracy, resolution and mean of the data-
 - SRD1: $e_t = 0.5$, $e_h = 1.0$, $e_l = 1$
 - SRD2: $e_t = 0.5$, $e_h = 2.0$, $e_l = 1$
 - SRD3: $e_t = 0.5$, $e_h = 1.0$, $e_l = 10$
 - SRD4: $e_t = 0.5$, $e_h = 2.0$, $e_l = 10$
- Accuracy and F1 score metrics were evaluated by doing a 10-fold stratified cross validation on the complete dataset with Shewhart being simulated during the testing on each iteration.

Experiment 3: Distributed Detection (DD)



- Sensor nodes run our ML-based data reduction scheme with the edge maintaining the occupancy count of each zone.
- ML model of each sensor node was trained on the data from that sensor node and its corresponding zonal occupancy count.
- Accuracy and F1 score metrics were evaluated by doing a 10-fold stratified cross validation on each of the sensor node datasets. These predictions were summed up to get the occupancy count of the room.

Section VI

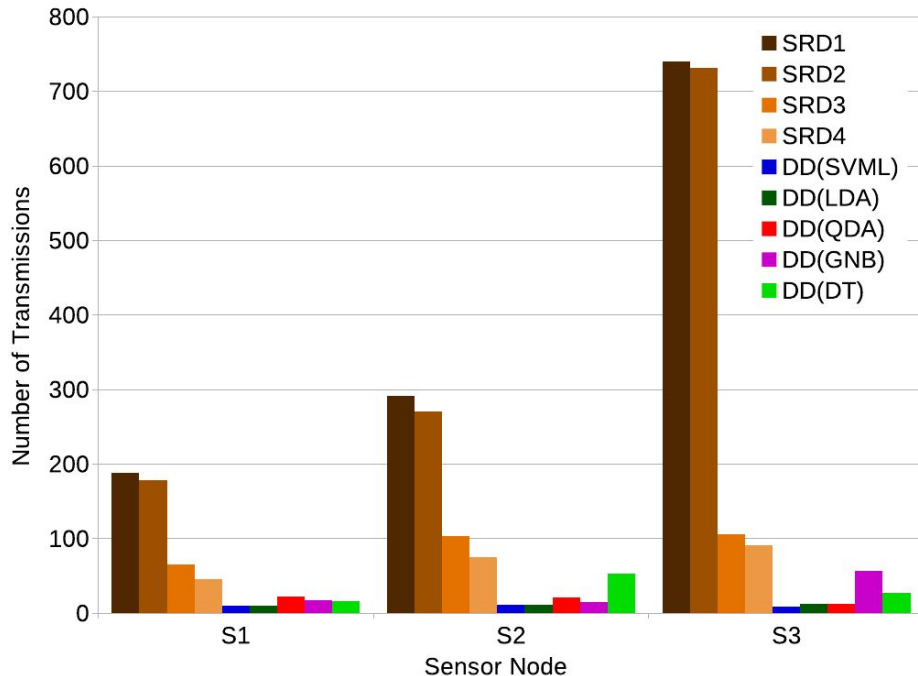
Results

Occupancy Estimation Accuracy and F1 Score

Experiment	Metric	SVM (RBF)	SVM (Lin)	LDA	QDA	GNB	DT	RF
CD	A	0.977	0.964	0.972	0.940	0.892	0.960	0.976
	F1	0.939	0.914	0.926	0.886	0.738	0.886	0.924
SRD1	A	0.972	0.962	0.972	0.852	0.872	0.921	0.969
	F1	0.928	0.909	0.926	0.815	0.717	0.825	0.907
SRD2	A	0.972	0.962	0.972	0.889	0.869	0.943	0.967
	F1	0.931	0.909	0.926	0.838	0.717	0.865	0.901
SRD3	A	0.970	0.960	0.972	0.786	0.866	0.905	0.964
	F1	0.922	0.903	0.924	0.781	0.721	0.807	0.891
SRD4	A	0.969	0.960	0.967	0.804	0.872	0.925	0.964
	F1	0.922	0.901	0.901	0.777	0.728	0.830	0.895
DD	A	N/A	0.968	0.965	0.933	0.915	0.890	N/A
	F1	N/A	0.916	0.907	0.812	0.759	0.743	N/A

- LDA outperforms Linear SVM => Gaussian assumption holds well.
- RF outperforms DT which is expected because of its ensemble nature.
- GNB performs the worst => Feature independence assumption doesn't hold well.
- SVM (RBF) on CD gives the highest F1 score of 0.939.
- For DD, best result is given by Linear SVM (0.916 F1 score) and the result is comparable to the best of others.

Transmission Reduction




- All five ML algorithms of DD outperform SRD schemes for each sensor node by a huge margin.
- S3 faces the most variation in light data. SRD schemes are clearly affected by fast changing variations.
- Linear SVM gives the least amount of transmissions: (10, 11, 9) out of 10,153 points.
- In general, ML schemes show 18 to 82 times reduction in transmissions as compared to Shewhart and 99.91% reduction overall in the best case while imparting similar performance.

Section VII

Conclusions and Future Work

Conclusions

- Offloading strategies for five traditional ML algorithms to memory and computationally constrained devices was described.
 - Based on this, a new ML-based data transmission reduction scheme was proposed for application-specific IoT networks.
 - The proposed scheme was validated on an occupancy estimation testbed.
 - ML-based data reduction schemes outperformed Shewhart-based schemes by reducing transmissions by 18 to 82 times whilst imparting similar performance.
 - Overall, 99.91% reduction in data transmissions was achieved in the case of Linear SVM.
- 

Future Work

- Adapting the ML-based data-transmission reduction scheme for more IoT applications.
- Optimizing more supervised and unsupervised learning algorithms for constrained devices.
- For resource-abundant devices like smartwatches and smart vehicles, more complex machine learning models can be used to intelligently prioritize data transmissions (device-edge-cloud collaboration).



References

1. **Prolonging the lifetime of wireless sensor networks using light-weight forecasting algorithms**, *IEEE 8th International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, 2013.
2. **A Survey About Prediction-Based Data Reduction in Wireless Sensor Networks**, *ACM Comput. Surv.* 49, 3, Article 58, Nov. 2016.
3. **Improving Accuracy of the Shewhart-based Data-Reduction in IoT Nodes using Piggybacking**, *IEEE 5th World Forum on Internet of Things (WF-IoT)*, 2019
4. **Adopting and Embedding Machine Learning Algorithms in Microcontroller for Weather Prediction**, *International Conference on Intelligent Systems (IS)*, 2018
5. **Real-time occupancy estimation using environmental parameters**, *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2015

