# Modelling Virtual Sensors for Indoor Environments with Machine Learning

Dawid Marek Polanski, Constantinos Marios Angelopoulos
*Department of Computing & Informatics*
*Bournemouth University*
Poole, United Kingdom
{i7708058, mangelopoulos}@bournemouth.ac.uk

***Abstract*—Virtual Sensors model the sensing operation of physical sensors deployed in an area of interest by generating sensory data with accuracy and precision close to those collected by physical sensors. Their use in applications such as augmenting the infrastructure of IoT facilities and test beds, monitoring and calibrating the operation of physical sensors, and developing Digital Twins of physical systems have led virtual sensors to attract research attention. Machine learning provides methods for modelling patterns in complex and big data generated by IoT sensing devices, allowing to model the behaviour of these devices. In this work, we investigate ML methods as means of implementation for virtual sensors. In particular, we evaluate the performance of six ML methods in terms of their effectiveness, accuracy and precision in generating sensory data based on data from physical sensors. In our study, we use a multi-modal dataset comprising IoT sensory data for temperature, humidity and illumination collected over a period of two years in an office space at University of Geneva. Our results show that the best performing model at predicting an output of a missing sensor is the Random Forest method, achieving MAPE error below 3%, 5% and 18% respectively for temperature, humidity and illuminance. The worst performing models were the linear radial basis function neural network and linear regression. In future research, we plan to deploy the best performing models natively on IoT devices, making use of tinyML and extreme edge computing methods.**

***Index Terms*—Internet of Things, Sensor, Virtual Sensor, Machine Learning**

## I. Introduction

Internet of Things (IoT) is a key enabling technology for innovative paradigms in several market and industry verticals, such as Industry 5.0, Smart Transportation, Digital Health and Smart Cities. IoT systems link the physical and cyber planes via sensing devices deployed *en masse* to monitor their immediate environment with the aim of collecting sensory data. In synergy with other enabling technologies, such as Artificial Intelligence (AI) and Machine Learning (ML), collected data are aggregated into complex, multi-modal and spatio-temporally dense datasets [1] used to improve other systems and processes. In the context of indoor environments, improvements typically pertain on improving ambient conditions and optimising the operation of corresponding systems, such as indoor lighting and air conditioning. Bearing in mind capital expenses and operational overheads, there is a need for developing novel evaluation and trialling methods for IoT systems that will consider limitations in the number of available physical resources, and will provide the needed agility and scalability [2].

### A. Applications of virtual sensors.

Virtual sensors are used to address these challenges and enable such methods. Virtual sensors are software-based devices modelling *in silica* the sensing functionalities of physical IoT devices. The aim is for these virtual sensors to generate simulated sensory data with accuracy and precision as close as possible to those of the physical sensors. This is achieved by developing methods that consider sensory data collected by physical devices actually deployed in the area of interest.

The advantages of using virtual sensors have promoted their use in several settings. One of their first applications was in the context of *experimenting facilities and testbeds*. Here, virtual sensors are used to augment the physical infrastructure of a testbed, thus facilitating the evaluation of developed solutions in terms of scalability. Furthermore, virtual sensors enable the concurrent execution of multiple experiments, thus allowing to increase the utilisation rate of the facility while providing a better experience to their users. A second application area of virtual sensors is *to monitor or calibrate the operation of deployed physical sensors*. Assuming that an accurate model has been derived that is able to generate sensory data of the same accuracy and precision as a set of physical sensing devices deployed in an area of interest, virtual sensors operating in tandem with physical ones can help identify deviations or discrepancies in sensory data that may indicate a miscalibrated or faulty device. This way preventive maintenance can take place, either via software update or by replacing the physical device.

Another important application of virtual sensors are *Digital Twins*. Digital Twins act as virtual representations of physical systems. They are used as digital counterparts for physical systems of varying scale and complexity, ranging from individual industrial parts to complex systems of systems, such as production lines and smart cities [3]. The operation of a Digital Twin relies on data characterising the physical system it virtualises. Initially, this data was provided in the form of datasets compiled in non-dynamic ways; for example, data from LiDAR scans providing the exact locations of city infrastructure. However, the roll-out of IoT has enabled the development of dynamic *digital threads* that allow the

2325-2944/22/$31.00 ©2022 IEEE
DOI 10.1109/DCOSS54816.2022.00046

continuous flow of data between the physical system and its Digital Twin. Virtual sensors are used in this context in order to provision the deployment of additional sensing devices or in order to develop hybrid cyber-physical components traversing the physical and digital planes. Such methods are highly relevant in state-of-the-art paradigms, such as Industry 5.0, Smart Agriculture and Smart Circular Economy.

Virtual sensors are also used to improve the operation and increase the resilience of deployed IoT systems. Assuming an accurate sensing model characterising the monitored process or phenomenon and a set of deployed physical devices, virtual sensors can increase the coverage density of the area of interest by inferring sensory measurements based on data collected by the deployed physical sensors. In such configurations, virtual sensors can also be used to "substitute" any malfunctioning physical sensors until their actual replacement, thus maintaining Quality of Service. Finally, virtual sensors can be deployed in the case of cyber-attacks, acting as "honey pots" to attract the attention of the adversary, effectively protecting the physical devices.

### *B. Eliciting accurate models for virtual sensors.*

Eliciting accurate sensing models is crucial for the successful deployment of virtual sensors. A successful sensing model should provide the mechanisms for generating sensory data that match as closely as possible sensory data generated by physical sensors deployed in the area of interest. One method of eliciting such a model would be to formally characterise the process or phenomenon under study; for instance, in the case of indoor illuminance to use a rigorous Physics model characterising light propagation in space. However, this would not suffice. The virtual sensing model would also need to consider the specificities of the area of interest; for instance, the orientation and layout of the building, the locations of any reflecting surfaces, the time of the day, season, etc. This is a copious task, that is not easily transferable to different settings or different modalities.

Machine Learning (ML) methods enable modelling and synthesising large and complex data such as those generated by sensory devices. ML can capture more complex underlying relationships across a variety of variables which may be more lacking in conventional modelling techniques. Models trained on actual sensory data can replicate or extrapolate virtual sensor data in form of virtual sensor systems, where multiple virtual sensors can be automatically generated based on the variable inputs specified by the user such as a state at a specific time and location [1], [2]. More importantly, ML methods provide a generic methodology that is highly transferable in different settings and applicable on different use-case scenarios considering various modalities.

**Our Contribution.** We investigate ML methods as means of implementation for virtual sensors. In particular, we evaluate the performance of six methods in terms of their effectiveness, accuracy and precision. We focus on use case scenarios where virtual sensors are employed to augment or complement physical sensors monitoring ambient environment conditions (illuminance, humidity and temperature) in indoor environments. In this context, we make use of a rich dataset comprising sensory readings spanning a 2-year period from an IoT system deployed in an office space of University of Geneva [2]. Our findings indicate that ensemble methods, such as Random Forest, are the most appropriate ones for such use case scenarios as they exhibit low mean absolute percentage errors below 5% for temperature and humidity, and below 18% for illuminance.

## II. Related Work

Monitoring of environmental data, in particular office conditions data is of interest to companies for optimising working conditions and to adhere to workplace health and safety regulations which dictate them. Being able to monitor the temperature, illumination and humidity can help optimise costs in running the offices. The use of machine learning for predicting room occupancy for reduction in electrical usage has been demonstrated in Peng et al. [4]. Use of virtual IoT in buildings is a relatively new application with initial research performed by Li and Braun [5] with their implementation of virtual refrigerant sensors for indoor air-conditioning. The indoor illumination was studied by Park and Athienitis [6] using linear correlation and by Drakoulelis et al. [1] using virtual sensors and machine learning methods with good results. The work by Sendorek et al. [7] outlines an IoT system that is developed inside a virtual environment allowing for reproduction of environmental conditions in the real world. This virtual system allows for simulation of physical environments, devices and sensors and removes the dependence on hardware, while still sustaining the sensor layers for testing. Work by Mattera et al. [8] demonstrates virtual sensors for monitoring faults of ventilation units. Their approach consisted using machine learning methods based on linear regression, support vector machines, artificial neural networks algorithms to model the temperature, airflow and fan speed to create virtual sensor data. By measuring the deviations between the physical and the virtual sensors their estimates were able to correctly determine anomalous behaviour. This project aims to cover the gap in research Kundig et al. and Drakoulelis et al. by extending the machine learning modelling for virtual testbeds to not only the illumination but to humidity and temperature as well over longer periods of time.

## III. Description of the Data Set

The dataset used for training and evaluating the learning models for the virtual sensors was first published in [2]. The UNIGE dataset consists of IoT sensory data generated from the Syndesi testbed, which is a smart building IoT testbed comprising various, IoT devices, sensors, and actuators. The aim of the Syndesi testbed was to provision a smart environment consisting of sensor networks, electrical devices, actuators, gateways and communication technologies [2]. The updated version of the testbed was expanded to also include crowdsourced resources (in the form of smartphones provided by the end users), and its back-end architecture was redesigned

as a System-as-a-Service with multiple entry points utilising RESTful APIs for interoperability between testbed devices and external services [9], [10].

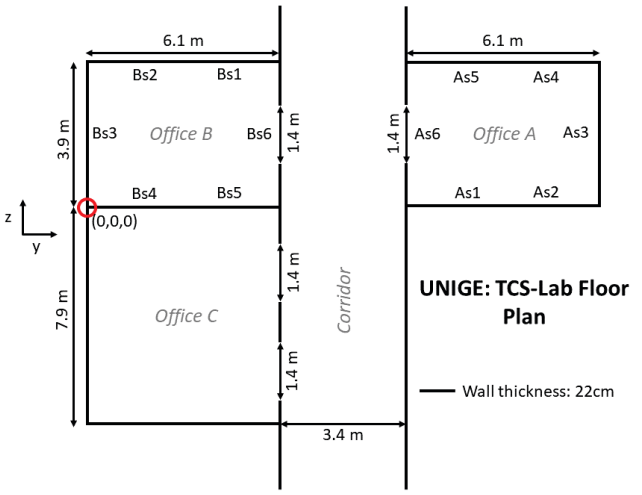


Fig. 1. Floor Plan of the UNIGE Offices

The dataset comprises data of illumination, temperature and humidity readings for 2 offices with 6 IoT sensing devices in each with their x-y-z coordinates relative to a reference point (Fig. 8). The data was collected over a period of two years with approximately 20 minutes separation between measurements for each sensor. The dataset consists of: 139,715 measurements for office A humidity and temperature; 198,880 measurements for office A illuminance; 181,658 measurements for Office B humidity; 181,659 measurements for Office B temperature; and 242,750 measurements for Office B illuminance [2].

## IV. Methodology

In order to ensure the generality of the model predictions, a k-fold cross-validation was performed. The data was divided by office rooms and measurements, and then it was split into 6 folds; 1 for each sensor to estimate how accurately the missing validation data can be predicted. The machine learning methods that were evaluated are linear regression, K-Nearest Neighbours, gradient boosting tree algorithms XGBoost and LightGBM, decision-tree ensemble classifier Random Forest, Support Vector Machines and a Neural Network with an RBF layer. The performance of the machine learning algorithms was evaluated using mean absolute error (MAE), mean square error (MSE), root mean square error (RMSE) and mean absolute percentage error (MAPE). The modelling has been performed using scikit-learn, LightGBM, XGBoost and Tensorflow packages on python version 3.9.7.

### A. Data Sampling

Cross validation (CV) was used to evaluate the predictive validity of the derived models, and to partition the data into multiple sets of differing training and test data. Those partitioned datasets were then modelled and compared in order to determine if the learning models are subject to overfitting, and how well they generalise from the data [11]. K-fold cross validation is one of the typically used techniques for model selection and estimating errors of classifiers. K-Fold cross validation works by splitting the dataset into a k number of folds where K-2 number of subsets of data are used for training and the remaining two are used for model selection and error estimation. Usually K values of 5, 10 or 20 are used in CV, with larger numbers allowing for greater generalisation. However, this incurs a cost of requiring more computational resources and the data being shuffled as the process is repeated [12]. For this work, data was separated by modality - measurements of humidity, temperature and illuminance - and then by offices A and B. Then, one sensor out of 6 was arbitrarily chosen to be removed, and the remaining sensors were then used as training data. The models were then evaluated based on their capability to predict the output of the missing sensor.

### B. Overview of Machine Learning Algorithms

The type of **linear regression (LR)** that was used in this work was ordinary least squares regression. The scikit-learn linear model uses coefficient fitting in order to produce the smallest residual sum of square of the observations in the datasets, where predictions are obtained as linear approximates. The linear regressor was used in its base form using 'deprecated' normalisation [13].

**K-Nearest Neighbours (KNN)** is a simple algorithm which provides predictions based on the commonalities in the data. The KNN works based on the principle that data is arranged in space based on specific features. When making a prediction based on new data the algorithm will provide a regression output based on the proximity to modelled data points [14]. In this case in order to capture higher complexities in the data a value of 25 neighbours was used.

**Random Forest** is an ensemble classifier based on decision tree algorithms. A Random Forest can be defined as a classifier that consists of a collection decision tree-classifiers where each tree casts a unit vote for the most popular class at the input [15]. The Random Forest implementation in Scikit-Learn is based on the classification and regression tree algorithm (CART). The parameters used for the Random Forest classifier were 100 estimators, 'gini' criterion, minimum sample split of 2 and minimum sample leaf of 1.

The **XGBoost** is an open-source library for gradient boosted trees which is available for supervised classification and regression problems. It is based on CART decision tree ensembles allowing for multiple trees being used for prediction. It improves further on the gradient boosting trees by incorporating novel-sparsity awareness, effective cache-ware block structure for out-of-core tree learning and parallel computation. XGBoost implements exact greedy and approximate algorithms to find the best split for the trees [16]. The XGB regressor was used in its base form with 500 estimators.

**LightGBM** is an open-source library for gradient boosted trees like XGBoost, but it offers significant improvements in terms of speed and memory usage. The LightGBM algorithm uses two novel techniques, gradient-based one-side sampling (GOSS) and exclusive feature bundling (EFB) to deal with the larger amount of data. GOSS allows for discarding instances with small data gradients where the training error is small, and keeps all the large sample gradients and randomly samples the instances with small gradients. To compensate for the data distribution imbalance a constant multiplier is used for the small gradients. This method allows GOSS to focus on the under-trained instances. The EFB works by partitioning exclusive features into a single feature by continuously scanning data features reducing the training time [17]. The LightGBM regressor was used in its base form with 500 estimators.

**Support Vector Machines (SVM)** are a form of a supervised learning model that works based on splitting two dimensional planes. The margin value C acts as a threshold, which is used to remove the values around it. The prediction ability of SVM relies on the type of kernel and its parameters and the C threshold values. The support vector machine regression attempts to reduce the error function value while ignoring the values close to the threshold [1]. The SVR was used in its base form with and 'rbf' kernel, C value of 1.0 and epsilon of 0.1.

**Neural networks (NN)** are relying on the propagation of data between weighted connections between a large number of non-linear elements called neurons [18]. The **radial-basis function neural network (RBFNN)** allows for transformation of the data in high-dimensional spaces for better linear separation. The hidden layer of the RBFNN computes the parameters by local approximation. The RBFNNs have the ability to perform nonlinear fitting which can map nonlinear patterns and are better at finding the global minimum when compared to standard neural networks [19]. The architecture of the neural network consisted of two units of Dense layer with 256 neurons and ReLu activation, followed by an RBF layer with 256 neurons and 0.5 gamma and Dropout layer with dropout value of 0.5. Then the final layers consisted of a dense layer of 256 neurons with an output layer of 1 neuron, both with ReLU activation.

### C. Overview of Evaluation Metrics

The following performance evaluation metrics were used to evaluate the trained learning models. The *mean absolute error* is a measure of the mean of absolute difference between the predictions and the observed values with equal weight for all calculated differences [20]. The *mean absolute square error* is calculated by squaring the absolute difference between predictions and the observed values and then calculating the mean. The *root mean square error* is computed by calculating the root of the square error [21]. *Mean absolute percentage error* is obtained by averaging the ratios of the absolute value of the difference between the predicted values and the observation divided by the value of the observation [22].

## V. Performance Evaluation

This section presents the evaluation results modelling the virtual sensors. Figures 2 through 5 depict box-plots of the errors for all three measurements at both offices, and figures 6 through 8 depict heatmaps of the extrapolated values inside office B at a randomly selected date of 17 of March 2016.

### A. Results on Humidity

Tables 1 and 2 show the results for the humidity predictions in Offices A and B based on training data from each corresponding room. The best performing machine learning algorithm on average in both offices was Random Forest with MAPE values below 5%. The worst performing model was RBFNN with the highest levels of MAPE. The decision tree-based models have achieved generally the best performance in the humidity tests resulting overall with the smallest amounts of accumulated errors.

TABLE I
Office A Humidity Results

| Model | MAE | MSE | RMSE | MAPE |
|---|---|---|---|---|
| LR | 4.198027 | 29.125630 | 5.377169 | 13.509100 |
| KNN | 3.731337 | 22.426156 | 4.683310 | 12.357764 |
| RF | **1.163870** | 2.944406 | 1.623279 | **3.648200** |
| XGB | 1.233387 | **2.363988** | **1.515551** | 3.918226 |
| LGBM | 1.287947 | 2.787845 | 1.622912 | 4.068255 |
| RBFNN | 5.387845 | 45.742973 | 6.707297 | 17.467788 |
| SVR | 5.205127 | 42.809196 | 6.476049 | 16.93824 |

TABLE II
Office B Humidity Results

| Model | MAE | MSE | RMSE | MAPE |
|---|---|---|---|---|
| LR | 5.767395 | 50.011194 | 7.070313 | 21.053756 |
| KNN | 3.698532 | 21.141502 | 4.590419 | 13.521766 |
| RF | **1.341465** | **2.599065** | **1.488219** | **4.700466** |
| XGB | 1.498381 | 3.361951 | 1.729482 | 5.271875 |
| LGBM | 1.622413 | 4.043927 | 1.924138 | 5.759617 |
| RBFNN | 5.984802 | 54.022779 | 7.343447 | 21.799352 |
| SVR | 5.797765 | 51.232564 | 7.15558 | 20.755587 |

### B. Results on Temperature

Temperature modelling had two best performing models, Random Forest and LGBM; Random Forest achieved a better result in office B, whereas the LightGBM regressor performed better at office A. The lowest scoring models were RBFNN and SVR for both offices A and B. The overall error for the models appears to be the lowest across models for the temperature measurement resulting in MAPE's below 10%.

### C. Results on Illuminance

Predictions for illumination values display the highest accumulation in error across all metrics when compared to both temperature and humidity. The algorithms with best predictive capacities were KNN and LGBM for Office A and Random Forest for Office B, attaining the highest results for all metrics. The worst performing model was the linear regressor, with the lowest scores across all metrics. The overall error across

TABLE III
OFFICE A TEMPERATURE RESULTS

| Model | MAE | MSE | RMSE | MAPE |
|---|---|---|---|---|
| LR | 1.914671 | 6.426534 | 2.525238 | 6.805640 |
| KNN | 1.984283 | 6.387426 | 2.505278 | 7.190211 |
| RF | 0.681660 | 0.777214 | 0.841143 | 2.444063 |
| XGB | 0.564874 | 0.536739 | 0.683781 | 2.021858 |
| LGBM | **0.549936** | **0.513543** | **0.679965** | **1.967692** |
| RBFNN | 2.445299 | 9.819664 | 3.108505 | 8.790054 |
| SVR | 2.44263 | 9.79115 | 3.121935 | 8.746947 |

TABLE IV
OFFICE B TEMPERATURE RESULTS

| Model | MAE | MSE | RMSE | MAPE |
|---|---|---|---|---|
| LR | 2.366994 | 9.285936 | 3.032516 | 8.602563 |
| KNN | 1.624301 | 4.258292 | 2.028955 | 5.913842 |
| RF | **0.456958** | **0.591605** | **0.547920** | **1.623569** |
| XGB | 0.528572 | 0.681514 | 0.639628 | 1.887506 |
| LGBM | 0.567967 | 0.739195 | 0.695612 | 2.039303 |
| RBFNN | 2.818290 | 12.852602 | 3.464867 | 10.131078 |
| SVR | 2.414193 | 9.688536 | 3.082528 | 8.646956 |

predictions in MAPE varied between approximately 15% and 21% for Office A and 10% to 27% for Office B.

TABLE V
OFFICE A ILLUMINANCE RESULTS

| Model | MAE | MSE | RMSE | MAPE |
|---|---|---|---|---|
| LR | 35.710120 | 3764.413404 | 58.753953 | 21.725152 |
| KNN | **25.040180** | 2279.148613 | 45.840964 | **15.212611** |
| RF | 25.348496 | 2138.778611 | 45.149898 | 16.593426 |
| XGB | 26.648700 | 2216.546522 | 45.997883 | 17.042875 |
| LGBM | 26.686563 | **1950.823995** | **42.549761** | 17.074290 |
| RBFNN | 31.563508 | 3637.336493 | 58.135234 | 17.710077 |
| SVR | 29.311571 | 3079.040865 | 53.328982 | 17.205404 |

TABLE VI
OFFICE B ILLUMINANCE RESULTS

| Model | MAE | MSE | RMSE | MAPE |
|---|---|---|---|---|
| LR | 51.923726 | 7577.556759 | 84.139777 | 26.894763 |
| KNN | 30.242304 | 3431.465532 | 56.832702 | 14.769444 |
| RF | **19.119812** | **1180.212147** | **33.531889** | **10.376000** |
| XGB | 23.872665 | 1621.414400 | 39.209127 | 12.950563 |
| LGBM | 24.618307 | 1838.003028 | 41.878912 | 13.021734 |
| RBFNN | 42.769991 | 8926.324794 | 90.594729 | 16.573350 |
| SVR | 37.147957 | 6800.841724 | 78.547096 | 15.22319 |

## VI. DISCUSSION

The best results across all evaluations were achieved by using decision-tree based algorithms. The highest results across most measurements and office rooms were achieved by using the Random Forest algorithm with MAPE's below 5%, 3% and 18% for humidity, temperature and illuminance respectively. On the other hand, the worst performing models were RBFNN and Linear Regression, with the latter having the highest errors in case of illuminance, whereas RBFNN had the highest errors in temperature and humidity measurements. Most models have performed better on the temperature and humidity estimations

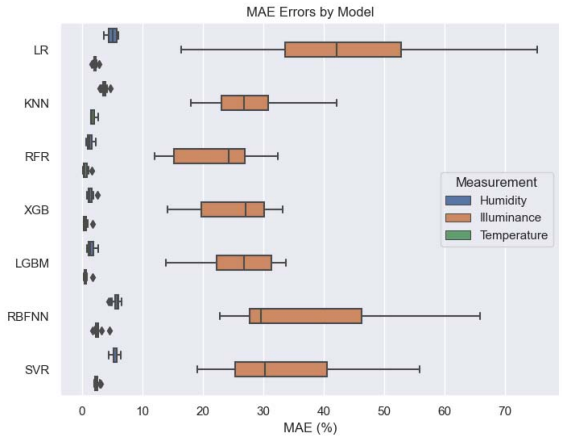


Fig. 2. Mean Absolute Error Box Plot

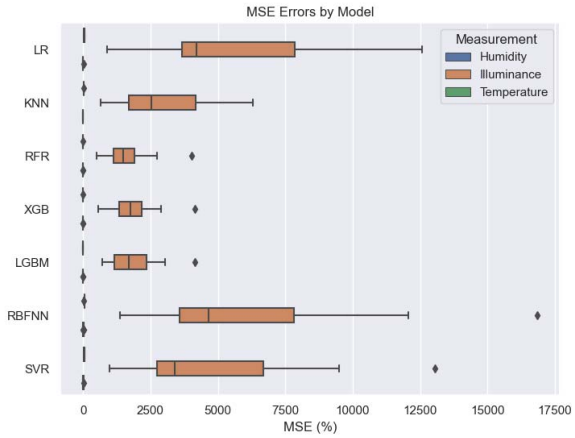


Fig. 3. Mean Square Error Box Plot

when compared to illuminance, apart from RBFNN where it performed better for illuminance.

With regard to illuminance, the results are compared to previous work by Drakoulelis et al. [1]. In their study, neural networks exhibited the lowest performance compared to SVM and linear regression. The results obtained in this paper for tree-based algorithms were better given that M5P algorithm was excluded from the investigation due to low performance. In Kundig et al. [2] the average RMSE and MAPE values reported are comparable to the results achieved in this work, with slightly lower levels of error. However, the lower levels of error may be explained by a shorter interval of analysis of 3 months when compared to the two-year interval we consider, thus creating a larger set of points which add to the errors in predictions. In Syafrudin et al. [23], Random Forest was used in combination with Density-Based Spatial Clustering of Applications to model data related to fault detection demon-

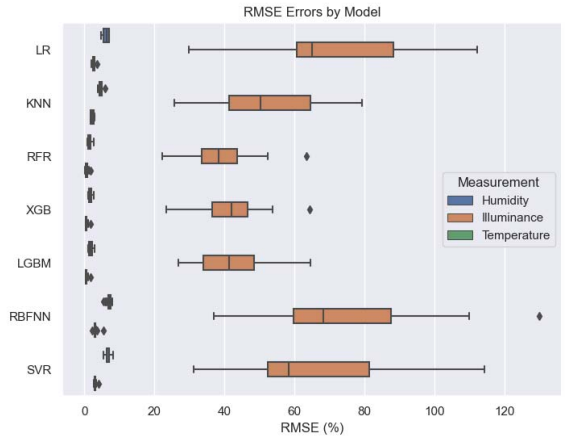


Fig. 4. Root Mean Square Error Box Plot

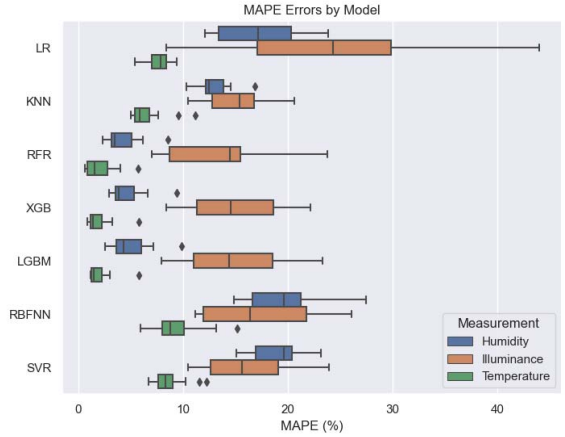


Fig. 5. Mean Absolute Percentage Error Box Plot

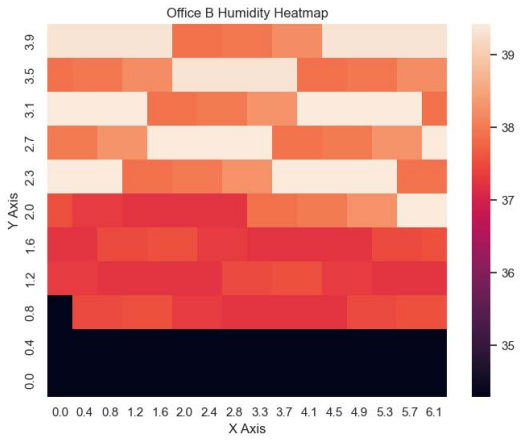


Fig. 6. Office B Humidity Heatmap

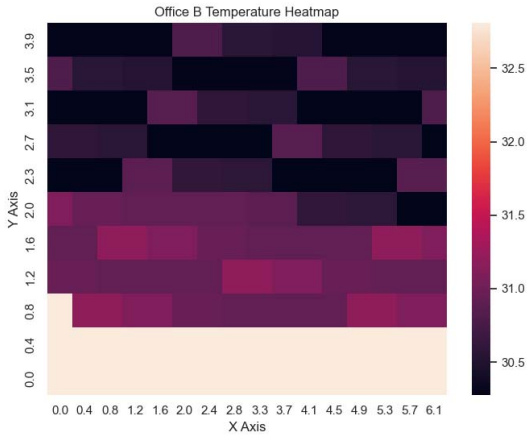


Fig. 7. Office B Temperature Heatmap

strating near 100% accuracy showing the robustness of the algorithm in handling large amounts of data.

The difference in the results among the modalities considered, especially in the case of the illuminance, is that humidity and temperature are less fluctuant and are more likely to have more consistent change patterns over time. Illuminance in the offices is not only controlled by seasonal weather patterns but also the ambient and indoor lighting which adds randomness to the data making it more difficult to predict.

## VII. Conclusion and Future Work

In this work, we investigated the ability of different machine learning methods to predict the output measurements of virtual sensors in indoor environments. This investigation is based on the data supplied by the Syndesi 2.0 testbed of University of Geneva comprising sensory data for temperature, humidity and illuminance. The best results have been generally achieved by the decision-tree based algorithms with Random Forest demonstrating the best modelling performance, while the worse results were obtained when using the RBFNN and linear regression. The best results in accuracy have been achieved for temperature and humidity, with MAPE results below 3% and 5% for temperature and humidity, and below 17% for illuminance.

In our future work will focus on further investigating the parameters' optimisation of the learning methods, in particular for the ensemble tree based models that demonstrate the most promising performance. We will also further investigate techniques of embedded machine learning and Xtreme edge computing in order to develop fully decentralised architectures that greatly reduce their dependency on cloud-based back-end infrastructure.

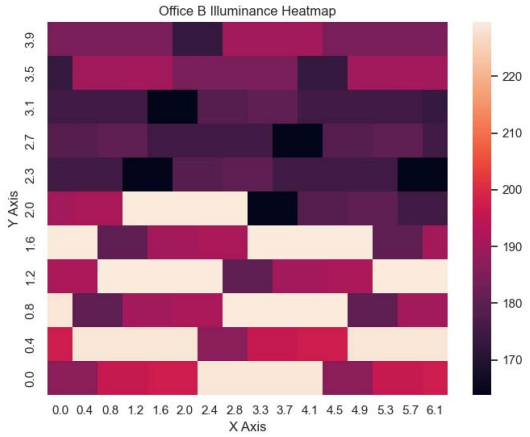


Fig. 8. Office B Illuminance Heatmap

## REFERENCES


[1] M. Drakoulelis, G. Filios, V. G. Ninos, I. Katsidimas, and S. Nikoletseas, "Virtual sensors: an industrial application for illumination attributes based on machine learning techniques," Annals of Telecommunications, vol. 76, no. 7/8, pp. 529-535, 08// 2021.

[2] S. Kundig, C. Angelopuolos, Marios,, and J. Rolim, "Modelled Testbeds: Visualizing and Augmenting Physical Testbeds with Virtual Resources" Advances in Intelligent Systems and Computing, vol. 721, pp. 804-812, 2018.

[3] M. Vukovic, D. Mazzei, S. Chessa, and G. Fantoni, "Digital Twins in Industrial IoT: a survey of the state of the art and of relevant standards," ed: IEEE, 2021, pp. 1-6.

[4] Y. Peng, A. Rysanek, Z. Nagy, and A. Schlüter, "Using machine learning techniques for occupancy-prediction-based cooling control in office buildings," Applied Energy, Article vol. 211, pp. 1343-1358, 02/01/February 2018 2018, doi: 10.1016/j.apenergy.2017.12.002.

[5] L. Haorong and J. E. Braun, "Development, Evaluation, and Demonstration of a Virtual Refrigerant Charge Sensor," HVAC&R Research, Article vol. 15, no. 1, pp. 117-136, 2009, doi: 10.1080/10789669.2009.10390828.

[6] K.-W. Park and A. K. Athienitis, "Workplane illuminance prediction method for daylighting control systems," Solar Energy, Article vol. 75, no. 4, pp. 277-284, 01/01/January 2003 2003, doi: 10.1016/j.solener.2003.08.013.

[7] S. Joanna, S. Tomasz, and B.-W. Robert, "Software-Defined Virtual Testbed for IoT Systems," Wireless Communications and Mobile Computing, article vol. 2018, 01/01/ 2018, doi: 10.1155/2018/1068261.

[8] C. G. Mattera, J. Quevedo, T. Escobet, H. R. Shaker, and M. Jradi, "A Method for Fault Detection and Diagnostics in Ventilation Units Using Virtual Sensors," Sensors (Basel, Switzerland), vol. 18, no. 11, 2018, doi: 10.3390/s18113931.

[9] C. M. Angelopoulos, O. Evangelatos, S. Nikoletseas, T. P. Raptis, J. D. Rolim, and K. Veroutis, "A user-enabled testbed architecture with mobile crowdsensing support for smart, green buildings". In 2015 IEEE International Conference on Communications (ICC) (pp. 573-578), 2015, doi: 10.1007/978-3-319-40509-4_13.

[10] CM Angelopoulos, G Filios, S Nikoletseas, D Patroumpa, TP Raptis and K. Veroutis. "A holistic IPv6 test-bed for smart, green buildings". In 2013 IEEE International Conference on Communications (ICC), 2013, doi:10.1109/ICC.2013.6655569.

[11] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[12] D. Anguita, L. Ghelardoni, A. Ghio, L. Oneto, and S. Ridella, "The 'K' in K-fold Cross Validation," ed. Bruges (Belgium): European Symposium on Artificial Neural Networks, 2012, pp. 441-446.

[13] Scikit-Learn. "Linear Models." Scikit-Learn. https://scikit-learn.org/stable/modules/linear_model.html (accessed 07 April, 2022).

[14] B. Rahman, H. L. Hendric Spits Warnars, B. Subirosa Sabarguna, and W. Budiharto, "Heart Disease Classification Model Using K-Nearest Neighbor Algorithm," ed: IEEE, 2021, pp. 1-4.

[15] L. Breiman, "Random Forests," ed. University of California, Berkeley, CA 94720: Statistics Department, University of California, 2001.

[16] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," ed, 2016.

[17] G. Ke et al., "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," presented at the 31st Conference on Neural Information Processing Systems (NIPS 2017), 2017.

[18] N. Semenova, L. Larger, and D. Brunner, "Understanding and mitigating noise in trained deep neural networks," Neural Networks, vol. 146, pp. 151-160, 2022, doi: 10.1016/j.neunet.2021.11.008.

[19] Zhang, A. R. Abdullah, C. W. Chong, and M. H. Ali, "A Study on Regional GDP Forecasting Analysis Based on Radial Basis Function Neural Network with Genetic Algorithm (RBFNN-GA) for Shandong Economy," Computational Intelligence & Neuroscience, Article pp. 1-12, 2022, doi: 10.1155/2022/8235308.

[20] B. Jonathan, Z. Rahim, A. Barzani, and W. Oktavega, "Evaluation of Mean Absolute Error in Collaborative Filtering for Sparsity Users and Items on Female Daily Network," ed: IEEE, 2019, pp. 41-44.

[21] D. S. K. Karunasingha, "Root mean square error or mean absolute error? Use their ratio as well," Information Sciences, Article vol. 585, pp. 609-629, 03/01/March 2022 2022, doi:

[22] A. de Myttenaere, B. Golden, B. Le Grand, and F. Rossi, "Mean Absolute Percentage Error for regression models," Neurocomputing, Article vol. 192, pp. 38-48, 06/05/June 2016 2016, doi: 10.1016/j.neucom.2015.12.114.

[23] M. Syafrudin, G. Alfian, N. L. Fitriyani, and J. Rhee, "Performance Analysis of IoT-Based Sensor, Big Data Processing, and Machine Learning Model for Real-Time Monitoring System in Automotive Manufacturing," Sensors (14248220), Article vol. 18, no. 9, p. 2946, 2018, doi: 10.3390/s18092946.