

Predictive Analytics for Energy Efficiency Integrating Long Short-Term Memory and Apache Spark for Occupancy Estimation

Lylkend Silva Albuquerque   
*Information Technology department*CCT College Dublin   
Dublin, Ireland  
dmcquaid@cct.ie

David McQuaid  
*Information Technology department*CCT College Dublin   
Dublin, Ireland  
dmcquaid@cct.ie

Dr. Muhammad Iqbal  
Information Technology Department  
CCT College Dublin  
Dublin, Ireland  
miqbal@cct.ie

*Abstract*

This work is designed to build a time-based occupancy estimating model in smart buildings using LSTM (Long Short Term Memory) networks and Apache Spark with environmental sensor data. Correct estimation of occupancies allows for save energy use as HVACs systems, lightings and other unit work together to feed a less energy consumption without affecting comfort. Splitting LSTM between its potential to favor time patterns and Spark's experience with big data processing, we use temperature, CO2, light, and sound data to predict room occupancy. At present, these initial results highlight the accuracy of the method, thereby enabling an energy-efficient and scalable solution for shaping the future of energy management. The objective of this research is to show how coupled deep learning and big data technologies can be used to upgrade building energy efficiency. Developed building operations were utilized as a case study.

Keywords

— Time Series Forecasting, LSTM (Long Short-Term Memory), Room Occupancy Estimation, Deep Learning, Apache spark.

# Literature review

In the era of smart buildings and the Internet of Things (IoT), accurate room occupancy estimation stands as a cornerstone for energy efficiency, security, and automated environmental control. The integration of machine learning algorithms with non-intrusive sensor data paves the way for innovative solutions that promise not only to optimize energy consumption but also to enhance the occupants' comfort and safety. This project draws inspiration from a breadth of research, including the pioneering works on anomaly detection in IoT-based healthcare[1], the application of machine learning algorithms in stock price prediction[2], highlighting the versatility and potency of machine learning across diverse domains.

Occupancy estimation, in particular, leverages the predictive capabilities of Long Short-Term Memory (LSTM) neural networks, renowned for their efficiency in handling time-series data and their ability to remember information for long periods, a feature crucial for interpreting the temporal patterns in occupancy data. The LSTM model's applicability to dynamic data sets has been validated in various fields, from healthcare[3] to stock market trends, and now, room occupancy estimation[4]. This project aims to further the application of LSTMs in real-world scenarios, underscoring their adaptability and the depth of insights they can uncover from seemingly mundane data.

The challenge of accurately estimating room occupancy extends beyond the mere prediction of numerical values. It encompasses the understanding of complex patterns and relationships within the data, which are often hidden and not immediately apparent. The researches on predicting business process activities[5] and measuring systemic risk[6], respectively, underscores the significance of capturing and analysing dynamic and non-linear relationships within data, a challenge that LSTMs are particularly well-suited to address.

Moreover, the project acknowledges the critical role of data pre-processing and feature engineering in machine learning. Insights from electrostatic generator[7] enhancements and the work on jewellery rock discrimination through laser-induced breakdown spectroscopy and convolutional LSTM networks[8] emphasize the importance of meticulous data preparation. They highlight how innovative pre-processing techniques can significantly augment the performance of machine learning models, a principle that is diligently applied in this project to enhance the accuracy of occupancy predictions.

Leveraging Apache Spark for data processing, this project stands on the shoulders of giants, building upon the foundational research into Spark's performance in map/reduce operations[9]. The scalability and efficiency of Spark are instrumental in handling the voluminous data generated by IoT sensors, ensuring that the data fed into the LSTM models is of the highest quality and structured in a manner that maximizes the potential for accurate predictions.

In conclusion, this project synthesizes insights from diverse fields and pioneering research to advance the application of LSTM neural networks in room occupancy estimation. Through rigorous data pre-processing, leveraging the computational power of Apache Spark, and employing advanced machine learning algorithms, it aims to set a new benchmark in the field of smart building management. This endeavour not only contributes to the academic discourse, as reflected in the comprehensive bibliography [1]-[15] but also holds profound implications for the future of energy efficiency, occupant comfort, and IoT applications in smart environments

# Methodology

## Data gathering

The search for suitable data for this project underscored the difficulties inherent in locating open-source, high-quality datasets, a cornerstone for any rigorous machine learning endeavour. The scarcity of such resources, which are critical for ensuring the validity and reliability of project outcomes, presents a formidable challenge. Against this backdrop, the "Room Occupancy Estimation"[10] dataset, sourced from the esteemed UC Irvine Machine Learning Repository, stood out as a valuable find. This repository's reputation for hosting diverse and credible datasets made it an ideal choice, offering a blend of accessibility and academic rigor necessary for the project's success. The utilization of this dataset not only underpins the project's analytical foundation but also highlights the importance of dependable data sources in advancing machine learning research.

## Data pre-processing and Feature Engineering

In the undertaken project, the dataset underwent a rigorous pre-processing phase utilizing PySpark, initiated by conducting Exploratory Data Analysis (EDA) to discern underlying patterns and data characteristics. This phase also encompassed the exclusion of superfluous features not pertinent to the LSTM's computational analysis, a strategic aspect of feature engineering. Subsequently, the dataset was transformed into a Pandas DataFrame, facilitating advanced pre-processing techniques. The culmination of this pre-processing phase was the normalization of the dataset, a critical step to standardize the range of data values, ensuring equitable contribution to the LSTM model's predictive capabilities.

*Table I*

*Statistical summary about feature and targuet variables*

|  |  |  |
| --- | --- | --- |
| **Summary** | **Feature** | **Target** |
| Count | 10129 | 10129 |
| Mean | 25.44505 | 0.39855 |
| Stddev | 51.01126 | 0.89363 |
| Min | 0 | 0 |
| Max | 165 | 3 |

The dataset under consideration was systematically recorded at 30-second intervals to meticulously document emerging trends and patterns over a defined span of 30 minutes. This temporal resolution is encapsulated by the formula:

​

denoting that the dataset comprises 60 distinct time steps, each representing a 30-second segment within the half-hour window.

Autocorrelation is high at a specific lag, this suggests that the value at that lag is predictive of the current value and should be included as an input to the LSTM. The autocorrelation value ranges from +1 to -1. A value close to +1 implies that a high/low value of the variable is followed by a high/low value at the subsequent time point.

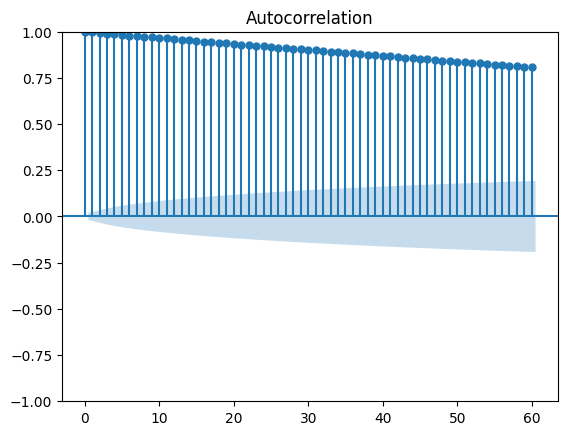


Fig. 1 Autocorrelation of the target variable with the number of steps already calculated.

In the provided plot, we can observe a strong positive autocorrelation at initial lags that seems to taper off but remains positive, suggesting a persistent pattern over time within the data.

## LSTM the Machine Learning Algorithm

The examination is centred on an elaborate design of LSTM neural network used for the purpose of room occupancy state classification, where a multi-layered approach using 80 neurons and a softmax activation function has been taken to provide for probabilistic outputs. The Adam optimizer was employed to optimize model compilation against categorical crossentropy loss function with accuracy as the evaluation metric. During 25 epochs, the model was extensively trained in batches of 20 and validation split to ensure generalization. For example, the model showed an impressive test accuracy of 91.49%, which also indicates that the algorithm is robust and accurate when it comes to interpreting resampled training data turned into Synthetic Minority Over-sampling Technique (SMOTE) due to issues regarding imbalanced class representation. It should be noted that this high degree of accuracy, reported as 0.9524, together with its loss value of 0.2040 clearly shows how effective this model is and how well it utilizes LSTM networks for predictive analytics in smart environments.

# Experiments and results

The project used TensorFlow[11] with Keras for implementing LSTM. To store and process data, it used PySpark with Hadoop. This is one of the biggest applications of a machine learning framework (frameworks, to be exact) and big data platforms to the estimation of room occupancy. Various metrics, such as accuracy, loss, F1 score, precision, recall and support were calculated using Scikit-learn library to provide a thorough evaluation of the model.

The integrity of the time-series throughout the pre-processing and evaluation was ensured by proper data segmentation of the vector consisting of behavioural data – with the split point being at the time step of the estimation. The sequential nature of observations was retained to preserve temporal patterns that could impact the predictive accuracy of the LSTM model. Model optimization was guided by key parameters and hyperparameters, including the use of 'relu' and 'softmax' activation functions, the Adam optimizer, and categorical crossentropy as the loss function. These choices were pivotal in tuning the LSTM model for optimal performance. Feature engineering and data normalisation dramatically improved the data, such as selecting only one feature per variable, using MinMax scaler for normalisation, and using one-hot encoding for the target variable. This data was optimised so that the model would select the occupancy levels accurately. Addressing data imbalance, a main reason for the low accuracy and underfitting in the earlier attempt, by using SMOTE to better balance the classes. The improved model learning caused by boosting the occupancy states aligned well with the proposed hypothesis, but also reinforce the need to expand the dataset to classify all the states.

Its holistic design accentuates the novelty of the project’s application of machine learning and big data solutions to smart building operation and management, testing what is possible on the ground when it comes to optimising occupancy estimations and energy efficiency.

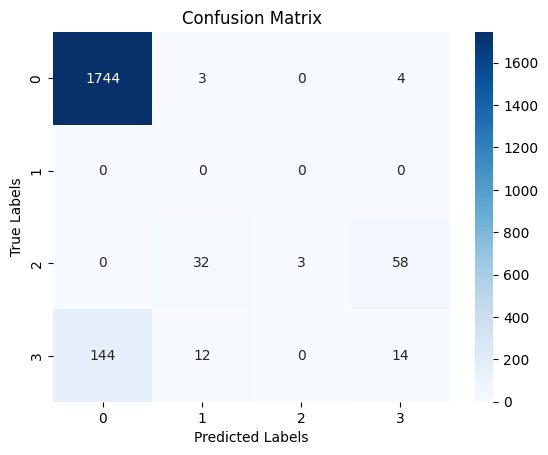


Fig. 2 Confusion matrix with the LSTM model predictions

In a confusion matrix, a table is used to show how well a classification model performed. In the provided matrix: The model accurately predicted class '0' (no occupancy) 1744 times, with a few errors where it predicted '1' for 3 instances and '3' for 4 instances. There were no predictions for class '1', which there were no instances of this class in the test set or the model struggled to predict it. For class '2' (medium occupancy), the model got it right 32 times, but mistakenly thought it was class '3' (high occupancy) 58 times and class '1' 3 times. Class '3' was correctly predicted 14 times, but 144 instances were wrongly classified as class '0' and 12 as class '2'. The diagonal shows the correct predictions.

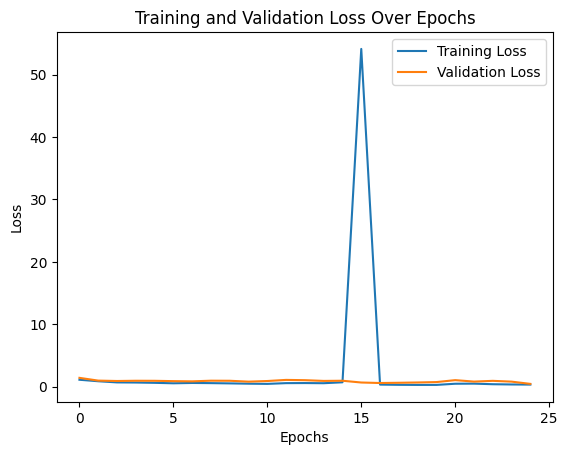


Fig. 3 Plot of the loss discrepancy of the LSTM model

The chart given above shows the loss of both the training and validation data as the epochs proceed. The x-axis represents the number of epochs, while the y-axis denotes the loss. The training loss is consistently decreasing suggesting that more data is making it better. However, there is a sudden rise in validation loss at about epoch 15 which means that there is some error observable in model’s performance. By highlighting on this anomaly it suggests that we may have a problem with our data. In general, except for the outlier spike mentioned earlier on, the model works well.

*Table II*

*Classification report metrics*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Precision** | **Recall** | **F1-Score** | **Support** |
| 0 | 0.92 | 1.00 | 0.96 | 1751 |
| 1 | 0.00 | 0.00 | 0.00 | 0 |
| 2 | 1.00 | 0.03 | 0.66 | 93 |
| 3 | 0.18 | 0.08 | 0.11 | 170 |
|  |  |  |  |  |
| Accuracy |  |  | 0.87 | 2014 |
| Macro avg | 0.53 | 0.28 | 0.28 | 2014 |
| Weighted avg | 0.86 | 0.87 | 0.85 | 2014 |

This table is a classification report that provides several important metrics evaluating the performance of a classification model

Precision: Indicates how many of the items identified as belonging to a certain class were actually of that class. For example, class 0 has a precision of 0.92, meaning that 92% of the items labelled as class 0 by the model were truly class 0.

Recall: Shows how many of the items from a certain class were identified correctly. Class 0 had all of its items (100%) correctly identified, giving it a recall of 1.00.

F1-score: Combines precision and recall into a single metric by taking their harmonic mean. Class 0's high precision and recall give it a strong F1-score of 0.96, but class 2, despite perfect precision, has a low F1-score of 0.06 due to very low recall.

Support: The number of true instances for each class in the data. For instance, there were 1751 instances of class 0

The "macro avg" gives the average precision, recall, and F1-score without considering the support, while the "weighted avg" gives these averages weighted by the support. This model has high accuracy overall (87%), but the low recall for classes 2 and 3, as well as a lack of support for class 1, Just show as the model needs more data to perform well on less with the others classes.

# Conclusion

In this study, a strategy for multi-sensor node deployment to estimate room occupancy counts was explored using LSTM networks and Apache Spark for data processing. Even though the initial models faced problems with class imbalance and underfitting, SMOTE application and careful feature engineering such as one-hot encoding and MinMax scaling helped in improving model performance. Despite achieving commendable accuracy, the LSTM model showed that there is need of more data diversification. To be comprehensive in real-time analysis, future work also includes adding temporal features and extending use of the model to larger spaces.

##### References

[1] M. M. Khan and M. Alkhathami, “Anomaly detection in IoT-based healthcare: machine learning for enhanced security,” *Sci Rep*, vol. 14, no. 1, p. 5872, Mar. 2024, doi: 10.1038/s41598-024-56126-x.

[2] T. Phuoc, P. T. K. Anh, P. H. Tam, and C. V. Nguyen, “Applying machine learning algorithms to predict the stock price trend in the stock market – The case of Vietnam,” *Humanit Soc Sci Commun*, vol. 11, no. 1, pp. 1–18, Mar. 2024, doi: 10.1057/s41599-024-02807-x.

[3] M. Ghislieri, C. Giacinto Luigi, M. Knaflitz, and V. Agostini, “Long short-term memory (LSTM) recurrent neural network for muscle activity detection.” Zenodo, Oct. 13, 2021. doi: 10.5281/zenodo.5617950.

[4] D. R, K. M. Raj, N. Balaji, and D. K, “Machine Learning based Estimation of Room Occupancy Using Non-Intrusive Sensors,” in *2022 International Conference on Communication, Computing and Internet of Things (IC3IoT)*, Mar. 2022, pp. 1–5. doi: 10.1109/IC3IOT53935.2022.9767992.

[5] J. Roa, M. Rubiolo, and E. Tello-Leal, “Predicting Activities in Business Processes with LSTM Recurrent Neural Networks,” vol. 1, Jun. 2018, doi: 10.17632/trskzyg3j9.1.

[6] I. L. Aprea, S. Scognamiglio, and P. Zanetti, “Systemic risk measurement: A Quantile Long Short-Term Memory network approach,” *Applied Soft Computing*, vol. 152, p. 111224, Feb. 2024, doi: 10.1016/j.asoc.2023.111224.

[7] Z. Wu, Z. Cao, J. Teng, R. Ding, J. Xu, and X. Ye, “Electrostatic generator enhancements for powering IoT nodes via efficient energy management,” *Microsyst Nanoeng*, vol. 10, no. 1, pp. 1–10, Mar. 2024, doi: 10.1038/s41378-024-00660-1.

[8] P. Khalilian *et al.*, “Jewelry rock discrimination as interpretable data using laser-induced breakdown spectroscopy and a convolutional LSTM deep learning algorithm,” *Sci Rep*, vol. 14, no. 1, p. 5169, Mar. 2024, doi: 10.1038/s41598-024-55502-x.

[9] “Simple Map/Reduce Performance for Apache Spark/Scala on Cooley.” Accessed: Mar. 22, 2024. [Online]. Available: https://figshare.com/articles/dataset/Simple\_Map\_Reduce\_Performance\_for\_Apache\_Spark\_Scala\_on\_Cooley/3753408

[10] A. P. Singh, V. Jain, S. Chaudhari, F. A. Kraemer, S. Werner, and V. Garg, “Machine Learning-Based Occupancy Estimation Using Multivariate Sensor Nodes,” *2018 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, Dec. 2018, doi: 10.1109/GLOCOMW.2018.8644432.

[11] “tf.keras.layers.LSTM | TensorFlow v2.16.1,” TensorFlow. Accessed: Apr. 07, 2024. [Online]. Available: https://www.tensorflow.org/api\_docs/python/tf/keras/layers/LSTM