A User-in-the-loop Digital Twin for Energy Consumption Prediction in Smart Homes

Davide Guizzardi[1], Barbara Rita Barricelli[1] and Daniela Fogli[1,*]

[1]Department of Information Engineering, University of Brescia, Brescia, Italy

Abstract

This paper describes a digital twin of smart homes able to predict future energy consumption and help the user make better decisions about the activation of smart appliances and the scheduling of automations comprising different appliances' activations. To deal with the problem of time series forecasting for energy consumption prediction, a deep learning approach based on Long-Short Term Memory has been adopted, and a grid search has been used to identify the values of hyperparameters with the best prediction accuracy. Proper information visualization and interaction features have been then implemented in the digital twin interface to explain to the user the predicted consumption data and the reasons underlying warnings and suggestions provided by the system. In this way, the digital twin becomes a system based on artificial intelligence that exhibits an explainable behavior, which allows the user to make decisions about smart home management in a more conscious and sustainable way.

Keywords

Digital twin, Smart home, Consumption prediction, Information visualization

1. Introduction

A Digital Twin (DT) is usually conceived as the digital counterpart of a physical system, which can exploit a real-time synchronization of the data coming from the field and mirror the life of the physical system, also thanks to machine learning algorithms, for the sake of simulation, forecasting, or remote control [1, 2, 3]. Evans et al. [4] provided a classification of maturity levels for DTs, where the highest levels (4 and 5) are the most interesting from the user's perspective: level 4 foresees the control of the physical system from the DT through a proper user interface, while level 5 regards DTs with complete self-governance but featuring total oversight and transparency for the benefit of their users. Therefore, in these levels, user-in-the-loop is fundamental for the effective and secure operation and control of the DT. Assuming this perspective, in [5], we reviewed the literature discussing the interaction with DTs and underlined the lack of research that considers these advanced levels of DTs and DT's actual use by end users while at work or generally in their everyday lives. With our project, we intend to contribute to resolving this issue by designing and developing a DT for a smart home. A smart home is a specific application of the Internet of Things that encompasses interconnected smart sensors and appliances and whose behavior can be directly, remotely, or automatically controlled by home inhabitants through custom software applications or virtual assistants like Google Home or Amazon Alexa.

Specifically, we have developed a DT of smart homes that collects data about energy consumption and supports users in decision-making about the activation of appliances to avoid excessive consumption or even overcome the maximum load capacity available for the home. This is achieved in our system through i) a machine learning model capable of predicting the future energy consumption of the smart home based on historical data, and ii) an easy-to-use DT interface that provides users with an explainable output of the machine learning model, leveraging suitable information visualization. In particular, the activation of smart appliances, through the interaction with the DT, triggers real-time adaptation of the energy consumption prediction, thus making machine learning interactive and explanation generation dynamic.

Joint Proceedings of the ACM IUI Workshops 2025, March 24-27, 2025, Cagliari, Italy

*Corresponding author.

davide.guizzardi@unibs.it (D. Guizzardi); barbara.barricelli@unibs.it (B. R. Barricelli); daniela.fogli@unibs.it (D. Fogli)

0009-0000-7761-7103 (D. Guizzardi); 0000-0001-9575-5542 (B. R. Barricelli); 0000-0003-1479-2240 (D. Fogli)

Therefore, as a first contribution of this paper, a Long Short-Term Memory (LSTM)-based deep learning model [6] for the prediction of future energy consumption is presented. Its implementation required preliminary activities related to data extraction from the testing environment, data preparation, and dataset creation for training.

The second contribution of the paper is an interaction scenario to show how the DT can support the user in deciding about the activation of an appliance or the scheduling of a new automation comprising a sequence of actions on smart home appliances triggered by a time-based event. To this purpose, we have designed effective visualization of the energy consumption prediction and proper interaction features of the DT that allow the user to evaluate the consequences of their decisions before these are actually made.

2. Extracting data from a real smart home

To gather data on the devices' real-time energy consumption, Home Assistant was used. Home Assistant (HA) is an open-source platform designed to facilitate the management of smart devices in a connected home environment. It enables seamless integration with a wide range of devices and provides a centralized interface for monitoring their states and controlling their behavior. Using the API that HA offers, the Digital Twin was able to retrieve data on the devices' history and convert each recorded device state to its corresponding energy consumption value. The converted history was then saved inside the DT database and then used for multiple purposes such as providing users with consumption graphs and training the LSTM models.

A prototype of the system was implemented in the home of one of this work's authors and was used to retrieve consumption data from the 1st of September 2024. As this was an initial pilot test of the Digital Twin's features, the deployment included a restricted number of devices. In particular, the test home included:

- One Shelly smart plug, tasked to monitor and control a television;
- One Shelly smart plug, connected to a desktop computer;
- Two smart lights, one with RGB capabilities and a white one;
- One Google Home Mini smart speaker, used as the primary audio interface for the desktop computer;
- One Google Chromecast, connected to a secondary television, used mainly to access streaming platforms;
- One Shelly humidity and temperature sensor;
- One Raspberry Pi 5, tasked to run the Digital Twin APIs, its web interface and the Home Assistant Operating System.

3. Applying deep learning for energy consumption forecast

3.1. Data preparation

Although it is possible to feed a deep learning model with the sole consumption data, several works, such as [7] and [8], demonstrated how adding contextual information (e.g., temporal and weather data) could improve the prediction accuracy. This is because household energy consumption is deeply intertwined with human routines and environmental factors. For example, energy usage tends to fluctuate based on the time of day, with peaks often occurring in the morning and evening when inhabitants are most active.

The data preparation approach used in this work resembles the one presented in [8]. In particular, each entry of the data that came from the Digital Twin was enhanced with the following information:

- An integer value $H \in [0, 6]$, indicating the day of the week the record refers to, where 0 is equal to Monday and 6 to Sunday;

- An integer value $S \in [0, 3]$, indicating the "time session" of the row. Four possible time sessions exist: "Night" that goes from 00:01 to 06:00 in the morning, "Morning" that goes from 06:01 to 12:00, "Afternoon" from 12:01 to 18:00, and "Evening" from 18:01 to 23:59.

Additionally, the energy consumption values were scaled using various scaling models provided by the scikit-learning library, whose effects will be discussed in the next sections. Scaling features is a crucial step in most machine learning approaches as they provide several advantages, such as ensuring feature uniformity, accelerating algorithm convergence, and helping avoid some known issues, like the "vanishing gradient" [9].

The last step in the dataset preparation procedure is the creation of the input and output sequences. The training of an LSTM model for the prediction of real values from time series requires feeding the algorithm with data that helps it understand sequential dependencies and learn temporal patterns effectively. The network usually takes as input an arbitrary number of past observations and produces one (single-step forecast) or multiple future values (multi-step forecast). A sliding window approach is often used to build these sequences. This involves defining a fixed-length window that moves across the time series data. Each position of the window extracts a subsequence of past data points to form the input and the corresponding future points to serve as the output.

For example, given the number of past $n = 3$ observations and the number of predicted values ($m = 1$), the i-th row in the dataset would be $[(x_{i-3}, x_{i-2}, x_{i-1}), y_i]$. A summary of the steps that compose the data preparation phase is presented in Figure 1.

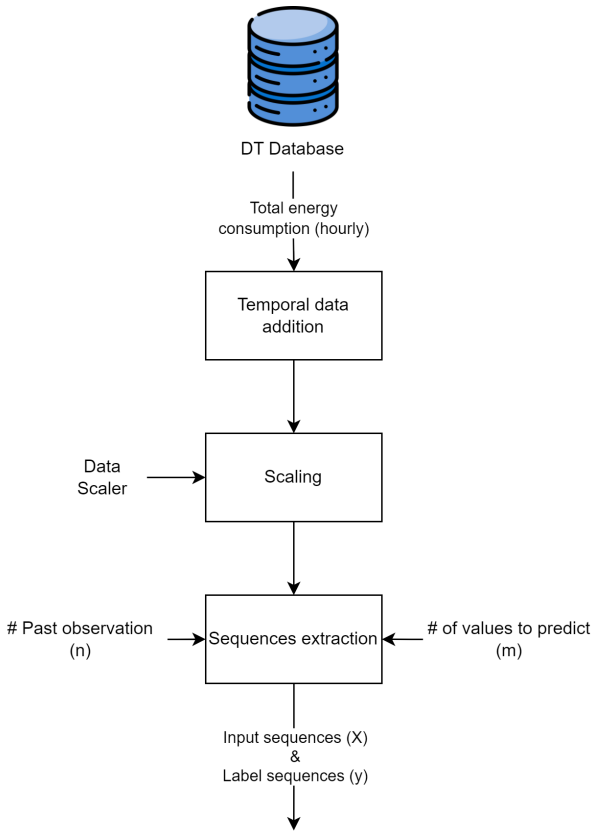

Figure 1: Diagram of the steps that compose the data preparation phase

3.2. Architectures of the LSTM models used

Presented for the first time in [6], a Long Short-Term Memory is a type of recurrent neural network (RNN) designed to model sequential data by capturing long-term dependencies. Its particular structure, composed of an internal state whose value is controlled by three functions ("Forget gate", "Input gate",

and “Output gate”), allows it to selectively remember or forget information, making it the preferred candidate for tasks like time series forecasting, natural language processing, and speech recognition. Several studies in which LSTMs are employed for energy consumption prediction are available. In [8], LSTMs are combined with a data preparation phase in which additional contextual data are added to the training set and used for the prediction. The paper [10] compares LSTMs with other state-of-the-art models, such as the backpropagation neural network (BPNN). In [11], an LSTM-based approach is used to predict hourly load from a publicly available dataset containing data from New England’s households. In this work two different LSTM architectures have been tested:

- *Single LSTM*: in this architecture, one single LSTM layer is implemented. The input sequences obtained after the data preparation explained in the previous sub-section are fed to a normalization layer first and then become the input of an LSTM. The classification head is composed of a single fully connected layer whose size is equal to the number of predicted steps (m);
- *Stacked LSTM*: in the stacked LSTM architecture, two LSTM layers are arranged sequentially. As for the previous architecture, a normalization layer is put before the first LSTM while the fully connected layer (with dropout) is connected to the output of the second LSTM. Stacking LSTM could improve the capability of the model to learn hierarchical relations or more complex temporal patterns in the data.

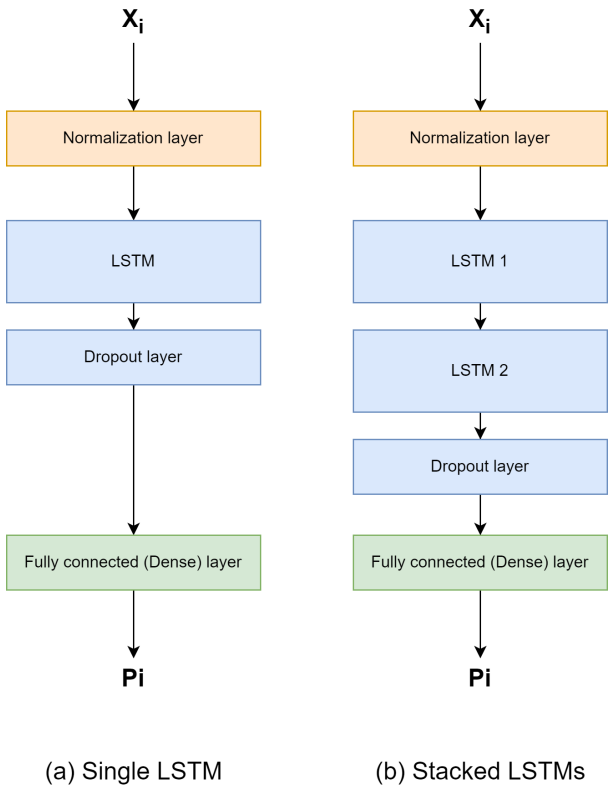

Figure 2: Neural network architectures tested for energy consumption prediction. (a) Single Layer LSTM. (b) Stacked LSTMs. X_i represents the i-th sequence data while P_i is the corresponding predicted value obtained from the model.

3.3. Finding best-performing models with grid search

To identify the values of hyperparameters that could have the best prediction accuracy over the test set, a grid search scheme was implemented. This method systematically explores a predefined range of hyperparameter values, evaluating each combination using a performance metric. In our case, the hyperparameters considered, with their respective admitted values, are presented in Table 1.

Table 1
Hyperparameters values tested in the grid search scheme.

Tested hyperparameter	Possible values
Data scaler	No scaler, MinMaxScaler, RobustScaler
Dense layer activation function	Linear, Sigmoid, ReLU
Number of past observations used (n)	12, 24, 48
Number of future steps to predict (m)	1, 6, 12
Architecture used	Single LSTM, Stacked LSTM

Mean Absolute Error (MAE) was chosen as the evaluation metric over Mean Absolute Percentage Error (MAPE) as MAPE can produce misleadingly large errors with small, scaled data, such as those transformed by MinMaxScaler. MAE, being unaffected by the data scale, provides a more stable and interpretable measure of prediction accuracy. For other hyperparameters, such as the number of epochs and training set size, a single configuration was applied across all grid search iterations (detailed in Table 2). To mitigate overfitting caused by excessive epochs, an early stopping method was used. This regularization technique halts training when validation performance ceases to improve, optimizing the number of training epochs without manual intervention.

Table 2
Hyperparameters configuration for each iteration of grid search.

Hyperparameter	Chosen value
Training set size	70%
Validation set size	10%
Test set size	20%
Optimizer	Adam
Loss function	MAE
Epoch	500
Batch size	32
Additional callbacks	EarlyStopping with patience 20

The method for generating predictions varies depending on the number of forecasted steps, m. For a single-step prediction ($m = 1$), recursive sequence prediction is used, where the model's predictions for earlier time steps are fed back as inputs to predict future steps. Figure 3 shows a graphic example of a three-step prediction from a model that uses $n = 3$ past observations and outputs $m = 1$ future steps.

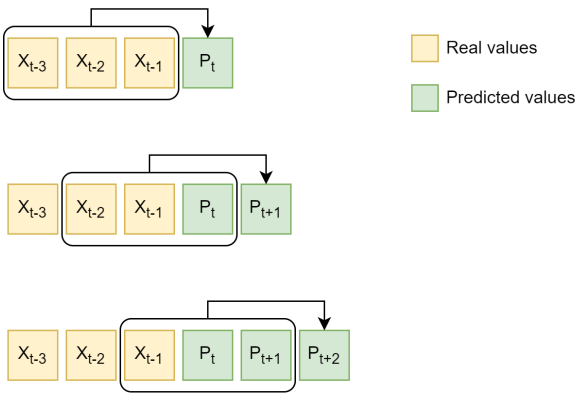

Figure 3: Graphical example of prediction of multiple future steps using recursion.

For multi-step predictions ($m > 1$), the final forecast is a weighted average of the m individual predictions. The rationale stems from the observation that when m is greater than 1, the subsequent prediction will be partially overlapping. For example, given a model that predicts 3 steps ahead, at time t it will produce $[P_{t+1}, P_{t+2}, P_{t+3}]$. However, the values P_{t+1} and P_{t+2} would also have been included in

the sequence predicted at time $t-1$, namely $[P_t, P_{t+1}, P_{t+2}]$. By combining these overlapping predictions, a more robust and smoothed final estimate for P_{t+1} and P_{t+2} can be obtained. The predictions are combined using a weighted average, where greater importance is assigned to the more recent predictions. This approach recognizes that predictions closer to the current time are typically more reliable, as they are based on more up-to-date real information.

A total of 162 hyperparameter combinations have been tested during the grid search execution. Table 3 presents the best-performing models depending on the value of the prediction steps variable (m). It is evident how simpler models seem to suit better the gathered data, implying the inhabitants of the test home could have followed recurrent routines when using the appliances available. Noticing that, apart from the last case, no scaler was needed, it is possible to assume that the scale of the input data wasn't large enough to disrupt the LSTM's learning process. A similar assumption could be made with respect to the activation function, as the best-performing function was the Linear activation, which doesn't apply any transformation to the input it receives. It must be said that such promising results could also be heavily affected by having a restricted pool of monitored devices. In particular, the presence of more demanding and complex devices, such as a washing machine for example, could completely change the results for the worse as their "non-linear" behaviors could pose a bigger challenge for the models presented in this work.

Table 3
Best performing grid search models.

Data scaler	Activation	Past obs.	Prediction steps	Architecture	MAE on Test [Wh]
No Scaler	Linear	24	1	Single LSTM	22.697
No Scaler	Linear	12	6	Single LSTM	30.404
MinMaxScaler	Linear	12	12	Single LSTM	30.248

4. Interaction scenario

The resident of a smart home decides to activate the air conditioner but opts to use the digital twin system instead of the remote control or the air conditioner proprietary app. Within the apartment's map on the application, the resident selects the air conditioner they wish to turn on. The system then allows them to simulate the activation to inform them about the potential impact on energy consumption and costs. Figure 4, shows the system providing information about the estimated duration and cost of operating the air conditioner. Additionally, an alert message appears, warning that turning on the air conditioner immediately could cause an interruption in the electricity supply (Figure 4 - left). The user can access an explanation of this potential issue: the interruption would be caused by an overload in energy consumption that would exceed the 3kW limit of the home's electrical meter[1] (Figure 4 - right).

The system also suggests possible solutions to avoid this issue, listing the currently active devices that could be turned off (Figure 5 - left), and the automations scheduled to start during the estimated duration time (Figure 5 - right) that might be disabled to reduce the electrical load.

If the simulation indicates no risk of overload, the digital twin system allows the user to proceed with activating the air conditioner.

5. Discussion and conclusion

In this paper, we have presented a DT of smart homes implementing a deep learning approach to predicting future energy consumption based on the home's historical data. The user can thus interact with the DT interface to directly activate appliances or create automations comprising sequences of appliances' activations/deactivations. Thanks to machine learning, the DT allows the user to evaluate

[1] 3 kW is the usual maximum meter capacity for energy domestic use in Italian households. The Italian case can be considered representative of those countries with energy availability issues.

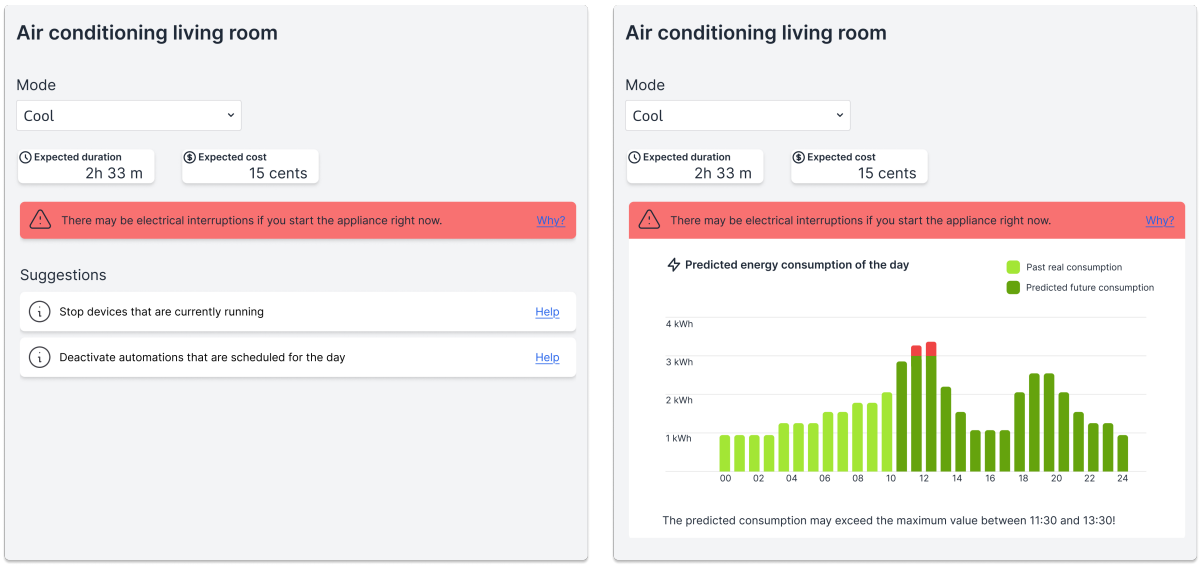

Figure 4: An alert message appears informing that turning on the air conditioner could cause electrical interruptions (left). The user accesses an explanation of the potential issue and obtains a chart of the predicted consumptions with red areas indicating the overcoming of the electrical meter (right).

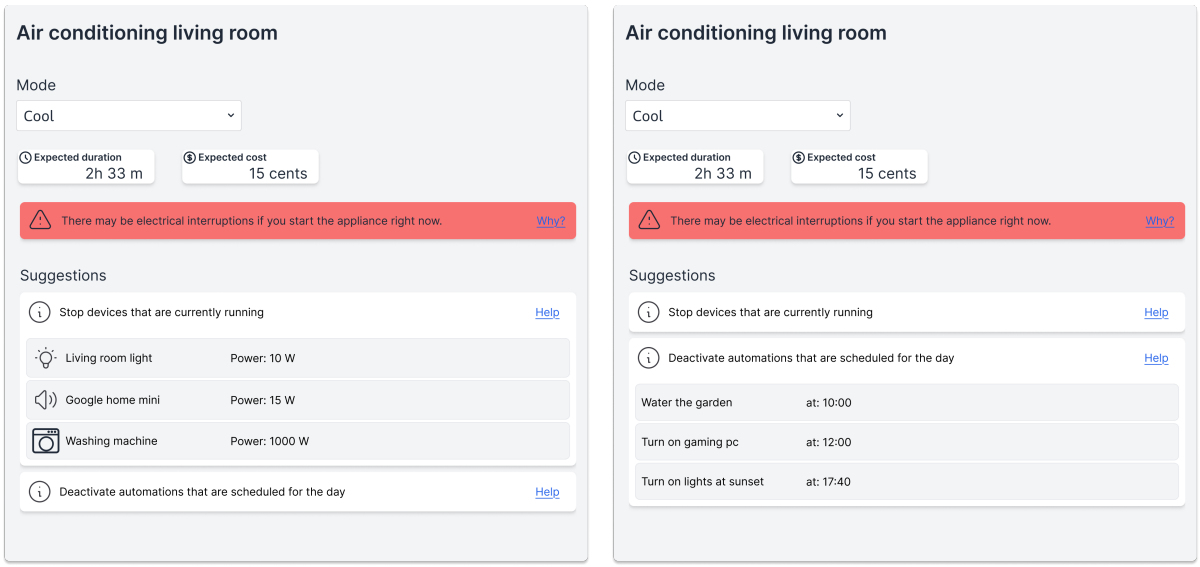

Figure 5: Suggestions about the devices that are currently running, which can be turned off (left). Suggestions about the automations scheduled for the day, which can be deactivated (right).

whether energy issues may occur in the future based on predicted consumption and helps the user solve these potential issues with explanations about the motivation underlying the problem and suggestions for useful actions. We chose to exploit information visualization to explain the data computed by the deep learning model following the visual approaches discussed in [12]. In particular, we decided to use colored bar charts to show the model output, namely the consumption estimation and potential consumption overload due to users' actions or scheduled automations. In addition, interaction with the DT interface implements the "what-why-how" framework proposed in [13], thus enhancing explainability for the benefit of users' decision-making.

The proposed approach has, however, some limitations and critical aspects. First, consumption prediction is affected by uncertainty: a prediction that is close in time will have low uncertainty, but the further away in time the prediction is, the worse it will be. Second, DT suggestions could contrast with users' contingent needs (e.g., the user would like to refresh the house now, not at a time when the

energy is less expensive) or users' values (e.g, the user could assign higher importance to well being than to environmental sustainability or energy expenditure). Third, the approach assumes that all appliance activations are performed through the DT interface to be scrutinized before actual execution, but this could overwhelm the user with annoying interactions and ultimately discourage them from using the system. In conclusion, further features should be designed and implemented that take into account users' habits and values in system suggestions, and increase user engagement in the adoption of such kinds of systems. Future work will be devoted to testing the DT in real households and to the study of mechanisms to improve system acceptance and appropriation.

Acknowledgments

This work has been supported by the Italian MUR PRIN 2022 PNRR Project P2022YR9B7, End-User Development of Automations for Explainable Green Smart Homes, funded by European Union - NextGenerationEU.

References

[1] E. Negri, L. Fumagalli, M. Macchi, A review of the roles of digital twin in cps-based production systems, Procedia Manufacturing 11 (2017) 939–948. doi:https://doi.org/10.1016/j.promfg.2017.07.198, 27th International Conference on Flexible Automation and Intelligent Manufacturing, FAIM2017, 27-30 June 2017, Modena, Italy.

[2] S. Yun, J.-H. Park, W.-T. Kim, Data-centric middleware based digital twin platform for dependable cyber-physical systems, 2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN) (2017) 922–926.

[3] B. R. Barricelli, E. Casiraghi, D. Fogli, A Survey on Digital Twin: Definitions, Characteristics, Applications, and Design Implications, IEEE Access 7 (2019) 167653–167671. doi:10.1109/ACCESS.2019.2953499.

[4] S. Evans, C. Savian, A. Burns, C. Cooper, Digital Twins for the Built Environment: An Introduction to the Opportunities, Benefits, Challenges and Risks, Technical Report, The Institution of Engineering and Technology (IET), 2019.

[5] B. R. Barricelli, D. Fogli, Digital twins in human-computer interaction: A systematic review, International Journal of Human–Computer Interaction 40 (2024) 79–97. doi:10.1080/10447318.2022.2118189.

[6] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (1997) 1735–1780. doi:10.1162/neco.1997.9.8.1735.

[7] I. Kiprijanovska, S. Stankoski, I. Ilievski, S. Jovanovski, M. Gams, H. Gjoreski, Houseec: Day-ahead household electrical energy consumption forecasting using deep learning, Energies 13 (2020). doi:10.3390/en13102672.

[8] D. Ageng, C.-Y. Huang, R.-G. Cheng, A short-term household load forecasting framework using lstm and data preparation, IEEE Access 9 (2021) 167911–167919. doi:10.1109/ACCESS.2021.3133702.

[9] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE Transactions on Neural Networks 5 (1994) 157–166. doi:10.1109/72.279181.

[10] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, Y. Zhang, Short-term residential load forecasting based on lstm recurrent neural network, IEEE Transactions on Smart Grid 10 (2019) 841–851. doi:10.1109/TSG.2017.2753802.

[11] R. K. Agrawal, F. Muchahary, M. M. Tripathi, Long term load forecasting with hourly predictions based on long-short-term-memory networks, in: 2018 IEEE Texas Power and Energy Conference (TPEC), 2018, pp. 1–6. doi:10.1109/TPEC.2018.8312088.

[12] G. Alicioglu, B. Sun, A survey of visual analytics for explainable artificial intelligence methods, Computers & Graphics 102 (2022) 502–520. doi:https://doi.org/10.1016/j.cag.2021.09.002.

[13] T. Munzner, Visualization Analysis and Design, AK Peters Visualization Series, CRC Press, 2014.