

Article

A Cloud-Based Framework for Creating Scalable Machine Learning Models Predicting Building Energy Consumption from Digital Twin Data

Elham Mahamedi ^{1,*}, Alaeldin Suliman ^{1,2}  and Martin Wonders ³ 

¹ Department of Architecture and Built Environment, Faculty of Engineering and Environment, Northumbria University, Newcastle upon Tyne NE1 8ST, UK; ala.suliman@northumbria.ac.uk

² Department of Civil Engineering, University of New Brunswick, Fredericton, NB E3B 5A3, Canada

³ Department of Computer and Information Sciences, Faculty of Engineering and Environment, Northumbria University, Newcastle upon Tyne NE1 8ST, UK; martin.wonders@northumbria.ac.uk

* Correspondence: e.mahamedi@northumbria.ac.uk

Abstract: Digital Twins (DTs) of buildings can generate large volumes of dynamic data from various sources (e.g., sensors and IoT devices), enabling real-time representation of physical building states in a digital environment. Although machine learning (ML) techniques are increasingly used to predict building energy consumption from this DT data, existing approaches often lack scalability in handling data growth (data scalability) and/or adapting to evolving data patterns (model scalability). This study aims to address both drawbacks by developing a scalable cloud-based framework for the prediction of the building energy consumption. A key contribution to the field is the inclusion of a “monitoring and maintenance” module, which continuously evaluates model performance and triggers retraining only when needed. This enables timely adaptation of the ML model while avoiding unnecessary retraining and the associated computational costs. The framework was implemented and tested in a case study of a commercial building for 90 days, demonstrating its applicability. In a practical setting, the developed model could detect anomalies in time when the accuracy declined below the set threshold (70%) for five days and prevented unnecessary retraining of ML models. The findings support the feasibility of using cloud-based approaches to implement scalable ML models for energy prediction in buildings.

Keywords: digital twin; building energy; cloud computing; machine learning; scalable models



Academic Editor: Antonio Messineo

Received: 1 April 2025

Revised: 14 April 2025

Accepted: 16 April 2025

Published: 23 April 2025

Citation: Mahamedi, E.; Suliman, A.; Wonders, M. A Cloud-Based Framework for Creating Scalable Machine Learning Models Predicting Building Energy Consumption from Digital Twin Data. *Architecture* **2025**, *5*, 29. <https://doi.org/10.3390/architecture5020029>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

A total of 40% of global energy use and 30% of the global greenhouse gas (GHG) emissions are attributed to the building sector [1]. In the UK, the built environment produces 25% of the total greenhouse gas emissions [2]. Many countries have set a target to reduce carbon emission to mitigate climate change impacts. For instance, the UK aims to reduce carbon emissions by 78% by 2035, compared to 1990 levels, and reach net zero by 2050 [2]. Improving building energy performance plays a crucial role in achieving these targets, as emphasized by the UK’s net zero strategy [3]. The adoption of emerging digital technologies in the building sector can support the endeavors in achieving these set goals. DT is one of these technologies that can improve building energy management by monitoring, optimizing, and predicting building energy consumption in real-time [4]. DT was coined by Grieves [5] and is defined as a “virtual replica of the physical asset”,

which interconnects the two entities with bidirectional flow of real-time information to improve decision making based on its ability to monitor, compare, predict, and conduct what-if analyses [6]. For the information flow from physical assets to the digital model, first, it requires sensing technologies, such as the Internet of Things (IoT), radio-frequency identification (RFID), and QR code, to capture the data from the physical asset [7]. Then, communication technologies such as wireless communication (i.e., Wi-Fi, Bluetooth, and 5G), and cloud data pipelines to transfer the data to the digital model are required [7]. The information flow from the digital model to the physical asset requires data integration and data visualization techniques to present the current condition of the asset, and predictive analytics using various methods and techniques such as ML, simulation, and statistical methods to extract actionable insights from the data and support informed decision making.

DT can provide the required data for model-free approaches for building energy control, and data-driven approaches for building energy prediction in real-time. In the literature, several studies (e.g., [8–10]) have used DT data for creating data-drive models using ML to predict the energy consumption of buildings. While in real world applications, sensory data from buildings is continuously generated and stored in DT models to represent the real-time status of the buildings, these studies used stationary historical data for developing their ML models and overlooked the scalability of the models. Scalability of AI/ML models could have several dimensions. In this research, two dimensions of scalability are considered: (1) data scalability, which refers to the ability (computational power) of a system to handle increasing data quantities, and (2) model scalability, which refers to the ability of ML models to adapt to changes in data patterns and address new use cases [11]. These two dimensions are selected due to the computational need for handling an increasing amount of data constantly generated by DT, and the dynamic nature of building energy performance influenced by several factors varying overtime such as occupant behavior. Some studies (e.g., [12,13]) have attempted to address data scalability in their DT framework by proposing cloud computing as a suitable technology for transmitting and storing a large volume of data. These studies demonstrated the suitability of cloud computing in handling an increasing volume of DT data; however, they have not properly addressed model scalability in their frameworks.

Model scalability requires the adaptability of ML models to the changing conditions of buildings. Some studies have developed adaptive ML models for enhancing building energy. There are two main approaches for triggering the initiation of the adaptation process: (1) triggering the cycle on a regular basis (a fixed interval approach) or (2) triggering the adaptation process after significant changes in real-time conditions (an event-based approach) [14]. Several studies have used the first approach and adapted their ML models by incorporating newly generated data into the model in a fixed time interval. For instance, ref. [15] developed adaptive ML-based building models for building automation and control applications. Their models could update the building model regularly every 24 h using online building operation data. They showed that the model adaptation feature can improve the prediction accuracy of the predictive models, and the prediction accuracy continues to improve as more data become available for the models. Ref. [16] developed an adaptive ML approach for the hourly prediction of building energy by incorporating new building data into the model every two weeks. While the fixed interval approach is easy to implement, there is no universal method for determining the optimum time interval for retraining. Relatively fewer studies focused on an event-based approach such as [17] that developed an event-triggered paradigm by setting temperature thresholds for the learning and control of micro-climate in buildings. They demonstrated that their proposed approach could improve the controller's performance measure by 70% and outperform the fixed time interval approach. However, these studies did not consider data scalability

in their approach. To the best knowledge of the authors, there is only one study [6] that has addressed both data scalability (by deploying cloud computing for handling DT data) and model scalability (by adapting ML models). However, this study used the fixed interval approach for retraining the models, which is not a very efficient approach because retraining the models too early entails extra computational cost and time and retraining the models too late leads to the underperformance of ML models. In addition, the dynamics of building energy requires adaptability to foreseeable and unforeseeable changes, which is not addressed thoroughly by a fixed retraining interval in this approach.

This research aims to bridge this gap and propose a framework that efficiently addresses both data scalability and model scalability of ML models for predicting energy performance of buildings from DT data. To this end, the objectives of this research are (i) developing a seamless data pipeline for transmitting DT data to ML models, training and executing the models, and visualizing the data in a DT dashboard on a single cloud platform, which addresses data scalability; (ii) devising a novel monitoring module to provide continuous feedback about the accuracy of the ML models and alert the users about any potential need for adapting the ML models, which addresses model scalability more efficiently; and (iii) implementing the framework in a case study to test and validate its applicability and practicality. In this study, Microsoft Azure is used as a cloud platform for developing the framework. The devised monitoring module adopts an event-based approach for triggering the initiation of the ML model adaptation. This module continuously monitors the defined accuracy metrics, and a fixed threshold is set to alert the user when the accuracy declines below the threshold. Upon receiving an alert, the user can decide whether the ML model adaptation is required and initiate it if needed.

In the next sections, first, the literature review on DT applications for improving building energy performance and cloud computing technology for creating DT is presented. Then, the research methodology and the proposed framework are described. Next, the applicability and practicality of the proposed framework are tested and validated in a demonstration case study, followed by a discussion on the evaluation of the framework. Finally, the research summary and conclusion along with some recommendations for future research are provided in the last section.

2. Literature Review

This section reviews the literature on DT applications for improving building energy performance, and cloud computing applications to enhance the scalability of DT, which are presented in the following subsections.

2.1. Digital Twin for Improving Building Energy Performance

Several studies have been conducted on leveraging DT to enhance building energy performance and reduce GHG emissions. Some studies have focused on reality capture technologies and investigated how to adopt these technologies for creating 3D models of existing buildings and using the models for visualization and energy analysis. For instance, ref. [18] used the BIM models created through 3D laser scanning and scan-to-BIM technologies for simulating building energy to identify and evaluate the feasibility of existing building retrofitting schemes based on the concept of nearly zero-energy buildings (nZEBs). Their approach could reduce building energy costs by 14.1%, increase solar photovoltaic power generation by 24.13%, and reduce carbon dioxide emissions by 4306.0 kg CO₂ eq/a (kilograms of carbon dioxide equivalent per year) in a case study. For using DT at the macro level, ref. [19] developed a geographic information system (GIS)-enabled DT framework to propose an optimal strategy for reducing carbon emissions in urban areas. In their framework, they deployed various data sources including monthly electricity usage, city gas

consumption, monthly household waste, and the amount of emitted CO₂. They analyzed these data using ML techniques to predict spatial trends of carbon emissions and finally visualized the results on a GIS-enabled dashboard for decision making.

Some DT models have been created to evaluate the impact of occupants' behavior on building energy performance. For instance, ref. [20] created a novel Occupant-Centric Digital Twin (OCDT) to explore the impact of live IoT temperature data on occupants' thermal satisfaction. OCDT enabled occupants to make choices about where to work based on their individual thermal preferences by providing them with a live thermal map. Another study [21] developed a DT-based framework integrating smart sensor data, real-time measurements, and BIM for the operational rating of buildings' energy performance.

DT is also being used for building energy monitoring as it can provide information related to energy performance of buildings in real-time. In one study, ref. [22] used interpretable regression models to give insights on the energy performance of the building. In this study, the dynamics of DT was considered by periodically calibrating the models and recomputing model coefficients. In another study, ref. [23] used Natural Language Processing (NLP) to advance applications of DT in automated monitoring for existing buildings by creating semantic Digital Twins of buildings, their systems, and technical components.

These studies demonstrated the benefits of DT in enhancing the energy performance of buildings in some pilot studies or small-scale projects. However, they have not considered the scalability of DT, which is one of the root causes of limited DT implementations on an industrial scale [24]. In the next subsection, the studies on using cloud computing as a technology that can address this challenge are reviewed.

2.2. Cloud Computing for Digital Twin

Due to the dynamic nature of DT, a large amount of data needs to be collected, transferred, and analyzed over the lifecycle of a project, which could span over several years. Handling this amount of data is a critical challenge, which could limit the data scalability of DT. Cloud computing is a technology that can be integrated with Digital Twins for streamlining extensive data storage and processing to ensure scalability and accessibility [25]. Ref. [26] proposed a five-layer IoT architecture, including (1) the perception layer (2) the Network Layer, (3) the middleware layer, (4) the Application Layer, and (5) the Business Layer, for developing linked devices in DT, and identified how to incorporate cloud computing into the development of DT in the proposed architecture.

Most studies that have developed cloud-based solutions for DT in the context of building energy performance have focused on the perception layer and middleware layer. In the perception layer, physical devices linked to the Internet collect data from the environment and transfer it to the cloud for further analysis [26]. Cloud computing can handle transferring and storing a large amount of DT data to facilitate the information flow from the physical asset to the digital model while enhancing data security and accessibility. For instance, ref. [27] developed a BIM-based DT for asset management and used cloud computing to host their developed intelligent platform and collect sensor data such as temperature and humidity with secured accessibility.

In the middleware layer, the data collected from the devices and stored in the database are processed and made available for further analysis using data analysis and interpretation technologies such as AI and ML [26]. Some researchers have deployed cloud computing for the middleware layer to visualize data and provide insights by further analyzing the data. For structural health monitoring and analyzing sensor data in real-time, ref. [28] transmitted the sensor data to an integrated cloud platform, which stored the data and displayed the real-time monitoring data, statistical parameters, and BIM models of the structures. In

another study, ref. [29] developed a DT model for sustainable comfort monitoring in a smart campus using an emotion detection system. Their system consisted of capturing the faces of the occupants with a camera lens assembled in a Raspberry Pi 3b+ sensor and used cloud computing for transferring and analyzing the captured data. In a recent study, ref. [6] created a DT model by collecting IoT data and visualizing it on the 3D BIM models. Finally, they used ML to predict CO₂ equivalent emissions and support decision making in facility management processes. In this study, Microsoft Azure was used as a cloud platform for connecting IoT devices with the DT model and integrating their real-time data with the BIM model for visualization.

Table 1 summarizes the applications of cloud computing in DT for enhancing building energy performance and how they addressed scalability. As seen in Table 1, all the reviewed studies used cloud computing to address data scalability in transferring, storing, and visualizing an increasing amount of DT data, and most of them neglected addressing model scalability for ML models. Only [6] attempted to address model scalability through retraining ML models every 15 days and incorporating newly generated data within a 15-day period into the retraining process. As mentioned earlier in the Introduction Section, it is an easy-to-implement approach, but no universal method has been developed for determining the optimum time interval for retraining ML models. Hence, this study aims to address both data scalability and model scalability in a more effective manner. In the Discussion Section, the approach proposed by [6] is compared with the new approach proposed in this study, and their effectiveness is evaluated.

Table 1 also presents the cloud platform used in different studies. As seen in Table 1, Microsoft Azure and Amazon AWS are the commonly used cloud platforms in developing DT for enhancing building energy performance. Microsoft Azure has a Digital Twin platform that provides Digital Twin architecture to create digital models of physical objects, such as buildings and factories, and manages them through real-time monitoring of asset performance and predictive analytics to optimize operations [30]. AWS offers an IoT Twin Maker service that enables importing pre-existing 3D models of the physical system components to develop a Digital Twin and overlaying the models with knowledge graph data to capture the relationships and properties of the physical system and create a realistic and dynamic Digital Twin [30].

Table 1. Studies that used cloud computing for middleware layer.

Study	Purpose of DT	Used Cloud Platform	Cloud Computing for Transferring and Storing Data	Cloud Computing for Data Visualization	Cloud Computing for Enhancing Data Scalability	Cloud Computing for Enhancing Model Scalability
[6]	Predictive monitoring of CO ₂ equivalent from existing buildings	Azure	Yes	Yes	Yes	Yes
[29]	Monitoring level of comfort in buildings	Azure and AWS	Yes	Yes	Yes	No
[27]	University campus management (asset management)	AWS	Yes	Yes	Yes	No
[26]	Modeling indoor thermal comfort in buildings	Azure	Yes	Yes	Yes	No
[31]	Monitoring the indoor environment of historic buildings	Azure	Yes	Yes	Yes	No

Table 1. Cont.

Study	Purpose of DT	Used Cloud Platform	Cloud Computing for Transferring and Storing Data	Cloud Computing for Data Visualization	Cloud Computing for Enhancing Data Scalability	Cloud Computing for Enhancing Model Scalability
[32]	Modeling of indoor environmental quality, electricity consumption, and user behavior	Not specified	Yes	Yes	Yes	No
[33]	Improving energy efficiency of indoor lighting	Not specified	Yes	No	Yes	No

In summary, it is understood that cloud computing is a promising technology to enhance the scalability of DT. However, most of the existing studies have focused only on data scalability, and they have not properly addressed model scalability. Therefore, this study proposes a framework that leverages cloud computing for addressing both data scalability and model scalability in analyzing DT data using ML to predict building energy consumption.

3. Research Methodology

In this research, the Design Science Research (DSR) method is adopted as a research methodology. DSR, which aims to generate knowledge of how things can and should be constructed [34], encompasses six main processes including (1) problem identification and motivation, (2) define the objectives for a solution, (3) design and development, (4) demonstration, (5) evaluation, and (6) communication [35]. If needed, this process can be iterated to refine objectives and/or the design and development. DSR is a suitable method for our research because our aim is to develop a framework as a solution for the identified problem, and DSR is a systematic method for clearly defining the problem and developing a problem-driven solution ensuring that the solution meets the defined objectives for the solution. In addition, developing a cloud-based ML framework requires several iterations for development, testing, and refinements, and DSR offers an iterative cycle of design, development, and evaluation, which aligns well with the requirement of this research. The six processes of DSR are followed in this research as described below.

1. Defining the problem: The problem is how to adopt cloud computing to develop and execute ML models using dynamic DT data for predicting the energy performance of buildings.
2. Defining objectives and requirements of a solution: The main objectives of the solutions are to (1) develop a seamless data pipeline for transmitting DT data to ML models, training and executing the models, and visualizing the data in a DT dashboard on a single cloud platform; and (2) devise a novel monitoring approach to provide continuous feedback about the accuracy of the ML models. The requirements of the solution are (i) addressing data scalability for handling an increasing amount of DT data and (ii) addressing model scalability to maintain performance of ML models at an acceptable level over time by adapting ML models when needed.
3. Design and development: Design and development of a framework using one of the common cloud platforms (i.e., Microsoft Azure) that has elements for handling DT data, developing and executing machine learning, and analyzing data.
4. Demonstration: Conduct a case study as an instance to test and validate the developed framework in predicting the energy consumption of a building in a scalable manner.

5. Evaluation: Analyze the results of the case study (demonstration) to assess the practicality and applicability of the framework. If any refinement to the framework is needed, the previous steps are repeated as needed to ensure that the framework can address the identified problem and meet the objectives and requirements.
6. Communication: Disseminate the research outcomes to researchers and practitioners (i.e., publishing this paper).

4. Proposed Framework

In this research, prior to developing the required framework, a conceptual model was created to identify the key components and their functionalities, as shown in Figure 1. The conceptual model has three main sequential modules for executing ML models: # (1) data transfer and data preparation module, which is required for transferring DT data to the ML models, and pre-processing the data as a preparation prior to training the ML models; # (2) model development and training, which is required for developing and training/retraining ML models using the pre-processed DT data; and # (3) model deployment and interface, which is required for executing ML models and visualizing the model input and output data on a dashboard. To address the requirement of data scalability, a seamless data pipeline transmitting DT data to ML models, training the models, and visualizing the data needs to be developed on a single cloud platform. It should be noted that in this research, it is assumed that a DT model has already been created, the reality capture technologies (e.g., sensor and IoT) are in place, and the required data are being collected. Therefore, the proposed framework focuses only on processing the DT data by developing and executing scalable ML models.

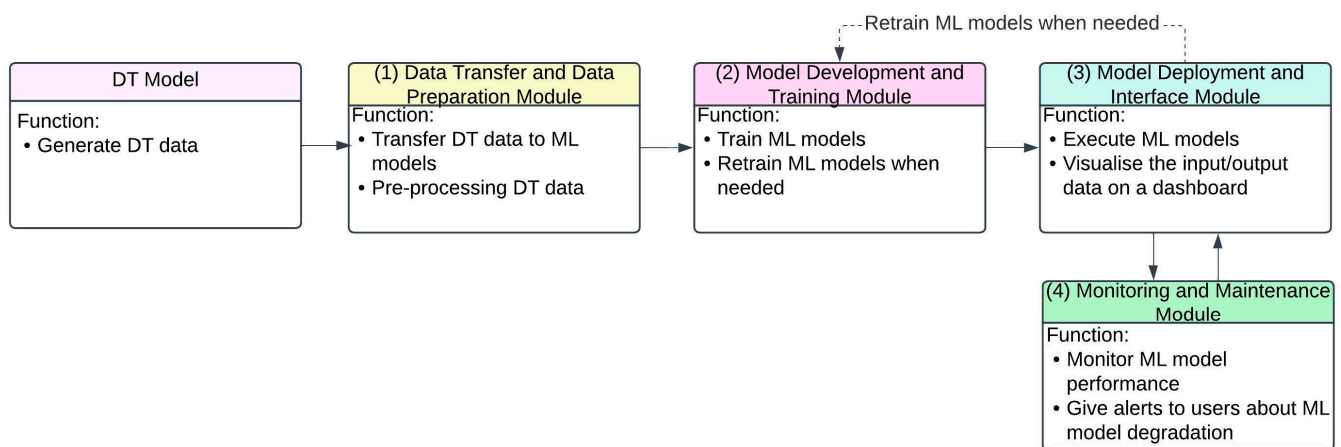


Figure 1. A conceptual model for developing the framework.

In addition to these three modules, a novel (4) monitoring and maintenance module is incorporated in the process, which is a key component to address model scalability by continuously monitoring the performance of ML models and providing alerts to users about ML model degradation and any potential need for adapting ML models. The monitoring and maintenance module receives the ML model performance data from the model deployment and interface module and sends the alert back to this module when any model performance degradation is detected. Analyzing the input and output data, the user can decide whether retraining the ML is needed or not. If needed, the ML model is retrained in the model development and training module, and this process is repeated.

Based on the conceptual model, a framework was developed in Microsoft Azure, as shown in Figure 2. Microsoft Azure was chosen as it is one of the most common cloud platforms that have required services for DT. Therefore, all the required functions can be

developed on a single platform, which enhances data scalability. Microsoft Azure offers the “Azure Digital Twins” (Azure DT) service, which is used for designing and developing DT architecture [36]. “Azure IoT (Internet of Things) Hub”, which is a central message hub for communication between an IoT application and its attached devices, is another service that complements Azure Digital Twins by scaling to millions of simultaneously connected devices and managing tons of events every second to support IoT workloads [37]. Azure Machine Learning (Azure ML), which is used for accelerating and managing the machine learning project lifecycle, and supports monitoring, retraining, and redeploying models [38], is a key component for training and executing scalable ML models using dynamic DT data. The detail of the processes in the framework is presented in the following subsections.

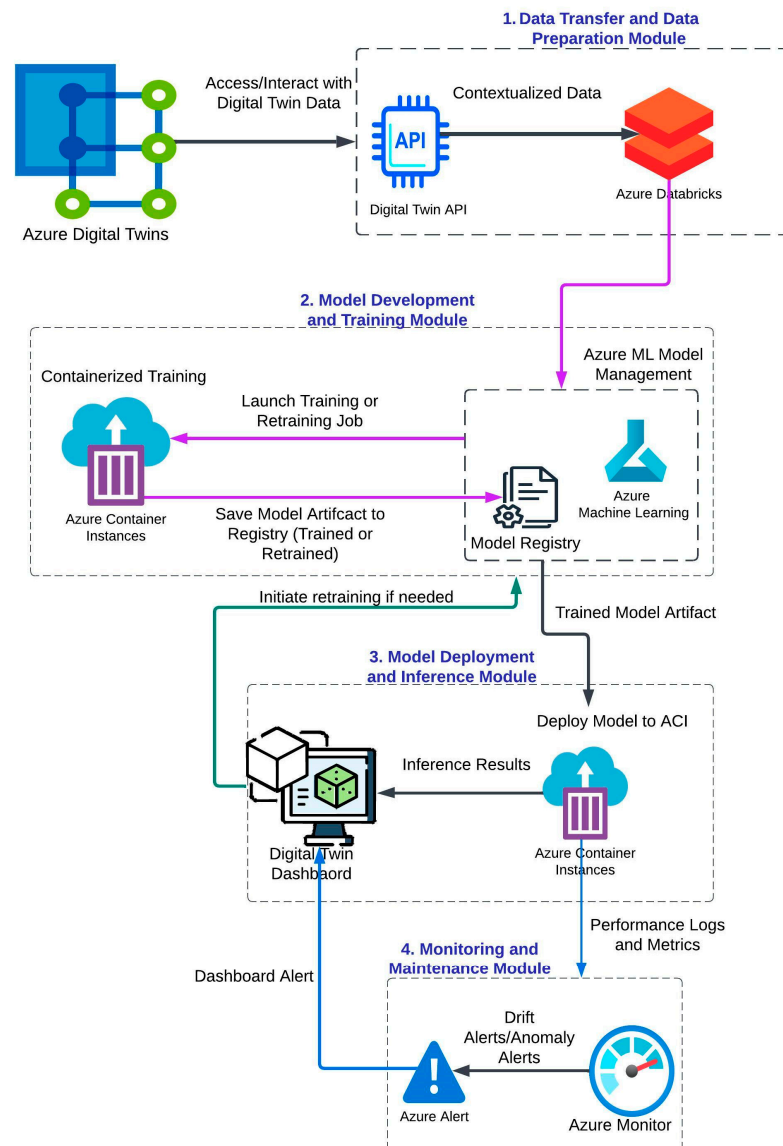


Figure 2. The proposed framework in Microsoft Azure.

4.1. Data Transfer and Data Preparation Module

The framework starts with Azure Digital Twin (Azure DT), which is the first element in this workflow. Azure DT contains related data from a physical asset to represent its state in real-time. This information could be sensor/IoT readings, states of devices, or other operational history that supports any data analytics or prediction by ML models in the workflow.

Azure provides the Digital Twin API to transfer these data to external services. This API serves as a link between Azure DT and other elements by providing contextualized data [36]. Contextualized data are the data that are enriched by relevant information and details (e.g., related building element or building space) derived from the Digital Twin model, which is useful for data analytics and machine learning applications. Using Digital Twin API, an application or service can easily access, query, and receive Digital Twin data for processing [39] in a structured, secure, and efficient manner, making it a key connection point for the upcoming processes.

In the next step, the data obtained from the API is pre-processed using Azure Databricks to make the data suitable for developing machine learning models. Azure Databricks, which is a unified analytics platform, helps process large amounts of raw data (from Azure DT) and prepare it with quality, structured, sanitized, and enriched features [40]. Since sensor and IoT data often contain noise or missing values that may degrade machine learning models, this pre-processing step is essential. Databricks provides tools to clean the data, as well as normalizing and data engineering, ensuring the data are accurate and fit for specific modeling requirements.

After preparing the data, it is transferred to Azure ML, which supports the full lifecycle of machine learning projects. At this stage, the data are processed for training machine learning models. This part of the pipeline, which is from Azure DT to Azure Databricks and Azure ML, creates the basis for developing intelligent systems. This supports the integration of Azure data management tools with machine learning platforms to ensure that the ML models are trained over the contextualized data. This step provides the foundation for accurate and usable input in the next two modules (i.e., model development and training module, and model deployment and interface module), as described in the next subsections.

4.2. Model Development and Training Module

Creating and training ML models that predict building energy consumption with an acceptable level of accuracy is an important component of the “Model Development and Training” module. The process in this module starts in the Azure ML Model Management environment, which consists of Azure ML and the Model Registry. Azure ML simplifies the training process by offering a space where experts in data can create, test and analyze the models [38], which enhances scalability and effectiveness by utilizing Azures capabilities to manage ML operations [41]. The Model Registry serves as a central repository for the models, storing and versioning them [42], which is a crucial task in ML model development and training. Model Registry designs proper model security for trained or retrained models with version control and assists experimental models to be transferred to a production stage [42].

To train or retrain models, the workflow transitions from the Model Management group to Azure Container Instances (ACI), which serves as the execution environment for containerized training tasks [43]. ACI is a serverless, on-demand container hosting service within the Azure platform, designed to run containerized workloads in a lightweight, cost-effective, and isolated environment [43]. As building energy predictions often require large datasets and complicated calculations during training, ACI offers the ability to run containerized training scripts. This allows for the efficient use of computational resources since they will be pooled and scaled up or down as necessary without the overhead of maintaining physical infrastructure [43].

For training ML models, a suitable ML algorithm needs to be selected and deployed for prediction. In this study, a type of Recurrent Neural Network (RNN) algorithm is considered for developing ML models. RNN is a suitable approach for processing and

learning long-term dependencies of time series data that are generated by DT. At the completion of the ML model training or retraining process, the generated models are pushed to the Model Registry. This flow is denoted by an arrow annotated as “Save Model Artifact to Registry (Trained or Retrained)” in Figure 2. This action indicates that the latest model, which could be a newly created mode or a refined model through retraining, is kept safely.

Incorporating Azure ML, ACI, and Model Registry provides a process that is efficient, repeatable, and scalable to accelerate the development and/or refinement of ML models by new data when needed. As DT constantly generates a large amount of data, ACI allows for accelerated processes for training and testing algorithms and retraining the models with new data to maintain the ML model’s performance. The trained ML model is then transferred to the “model development and interface” module for deployment and visualization of data.

4.3. Model Deployment and Inference Module

In this module, the initial step is to deploy the model artifact (artifact is “any file generated and captured from an experiment’s run or job” [44]) that was trained in the previous step and stored in the Model Registry. The Model Registry helps make sure the right and the latest version of the model is used while deploying the model. This step is necessary for version control and governance to ensure that the workflow is consistent and traceable across various steps in the machine learning lifecycle [42].

After selecting the model artifact, it will be deployed to ACI. In this scenario, ACI acts as a light, scalable, and effective containerized environment to perform inference tasks [43]. ACI allows for a quick deployment of the trained model for real-time predictions, abstracting the details for managing dedicated infrastructure [43]. These steps allow fast processing of inference requests to produce predictions of the building energy consumption.

The inference from ACI deployment is sent to the Digital Twin Dashboard, which is the interface to the user to view the prediction results. These results along with the input data can be visualized and analyzed using the Digital Twin Dashboard, which promotes data-informed decision making.

4.4. Monitoring and Maintenance Module

The monitoring and maintenance process plays an important role in ensuring the acceptable performance of the model after deployment, particularly for tracking model drift (i.e., degradation of ML models’ performance over time when the models are deployed in production environments [45]), anomalies, and retraining requirements. One of the underlying issues of ML model for energy prediction is “concept drift” [46], which refers to the change in the statistical properties of the target variable over time in unforeseen ways [47]. According to [48], there are several drift detection algorithms that are classified into three main categories: (1) error-based drift detection, which focuses on “tracking changes in the online error rate of base classifiers”; (2) data distribution-based drift detection, which uses “a distance function/metric to quantify the dissimilarity between the distribution of historical data and new data”; and (3) multiple hypothesis test drift detection, which uses multiple hypothesis test such as Kolmogorov–Smirnov test or the Cramer–Von Mises test to detect the drift in various ways. In this study, the error-based drift detection method is adopted by tracking a performance measure (i.e., accuracy) of ML models.

The monitoring and maintenance module contains two major steps to identify a reduction in model performance, and any potential need for retraining. The following subsections describe the main steps for monitoring and maintenance.

4.4.1. Integration of Azure Monitor and Azure Alert

Azure Monitor continuously monitors model performance metrics to detect any anomaly and/or potential drift. Azure Monitor is seamlessly integrated with other Azure services to make sure no performance degradation, model drift, or anomaly is overlooked.

From Azure Monitor, the process feeds into Azure Alerts, which is used as the alerting mechanism that sends notifications when the performance of the model is deteriorated below an acceptable threshold. This implies that there is a potential need for retraining the models. Performance metrics calculated for ML models indicate when the accuracy or other measures of the model decline over time, which can be a sign of data drift (i.e., “the change in model input data that leads to model performance degradation” [49]).

4.4.2. Digital Twin Dashboard Alerts

The alerts received in Azure Alerts are displayed on Digital Twin Dashboard allowing users to make appropriate decisions about the required actions following the alerts. The user can monitor the distribution of the input data in real-time in visual charts to identify significant deviation between the current data and historical data, and any potential drift or anomalies. This insight from the Digital Twin Dashboard can guide the user to figure out the necessity of the retraining process. For example, users can tag data anomalies due to the malfunction of IoT devices. Anomaly detection mechanisms are also used to identify abnormal records, which are instances where actual outcomes deviate significantly from what the model predicted. If retraining the ML model is needed, the user initiates its process in the Azure ML. This helps Azure ML integrate feedback from the dashboard into the model production. This iterative loop of feedback and retraining helps maintain model accuracy and mitigates the risk of performance degradation over time, which can address the dynamic nature and changing conditions of the building energy.

In summary, the devised continuous feedback loop in the framework helps ML models keep learning from new data and any changes in the environment and ensures that the models remain in an acceptable accuracy level throughout their operational lifecycle. Retraining ML models only when needed allows the model to adapt to the complexities of the real world, ensuring model scalability.

5. Case Study

The proposed framework is implemented in a case study of a commercial building to predict its energy consumption. We used Azure Machine Learning Studio for development and Python 3.10 as a programming language. The following subsections outline the implementation for each module of the framework.

5.1. Implementation of the Data Transfer and Data Preparation Module

In this module, the generated DT data are transmitted from Azure DT API to the Azure ML via Databricks. Databricks transforms the raw DT data into a format suitable for ML models through various processes such as data cleaning, normalizing, and data engineering. In this case study, the DT data used for training the ML model includes (1) temperature, (2) wind direction, (3) wind speed, and (4) dew point temperature, and the output for the prediction is the energy (electricity) consumption of the building. Since the data are time-dependent, temporal feature engineering was performed to create time-based features including quarter, month, week of year, day of year, day of week, a day, and weekend/weekday. These features help ML models understand patterns and seasonality in time series data. To impute data (i.e., substituting missing values in a dataset [50]), the linear interpolation technique was used. This technique, which estimates the missing values by drawing a straight line between the known data before and after the missing

data, is a suitable technique in weather data imputation because weather conditions such as temperature, dew point, wind speed, and wind direction generally change gradually. The DT data collected on an hourly basis were aggregated in Databricks, and their daily average was calculated to be used in Azure ML. Then, Azure ML predicts the daily electricity consumption of the building.

5.2. Implementation of the Model Development and Training Module

To develop and train the first ML model in the “model development and training” module, two years of historical building data are used. In this case study, Bidirectional Long Short-Term Memory (Bi-LSTM), which is a type of Recurrent Neural Network (RNN), is employed as an algorithm for developing ML models. Long Short-Term Memory (LSTM) has feedback connections allowing for processing the time dependency between sequential data that can address the long-term dependency of the data [51]. Unidirectional LSTM and Bidirectional LSTM are common ML algorithms for analyzing time series such as sensory data that have been adopted in many studies (e.g., [52–57]) for predicting building energy performance. Most of the RNN algorithms (including Unidirectional LSTM) only process the past data (one direction), while processing the future data as well as the past data of a given point in the series (bidirectional) could be useful. Bi-LSTM has addressed this shortcoming by using the history of previous hidden states in addition to the information of the future hidden states and showed a better performance in predicting building energy consumption comparing to Unidirectional LSTM [58]. Therefore, Bi-LSTM was selected as a suitable ML algorithm in this study.

After training the first ML model, the created model is pushed to the Model Registry to safely store the model. When any retraining procedure is triggered, the process of retraining the ML model is performed by incorporating the newly generated DT data into the training processes. Then, the Model Register as a central repository keeps all the versions of the ML models with the proper version control for the deployment. It should be noted that prior to incorporating new data in the retraining procedure, the data need to be processed and cleaned through Databricks.

5.3. Implementation of the Model Deployment and Inference Module

In this module, first, the right version of the trained ML model is selected in Model Register. Then, the selected model is deployed to ACI to perform inference tasks and real-time predictions. A DT dashboard is also developed in the “model deployment and interface” module. The inference from ACI deployment is sent to the DT dashboard to visualize the sensor data as well as the prediction data versus the actual values of the energy consumption. The user can navigate through the data by selecting various timeframes for visualization.

For measuring the performance of the ML model, the accuracy of the model is calculated using the following formula:

$$Accuracy = 100 \times \left(1 - \frac{|y_{true} - y_{pred}|}{\max(y_{true}, y_{pred})}\right) \quad (1)$$

where y_{true} represents the actual value and y_{pred} represents the predicted value. This formula calculates the closeness of the predicted value to the actual value while normalizing the error based on the maximum of the two values. This performance metric could be more useful than other formulas that have only actual values in the denominator because they

return to a high or a negative number when the actual values are relatively small. For instance, Mean Absolute Percentage Error (MAPE) is calculated as the following formula:

$$MAPE = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_{true(i)} - y_{pred(i)}}{y_{true(i)}} \right| \times 100 \quad (2)$$

where the actual values are in the denominator.

The calculated accuracy, as defined in Formula (1), is sent to the “monitoring and maintenance module” for monitoring the ML model’s performance.

5.4. Implementation of the Monitoring and Maintenance Module

In the “monitoring and maintenance” module, the value of the accuracy of the ML model is monitored, and the user can set a threshold for the accuracy, where, when the accuracy declines below that threshold, an alert is generated through integration of Azure Monitor and Azure Alert. Then, the alert is sent back to the “model deployment and interface” module to display the alert on the dashboard to notify the user. In this case study, an alert was set to trigger if the accuracy drops below 70%.

Following the alert, the user can investigate the root causes of the low ML model performance through reviewing the data and operational conditions of the building. If needed, retraining of the ML model is initiated by the user to adapt the ML model to the changing conditions. The model is retrained in the “model development and training” module and stored in the Model Registry. Then, the retrained ML model is deployed in the “model deployment and inference module”. This process is repeated throughout the lifecycle of the project.

6. Results

Training the ML model using the historical data on a container configured with 4 vCPUs and 32 GB of RAM, equivalent to the Azure VM size Standard_E4ds_v4 took about 9 min. Throughout training, the learning curve of the ML model was monitored to make sure that the training process is healthy. To this end, training loss, which refers to the error the model makes on the data it learns from, and validation loss, which illustrates the error that the model makes on unseen data, are monitored in each complete training iteration, which is called an epoch. Figure 3 shows how the values of training loss and validation loss changed over 36 epochs. As seen in this figure, the value of training loss was significantly declined at the beginning, which indicates effective learning at the early stage. This value continues to decline gradually, which shows that the model is fitting the training data to achieve higher accuracy. On the other hand, the value of the validation loss fluctuated throughout the training iterations, but the range of the fluctuation became smaller, which indicates stabilization of validation loss over time. Analyzing the learning curve confirms the healthy training process.

In this case study, 90 days (3 months) of DT data were streamed through Microsoft Azure for energy consumption prediction. During the first 69 days of the prediction and observation, the accuracy of the ML model ranged from 73% to 99%. However, on Day 70, the accuracy declined to 63.9%. Figure 4 shows the value of actual energy consumption versus the predicted value over 90 days.

When the significant decline in the accuracy of the ML model was detected on Day 70, an alert was sent to the user to notify them about potential drift or anomaly in the model. The user investigated the input data and the ML model to detect any potential anomaly or error. The DT dashboard can help with this investigation by visualizing the input and output data. Figure 5 shows a DT dashboard visualizing data for the input data of the ML model (the four charts on the top) and the ML model prediction values versus the actual

values (the chart on the bottom left). There is also an alert log in the dashboard showing the history of the alerts received over the past days.

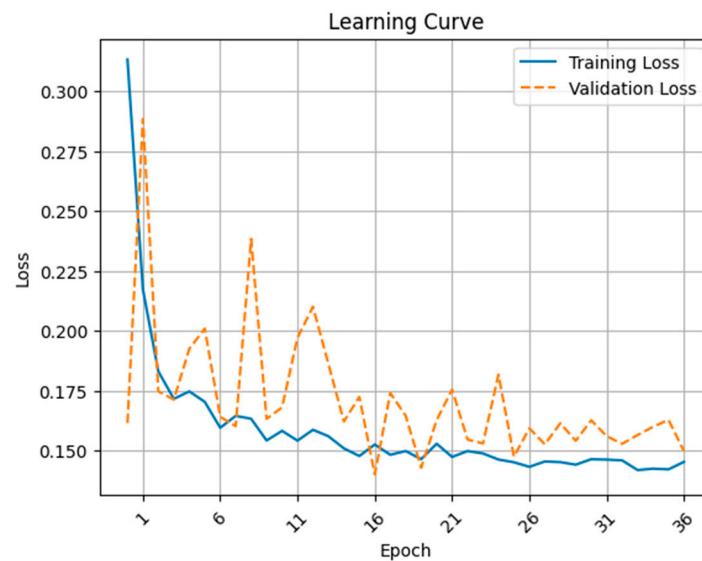


Figure 3. Learning curve of training ML model using historical data.

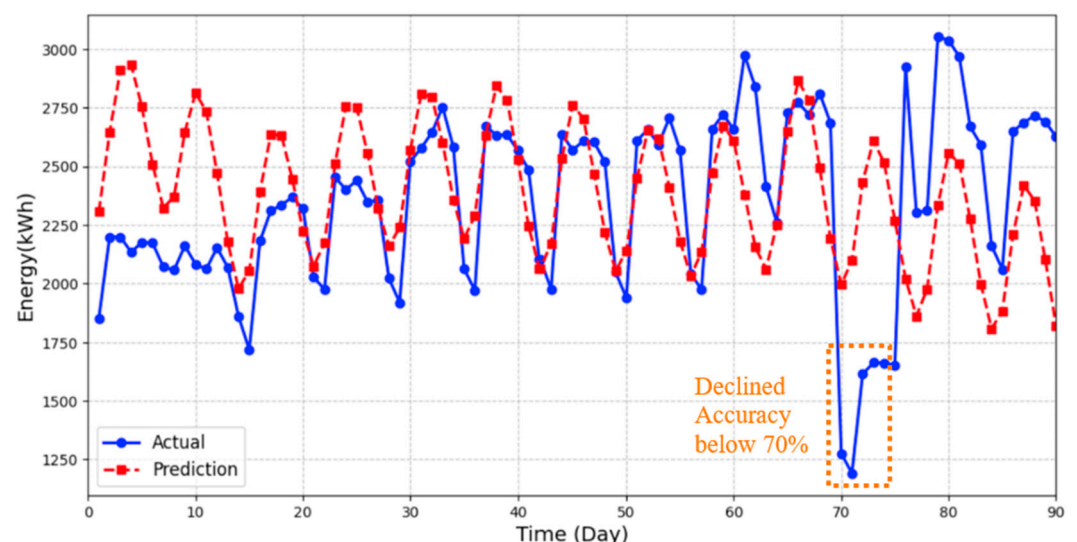


Figure 4. Actual versus predicted values of energy consumption in case study.

To diagnose the accuracy issue, the DT data were reviewed, and it was revealed that there is a significant decline in the actual energy consumption from Day 70 to Day 74 as shown in Figure 4. Investigating the root of this decline, it was revealed that the building was shut down for major maintenance and the consumed energy was much lower than anticipated under normal circumstances. Because of these temporary changes in energy consumption, and an acceptable level of accuracy before and after this anomaly, it was realized that there is no need for retraining the ML model.

This case study shows how the devised modules are scalable by handling an increasing amount of DT data and adapting ML models when needed. The monitoring and maintenance module is of great assistance in notifying users about any drift or anomaly in a timely manner. In addition, the retraining process of ML models triggers only when needed. The user is kept informed about the ML model's performance and the changes in the input and output data to be able to make the right decisions about initiating the retraining process. These are the main advantages of the proposed framework, which significantly

enhances the data scalability and model scalability of predictive ML models. This case study demonstrated applicability and practicality of the proposed framework in leveraging cloud computing to develop scalable ML models for predicting energy consumption of buildings from DT data.

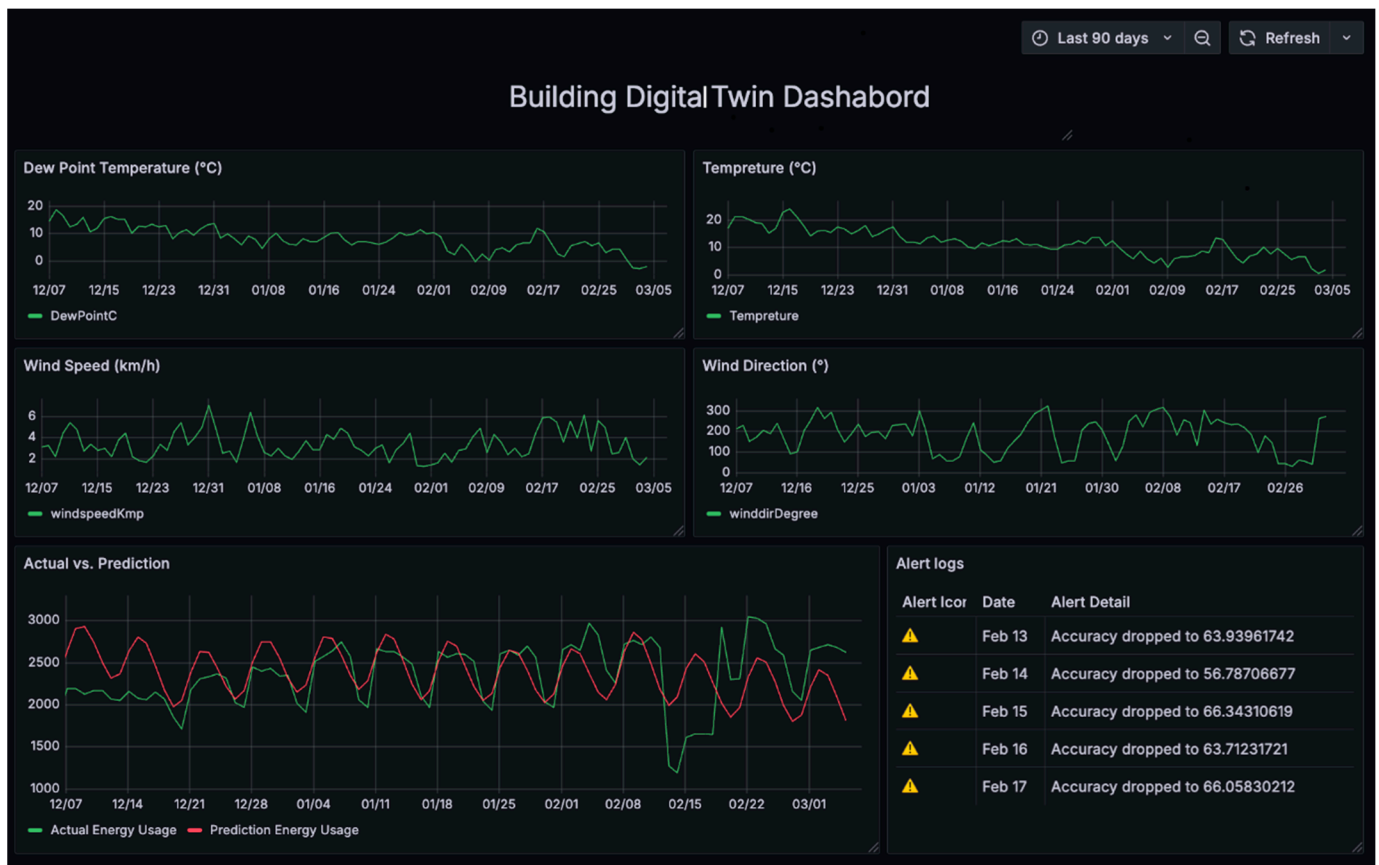


Figure 5. A screenshot of the Digital Twin Dashboard for the case study.

7. Discussion

This study developed a cloud-based framework that could address both data scalability and model scalability. This section evaluates the proposed framework by comparing it with other existing studies.

One of the key characteristics of DT is the dynamic flow of data and providing feedback and insight as new data are generated and processed using AI/ML. As discussed in the Introduction Section, handling a large volume of DT data (data scalability) and adapting ML to changes in data patterns (model scalability) are the underlying issues of most existing studies attempting to use DT data for enhancing building energy performance. The proposed framework addresses the data scalability by leveraging cloud computing and addresses the model scalability by devising a novel monitoring and maintenance module to address the model. Notably, by integrating multiple services from a single cloud platform (i.e., Microsoft Azure), all the data processing and ML model training, retraining, and execution are streamlined and handled seamlessly.

As per the discussion presented in the Literature Review Section, only [6] has attempted to address both data scalability and model scalability. Therefore, ref. [6] is taken as a basis for comparison with this study. Ref. [6] used Digital Twin data for the predictive monitoring of CO₂ equivalent from existing buildings and contributed significantly to the body of knowledge and practice by developing and implementing the Digital Twin of a

building in the real world. The key features of [6] are analyzed in comparison to those of this study, as shown in Table 2.

Table 2. A comparison between [6] and this study.

Feature	[6]	This Study
Purpose	DT development and IoT installation and integration with DT	Developing scalable ML models predicting building energy consumption from DT data
Data pipeline to transfer DT data to ML models	Yes, available	Yes, available
Prediction module	Yes, predicting carbon emission	Yes, predicting energy consumption
Single cloud platform infrastructure for integration of DT	Yes, Azure was used	Yes, Azure was used
Alert System	Yes, alerts are given when sensor data are not within an acceptable range	Yes, alerts are given when accuracy of ML models declines
Drift/anomaly detection	Not available	Yes, available
Continuous monitoring of ML model's performance	Not available	Yes, available
Retraining ML models	Yes, retraining every 15 days	Yes, retraining when needed
User intervention in retraining ML models	No	Yes
Number of retraining in 90 days of the case study	6	0

As seen in Table 2, the purpose of the studies are different: ref. [6] focused on IoT installation and integration with DT for predicting carbon emission, and this study focuses on developing scalable ML models for predicting energy consumption from DT data. However, both studies developed a cloud-based solution for creating data pipelines and transferring DT data to ML models using Microsoft Azure. While both studies have designed modules for retraining ML models and giving alerts, their processes and applications are different. Retraining ML models in [6] is undertaken every 15 days, and the alert system is only for detecting anomalies in sensor data. The lack of continuous monitoring of the ML model's performance and retraining ML models in a fixed interval as proposed by [6] can cause underperformance or inefficiency of ML models. Underperformance of ML models occurs when the model performance declines as no alert is created, and ML models may not be retrained in time. Therefore, in such situations, the ML models are retrained later than when needed. Inefficiency of ML models occurs when the performance stays at an acceptable level and retraining the models at a set interval causes extra computational time and cost for unnecessarily retraining the models. For instance, in this case study, the model is retrained 6 times over 90 days while no training was required, as per the proposed framework. This study has addressed this drawback by incorporating the monitoring and maintenance module in the workflow. This module undertakes an important task of detecting the degradation of ML models by identifying drifts or anomalies. When the accuracy of ML models declines below an acceptable level, an alert is generated to notify the user about it, and the ML model is retrained only when needed. Selecting a suitable performance measure for monitoring and setting a suitable threshold for triggering an alert are important factors influencing the effectiveness of the monitoring and maintenance module. As mentioned in Section 5.3, the accuracy, which is defined by Formula (1), is tracked in this study for evaluating the performance of the ML models. Figure 6 shows the values of the accuracy in the case study. MAPE, which is defined by Formula (2), is another performance measure that has been widely used to evaluate the performance of ML models. The values of MAPE in the case study are shown in Figure 7. In the case that

y_{true} is larger than y_{pred} , the defined accuracy equals $1-MAPE$. In this case, $MAPE > 30\%$ represents Accuracy $< 70\%$. However, when y_{true} is smaller than y_{pred} , this relationship does not exist, and $1-MAPE$ returns smaller values than the defined accuracy. If MAPE is selected for evaluating the ML model's performance, and the threshold sets for $MAPE > 30\%$, the alert is triggered for days 3, 4, 10, 11, and 75 while the accuracy is below 70% on those days. Therefore, normalizing the error in the defined accuracy reduces the number of alerts compared to MAPE in this case study. Further investigation could be undertaken to experiment with different ML model performance measures and assess their impact on the effectiveness of the monitoring and maintenance module.

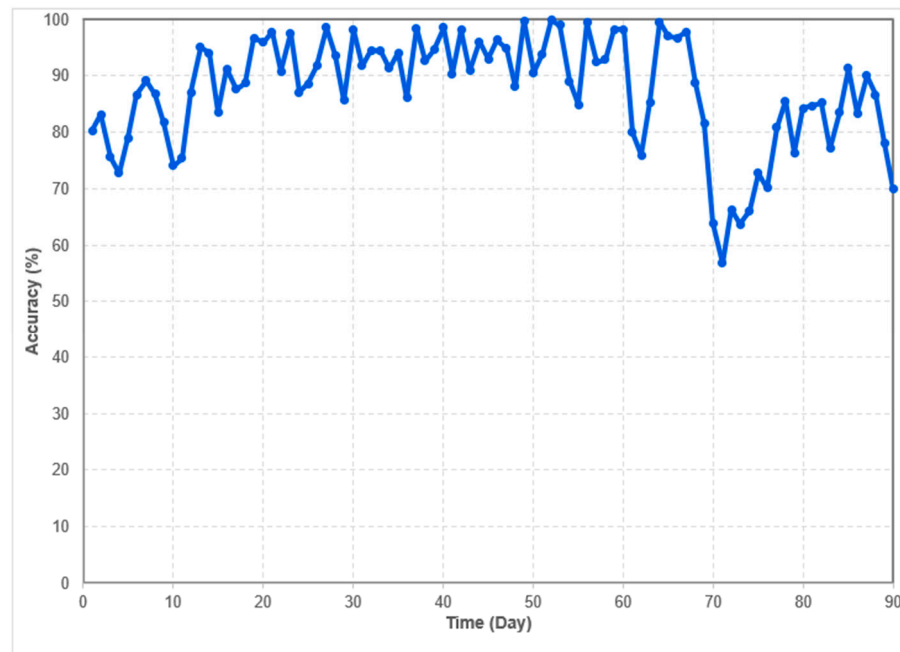


Figure 6. Accuracy of energy consumption prediction in case study over 90 days.

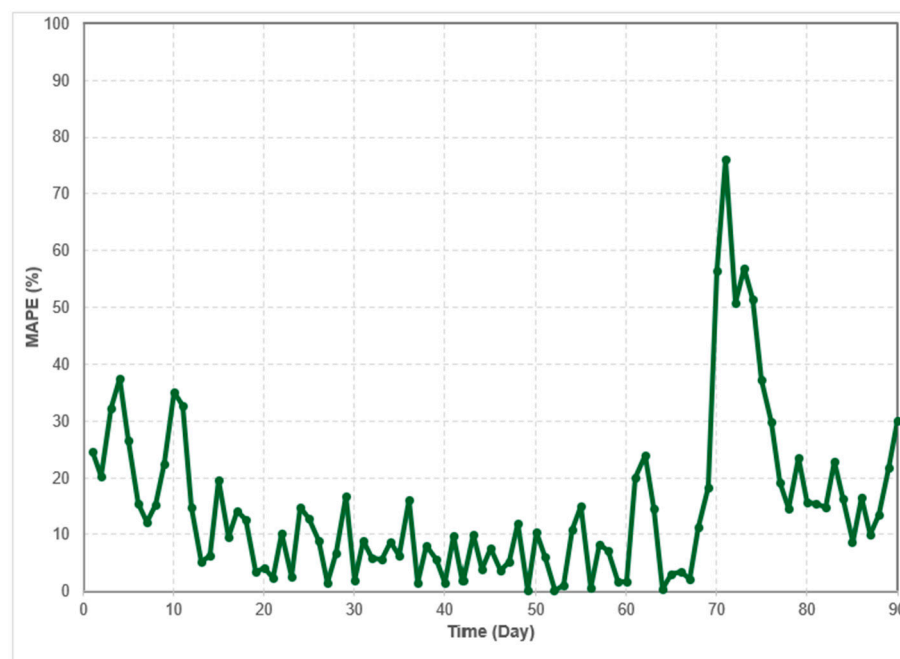


Figure 7. MAPE of energy consumption prediction in case study over 90 days.

In addition, setting a suitable threshold can impact the effectiveness of the monitoring and maintenance module. In this case study, 70% was set as a threshold for accuracy and five alerts were triggered on days 70 to 74. If the threshold is set at 75%, additional alerts are triggered on days 4, 10, 75, 76, and 90. In this study, the threshold was selected subjectively based on the accuracy in the training dataset. Further statistical analysis is required to find a suitable threshold for triggering alerts.

In the process of decision making on model retraining, the user intervention is required, which can enhance reliability of the decisions. Some examples of root causes of drift or anomaly include the following:

- Significant changes to the building occupant behavior affecting energy efficiency of the buildings (e.g., significant changes to the number of occupants, or building shutdown due to major maintenance) that affect energy consumption.
- Malfunction of IoT devices that affect reading their data.
- Significant changes to the physics of the buildings that can affect the energy efficiency of the buildings and ultimately energy consumption.

The case study is an example of significant change to building occupant behavior. This example illustrates the advantage of continuously monitoring ML models and providing timely feedback to the user about any drift and anomaly. The devised continuous monitoring and feedback loop in the proposed framework can help detect any drift or anomaly in a timely manner, and the ML models are adapted to the changes while the unnecessary retraining of ML models is prevented. Ultimately, the proposed framework can enhance the reliability of the data-driven models, in terms of accuracy, and promote their applications in predicting building energy consumption. More accurate prediction of energy consumption can help practitioners make timely adjustments to their building operation strategies, leading to potential building energy savings and more thermal comfort for building occupants.

8. Conclusions

This study proposed a framework for deploying cloud computing to develop scalable ML models for building energy performance prediction from DT data. The dynamic nature of building energy and increasing volumes of DT data impose challenges for model scalability and data scalability of the ML models, respectively. To address these challenges, DSR method was used to develop the framework, and Microsoft Azure was used as a cloud platform integrating several modules proposed in this framework. The proposed framework was tested and evaluated in a case study to demonstrate its practicality and applicability. The result of the case study showed that the devised monitoring and maintenance module in the framework plays a pivotal role in providing continuous feedback about model performance and addressing model scalability. This feedback and retraining the models can enhance the reliability of the data-driven models by maintaining the accuracy of the models above an acceptable threshold. The alerts generated for detected drift or anomaly and the real-time data visualized on the DT dashboard enable the user to make informed decisions about retraining ML models. In the case study, it was observed that over 90 days of monitoring the ML model's performance, the accuracy declined below the set threshold on a few days, and our model could timely detect data drift. However, retaining of the ML model did not trigger as it was recognized that the data drift occurred due to a temporary change to the occupant behavior. Therefore, there was no need for retraining the ML model, and the proposed framework could help avoid unnecessary retraining of the model and the associated computational costs. In addition, streamlining data processing by integrating multiple services on a single cloud platform could enhance the data scalability of the ML models handling an increasing amount of data.

The main contribution of this research is addressing both data scalability and model scalability that will help researchers in the following ways:

- Handling an increasing volume of DT data for predicting the energy consumption of buildings by leveraging cloud computing and developing a seamless data pipeline on a single cloud platform, and
- Adapting ML models to evolving patterns and changes in data only when needed (not too early, which increases the computational time and cost, and not too late, which causes degradation of the ML models) by continuously monitoring the performance of the models.

The proposed framework can advance the adoption of cloud-based approaches in creating more scalable ML models using the DT data of buildings, particularly for predicting the energy consumption of buildings. This can further contribute to broader decarbonization strategies by enabling scalable monitoring of energy use.

The study was undertaken with some assumptions and limitations that can be addressed in future research as follows:

- The proposed framework was tested and validated for a case study of a commercial building in London. Further tests and validations for other building types (e.g., residential buildings) on different climate conditions are required to generalize the framework.
- The main assumption of this study is that the DT of buildings is already in place because this study focuses on enhancing the scalability of ML models. Therefore, the development of DT (e.g., installing IoT devices, connecting them to the network, etc.) was not within the scope of this study. Future work can extend the proposed framework to include DT development in the processes. In addition, the integration of the proposed framework into building management systems (BMS) can be studied in future to further promote its application in practice.
- In this study, Microsoft Azure as one of the most common cloud platforms was utilized since it has the required services for cloud-based Digital Twin and machine learning such as Azure Digital Twins and Azure ML. Amazon AWS also has similar services that could be utilized and tested for comparison in future studies. For instance, AWS IoT TwinMaker, Amazon S3, Amazon SageMaker, SageMaker Registry, and SageMaker Endpoint (or AWS Fargate) have similar functionality as Azure Digital Twins, Azure Databricks, Azure ML, Azure Registry, and ACI, respectively. Following the conceptual model presented in Figure 1 and adjusting the proposed framework to replace Azure services by equivalent AWS services could help future research to develop a similar framework with AWS.
- Creating an initial ML model requires enough historical DT data that may not be available in some cases. Future research could investigate the possibility of deploying advanced approaches for developing ML models even with limited DT data.
- In the Discussion Section, different scenarios that can cause data drift in ML models were described. In this study, only one scenario, i.e., changes in occupant behavior, was considered. Future studies can experiment with other drift scenarios to demonstrate the effectiveness of the proposed framework in various operational conditions.
- This study used a fixed accuracy threshold for detecting drift. This method needs further investigation to avoid subjectivity in setting the threshold. Also, other drift detection methods, as described in Section 4.4, can be tested in future research to evaluate their effectiveness in the proposed framework.
- For developing ML models in this study, Bi-LSTM was used as one of the common algorithms for predicting the energy performance of buildings. While the proposed framework is independent of the ML algorithm's type, other algorithms could be tested, and their performances are evaluated in future studies.

Author Contributions: Conceptualization, E.M.; methodology, E.M., A.S. and M.W.; software, E.M.; validation, E.M.; formal analysis, E.M.; investigation, E.M.; data curation, E.M.; writing—original draft preparation, E.M.; writing—review and editing, A.S. and M.W.; visualization, E.M.; supervision, A.S. and M.W.; project administration, A.S.; funding acquisition, M.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the European Regional Development Fund, grant number (25R17P01847).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

DT	Digital Twin
ML	Machine learning
ACI	Azure Container Instances

References

1. Mirabella, N.; Röck, M.; Ruschi Mendes SAADE, M.; Spirinckx, C.; Bosmans, M.; Allacker, K.; Passer, A. Strategies to improve the energy performance of buildings: A review of their life cycle impact. *Buildings* **2018**, *8*, 105. [\[CrossRef\]](#)
2. House of Commons Environmental Audit Committee. *Building to Net Zero: Costing Carbon in Construction*; House of Commons Environmental Audit Committee: London, UK, 2022.
3. UK Government. *Net Zero Strategy: Build Back Greener*; HM Government: Westminster, UK, 2021.
4. Tahmasebinia, F.; Lin, L.; Wu, S.; Kang, Y.; Sepasgozar, S. Exploring the benefits and limitations of digital twin technology in building energy. *Appl. Sci.* **2023**, *13*, 8814. [\[CrossRef\]](#)
5. Grieves, M. Digital twin: Manufacturing excellence through virtual factory replication. *White Pap.* **2014**, *1*, 1–7.
6. Arsiwala, A.; Elghaish, F.; Zohar, M. Digital twin with Machine learning for predictive monitoring of CO₂ equivalent from existing buildings. *Energy Build.* **2023**, *284*, 112851. [\[CrossRef\]](#)
7. Lu, Q.; Xie, X.; Parlikad, A.K.; Schooling, J.M.; Konstantinou, E. Moving from building information models to digital twins for operation and maintenance. *Proc. Inst. Civ. Eng.-Smart Infrastruct. Constr.* **2020**, *174*, 46–56. [\[CrossRef\]](#)
8. Henzel, J.; Wróbel, Ł.; Fice, M.; Sikora, M. Energy consumption forecasting for the digital-twin model of the building. *Energies* **2022**, *15*, 4318. [\[CrossRef\]](#)
9. Fujii, T.Y.; Hayashi, V.T.; Arakaki, R.; Ruggiero, W.V.; Bulla Jr, R.; Hayashi, F.H.; Khalil, K.A. A digital twin architecture model applied with MLOps techniques to improve short-term energy consumption prediction. *Machines* **2021**, *10*, 23. [\[CrossRef\]](#)
10. Fathy, Y.; Jaber, M.; Nadeem, Z. Digital twin-driven decision making and planning for energy consumption. *J. Sens. Actuator Netw.* **2021**, *10*, 37. [\[CrossRef\]](#)
11. Mishra, A. *Scalable AI and Design Patterns: Design, Develop, and Deploy Scalable AI Solutions*; Apress: Berkeley, CA, USA, 2024.
12. Gourisetti, S.N.G.; Bhadra, S.; Sebastian-Cardenas, D.J.; Touhiduzzaman, M.; Ahmed, O. A theoretical open architecture framework and technology stack for digital twins in energy sector applications. *Energies* **2023**, *16*, 4853. [\[CrossRef\]](#)
13. Zhao, J.; Feng, H.; Chen, Q.; de Soto, B.G. Developing a conceptual framework for the application of digital twin technologies to revamp building operation and maintenance processes. *J. Build. Eng.* **2022**, *49*, 104028. [\[CrossRef\]](#)
14. Amini, H.; Alanne, K.; Kosonen, R. Building simulation in adaptive training of machine learning models. *Autom. Constr.* **2024**, *165*, 105564. [\[CrossRef\]](#)
15. Yang, S.; Wan, M.P.; Chen, W.; Ng, B.F.; Dubey, S. Model predictive control with adaptive machine-learning-based model for building energy efficiency and comfort optimization. *Appl. Energy* **2020**, *271*, 115147. [\[CrossRef\]](#)
16. Wang, L.; Kubichek, R.; Zhou, X. Adaptive learning based data-driven models for predicting hourly building energy use. *Energy Build.* **2018**, *159*, 454–461. [\[CrossRef\]](#)
17. Hosseinloo, A.H.; Ryzhov, A.; Bisch, A.; Ouerdane, H.; Turitsyn, K.; Dahleh, M.A. Data-driven control of micro-climate in buildings: An event-triggered reinforcement learning approach. *Appl. Energy* **2020**, *277*, 115451. [\[CrossRef\]](#)

18. Zhao, L.; Zhang, H.; Wang, Q.; Wang, H. Digital-Twin-Based Evaluation of Nearly Zero-Energy Building for Existing Buildings Based on Scan-to-BIM. *Adv. Civ. Eng.* **2021**, 2021, 6638897. [CrossRef]
19. Afzal, S.; Shokri, A.; Ziapour, B.M.; Shakibi, H.; Sobhani, B. Building energy consumption prediction and optimization using different neural network-assisted models; comparison of different networks and optimization algorithms. *Eng. Appl. Artif. Intell.* **2024**, 127, 107356. [CrossRef]
20. Saadatifar, S.; Zhang, Y. Balancing thermal comfort with energy consumption in buildings using digital twins, IoT sensors, and real-time dashboards to inform occupant decision making. *ASHRAE Trans.* **2023**, 129, 720–729.
21. Spudys, P.; Afxentiou, N.; Georgali, P.-Z.; Klumbyte, E.; Jurelionis, A.; Fokaides, P. Classifying the operational energy performance of buildings with the use of digital twins. *Energy Build.* **2023**, 290, 113106. [CrossRef]
22. Manfren, M.; James, P.A.; Aragon, V.; Tronchin, L. Lean and interpretable digital twins for building energy monitoring—A case study with Smart Thermostatic Radiator Valves and Gas Absorption Heat Pumps. *Energy AI* **2023**, 14, 100304. [CrossRef]
23. Both, M.; Kämper, B.; Cartus, A.; Beermann, J.; Fessler, T.; Müller, J.; Diedrich, C. Automated monitoring applications for existing buildings through natural language processing based semantic mapping of operational data and creation of digital twins. *Energy Build.* **2023**, 300, 113635. [CrossRef]
24. Hu, Z.; Haghsheenas, A.; Hasan, A.; Sørensen, S.; Karlsen, A.; Alaliyat, S. Interoperable digital twin solutions for asset-heavy industry. In *Digital Twin Driven Intelligent Systems and Emerging Metaverse*; Springer: Singapore, 2023; pp. 195–208.
25. Ohueri, C.C.; Masrom, M.A.N.; Seghier, T.E. Digital twin for decarbonizing operating buildings: A systematic review and implementation framework development. *Energy Build.* **2024**, 320, 114567. [CrossRef]
26. ElArwady, Z.; Kandil, A.; Afify, M.; Marzouk, M. Modeling Indoor Thermal Comfort in Buildings using Digital Twin and Machine Learning. *Dev. Built Environ.* **2024**, 19, 100480. [CrossRef]
27. Pavón, R.M.; Alberti, M.G.; Álvarez, A.A.A.; Cepa, J.J. Bim-based Digital Twin development for university Campus management. Case study ETSICCP. *Expert Syst. Appl.* **2025**, 262, 125696. [CrossRef]
28. Li, X.; Zhang, F.-L.; Xiang, W.; Liu, W.-X.; Fu, S.-J. Structural health monitoring system based on digital twins and real-time data-driven methods. *Structures* **2024**, 70, 107739. [CrossRef]
29. Zaballo, A.; Briones, A.; Massa, A.; Centelles, P.; Caballero, V. A smart campus' digital twin for sustainable comfort monitoring. *Sustainability* **2020**, 12, 9196. [CrossRef]
30. Knebel, F.P.; Trevisan, R.; do Nascimento, G.S.; Abel, M.; Wickboldt, J.A. A study on cloud and edge computing for the implementation of digital twins in the Oil & Gas industries. *Comput. Ind. Eng.* **2023**, 182, 109363.
31. Ni, Z.; Liu, Y.; Karlsson, M.; Gong, S. A sensing system based on public cloud to monitor indoor environment of historic buildings. *Sensors* **2021**, 21, 5266. [CrossRef] [PubMed]
32. Hadjidemetriou, L.; Stylianidis, N.; Englezos, D.; Papadopoulos, P.; Eliades, D.; Timotheou, S.; Polycarpou, M.M.; Panayiotou, C. A digital twin architecture for real-time and offline high granularity analysis in smart buildings. *Sustain. Cities Soc.* **2023**, 98, 104795. [CrossRef]
33. Tan, Y.; Chen, P.; Shou, W.; Sadick, A.-M. Digital Twin-driven approach to improving energy efficiency of indoor lighting based on computer vision and dynamic BIM. *Energy Build.* **2022**, 270, 112271. [CrossRef]
34. Vom Brocke, J.; Hevner, A.; Maedche, A. Introduction to design science research. In *Design Science Research. Cases*; Springer: Cham, Switzerland, 2020; pp. 1–13.
35. Peffers, K.; Tuunanen, T.; Rothenberger, M.A.; Chatterjee, S. A design science research methodology for information systems research. *J. Manag. Inf. Syst.* **2007**, 24, 45–77. [CrossRef]
36. Available online: <https://learn.microsoft.com/en-us/azure/digital-twins/overview> (accessed on 13 January 2025).
37. Available online: <https://learn.microsoft.com/en-us/azure/iot-hub/iot-concepts-and-iot-hub> (accessed on 13 January 2025).
38. Available online: <https://learn.microsoft.com/en-us/azure/machine-learning/overview-what-is-azure-machine-learning?view=azureml-api-2> (accessed on 13 January 2025).
39. Available online: <https://learn.microsoft.com/en-us/azure/digital-twins/concepts-apis-sdks> (accessed on 10 February 2025).
40. Available online: <https://learn.microsoft.com/en-us/azure/databricks/introduction/> (accessed on 10 February 2025).
41. Available online: <https://learn.microsoft.com/en-us/azure/machine-learning/concept-automated-ml?view=azureml-api-2> (accessed on 13 January 2025).
42. Available online: <https://learn.microsoft.com/en-us/azure/machine-learning/how-to-manage-models?view=azureml-api-2&tabs=python> (accessed on 13 January 2025).
43. Available online: <https://learn.microsoft.com/en-us/azure/container-instances/container-instances-overview> (accessed on 13 January 2025).
44. Available online: <https://learn.microsoft.com/en-us/azure/machine-learning/concept-mlflow-models?view=azureml-api-2> (accessed on 13 January 2025).
45. Available online: <https://techcommunity.microsoft.com/blog/fasttrackforazureblog/identifying-drift-in-ml-models-best-practices-for-generating-consistent-reliable/4040531> (accessed on 13 January 2025).

46. Samarajeewa, C.; De Silva, D.; Manic, M.; Mills, N.; Moraliyage, H.; Alahakoon, D.; Jennings, A. An artificial intelligence framework for explainable drift detection in energy forecasting. *Energy AI* **2024**, *17*, 100403. [CrossRef]
47. Li, A.; Zhang, C.; Xiao, F.; Fan, C.; Deng, Y.; Wang, D. Large-scale comparison and demonstration of continual learning for adaptive data-driven building energy prediction. *Appl. Energy* **2023**, *347*, 121481. [CrossRef]
48. Lu, J.; Liu, A.; Dong, F.; Gu, F.; Gama, J.; Zhang, G. Learning under concept drift: A review. *IEEE Trans. Knowl. Data Eng.* **2018**, *31*, 2346–2363. [CrossRef]
49. Available online: <https://learn.microsoft.com/en-us/azure/machine-learning/how-to-monitor-datasets?view=azureml-api-1&tabs=python> (accessed on 15 March 2025).
50. Yuan, Y.C. *Multiple Imputation for Missing Data: Concepts and New Development*, 9th ed.; SAS Institute Inc.: Rockville, MD, USA, 2010; Volume 49, p. 12.
51. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
52. Li, G.; Zhao, X.; Fan, C.; Fang, X.; Li, F.; Wu, Y. Assessment of long short-term memory and its modifications for enhanced short-term building energy predictions. *J. Build. Eng.* **2021**, *43*, 103182. [CrossRef]
53. Sülo, I.; Keskin, S.R.; Dogan, G.; Brown, T. Energy efficient smart buildings: LSTM neural networks for time series prediction. In Proceedings of the 2019 International Conference on Deep Learning and Machine Learning in Emerging Applications (Deep-ML), Istanbul, Turkey, 26–28 August 2019; pp. 18–22.
54. Durand, D.; Aguilar, J.; R-Moreno, M.D. An analysis of the energy consumption forecasting problem in smart buildings using LSTM. *Sustainability* **2022**, *14*, 13358. [CrossRef]
55. Ullah, F.U.M.; Ullah, A.; Haq, I.U.; Rho, S.; Baik, S.W. Short-term prediction of residential power energy consumption via CNN and multi-layer bi-directional LSTM networks. *IEEE Access* **2019**, *8*, 123369–123380. [CrossRef]
56. Ma, J.; Cheng, J.C.; Jiang, F.; Chen, W.; Wang, M.; Zhai, C. A bi-directional missing data imputation scheme based on LSTM and transfer learning for building energy data. *Energy Build.* **2020**, *216*, 109941. [CrossRef]
57. Zhang, C.; Ma, L.; Luo, Z.; Han, X.; Zhao, T. Forecasting building plug load electricity consumption employing occupant-building interaction input features and bidirectional LSTM with improved swarm intelligent algorithms. *Energy* **2024**, *288*, 129651. [CrossRef]
58. Atef, S.; Eltawil, A.B. Assessment of stacked unidirectional and bidirectional long short-term memory networks for electricity load forecasting. *Electr. Power Syst. Res.* **2020**, *187*, 106489. [CrossRef]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.