SCALABLE CLOUD-BASED TRANSFER LEARNING FOR BUILDING ENERGY PREDICTION FROM LIMITED DIGITAL TWIN DATA

SUBMITTED: May 2025
PUBLISHED: February 2026
EDITOR: Žiga Turk
DOI: 10.36680/j.itcon.2026.005

Elham Mahamedi, PhD
Department of Architecture and Built Environment, Faculty of Engineering and Environment, Northumbria University, Newcastle upon Tyne, UK
e.mahamedi@northumbria.ac.uk

Alaeldin Suliman, Assistant Professor
Department of Architecture and Built Environment, Faculty of Engineering and Environment, Northumbria University, Newcastle upon Tyne, UK
Department of Civil engineering, University of New Brunswick, Fredericton, NB, E3B 5A3, Canada
ala.suliman@northumbria.ac.uk

Martin Wonders, Assistant Professor
Department of Computer and Information Sciences, Faculty of Engineering and Environment, Northumbria University, Newcastle upon Tyne, UK
martin.wonders@northumbria.ac.uk

SUMMARY: *Data scarcity is one of the major barriers to using Machine Learning (ML) techniques for building energy predictions. Transfer Learning (TL) has proven to be an effective approach to address this issue. However, most TL approaches have not considered scalability, which is an important factor when using Digital Twin (DT) data for predictions. To bridge this gap, this research aims to develop a scalable TL approach for predicting building energy consumption from limited DT data using cloud computing. Two frameworks, using Microsoft Azure and Amazon AWS, are developed and implemented in a case study to demonstrate their applicability and practicality. The findings show that both frameworks possess unique strengths in terms of scalability and improving accuracy. Azure performed better in training time and resource efficiency, while AWS excelled in data preprocessing speed and prediction accuracy. This paper supports researchers and practitioners in developing a scalable model (i.e., cloud-based TL) for predicting building energy consumption with limited DT data. Future research can explore the cost of using cloud platforms for building energy prediction and expand the list of decision factors for platform selection.*

KEYWORDS: *machine learning, transfer learning, digital twin, cloud computing, building energy.*

REFERENCE: *Mahamedi, E., Suliman, A., & Wonders, M. (2026). Scalable cloud-based transfer learning for building energy prediction from limited digital twin data. Journal of Information Technology in Construction (ITcon), 31, 106-128. https://doi.org/10.36680/j.itcon.2026.005*

1. INTRODUCTION

The built environment accounts for approximately 37% of global energy-related CO_2 emissions (Awan et al., 2025), and in Europe, buildings consumed 40% of energy according to data collected in 2023 (Energy Performance of Buildings Directive, 2025). Hence, improving building energy performance can significantly contribute to saving energy and advancing decarbonisation efforts in this sector. Recent advancements in computer science and digital technologies have enabled ML to improve building energy performance across various applications, including the prediction of building energy consumption (e.g., Olu-Ajayi et al. (2022), Pham et al. (2020)), the automation of building operation (e.g., Kawa and Borkowski (2023), Elnour et al. (2022)), the reduction of carbon emissions and environmental impacts (e.g., Zheng et al. (2024), Kapoor et al. (2022)), and the assessment of occupant comfort and behaviour (e.g., Amasyali and El-Gohary (2021), Zhang et al. (2024)).

Predicting building energy consumption using ML can offer invaluable insights for informed decisions regarding building designs, materials, and technologies (Hussien et al., 2025). However, creating precise ML models requires an adequate amount of data imposing constraints on deploying these models for buildings with no or limited historical data (Elnour et al., 2024). Pinto et al. (2022) emphasised that most buildings lack appropriate sensing technologies or lack the IT infrastructure to collect and store data, which makes collecting and preparing quality data to train ML models a time-consuming process.

DT can contribute to addressing the technological gap of data collection by deploying reality capture technologies to provide real-time data from built assets. The idea of DT was first introduced by Grieves in 2003 (Grieves, 2014) and the term itself was coined by NASA technologist John Vickers in 2010 (NASA, 2025). Grieves (2014) defined DT as a virtual representation of a physical product, which can improve building energy management by monitoring, optimising, and predicting building energy performance (Kawa and Borkowski, 2023). Some studies (e.g., Han et al. (2024), Jradi et al. (2023)) used DT data and ML for predicting building energy consumption. However, DT is recently being adopted in industry, and in many cases, the amount of DT data available for creating ML models is not sufficient to achieve acceptable accuracy (Tahmasebinia et al., 2023). Therefore, the adoption of DT for developing ML models is not feasible for a new building or a building with a new DT system.

Recently, TL has attracted the attention of researchers to bridge the gap of data scarcity for creating ML models for building energy. TL is able to transfer ML models that are trained and validated for buildings with rich data to buildings with limited or poor data (Pinto et al., 2022). TL has been shown by several studies (e.g., Li et al. (2021), Li et al. (2024), Xing et al. (2024)) as a promising method for predicting energy performance of buildings when limited quality data is available. However, these studies have focused only on investigating the effectiveness of TL in different scenarios, or advancing the performance of TL methods, and overlooked the scalability of their models. Here, scalability refers to two aspects: i) data scalability, which means the ability of a system to handle an increasing data quantity and ii) model scalability, which means the ability of ML models to adapt to changes in data patterns and address new use cases (Mishra, 2024). Data scalability is a critical factor when DT data is used for prediction because an increasing amount of data should be handled efficiently in the workflow of the prediction process. Model scalability is also important due to the dynamic nature of building energy performance, and the need for adapting the models to changing conditions such as occupant behaviour and weather conditions that can affect energy consumption (De Simone, 2022; Mahamedi et al., 2025).

To address the scalability of ML models, cloud computing is a suitable solution as it can provide high computing power and data storage capacity enabling scaling up models easily (Digitemie & Ekemezie, 2024). Cloud computing has been adopted in some studies (e.g., Moudgil et al. (2024), Ni et al. (2021)) for enhancing building energy performance. However, there is little research on how to use cloud computing for developing TL models predicting energy consumption.

To address this gap, this research aims to develop a cloud-based approach to enhance the scalability of TL models for predicting building energy consumption from limited DT data. To this end, two frameworks using Microsoft Azure and Amazon AWS as two leading cloud platforms for ML are developed and tested in a case study for demonstrating their applicability and evaluating their performance.

The rest of this paper is organised as follows. Section 2 presents the a explanation of the TL principles and related work on the applications of TL in predicting building energy. Section 3 outlines the research methodology. In Section 4, the proposed frameworks are described. Section 5 demonstrates the implementation of the proposed frameworks in a case study. The next section presents the results, followed by a discussion of the case study findings. Finally, the last section presents the conclusion of the paper.

2. LITERATURE REVIEW

As highlighted in the introduction section, the scarcity or lack of quality historical data for training predictive ML models poses a significant to adopting ML for building energy predictions. In addition, preparing and processing data for ML is a time consuming and costly endeavour. TL can reduce the reliance of conventional ML techniques on historical data, and speed up the training process by allowing cross-domain data utilisation (Chen et al., 2020). However, many existing studies have overlooked the scalability of TL models. This section first introduces the principles of TL from the literature. Then, a review of TL applications in enhancing building energy performance, and the identified the research gap are presented.

2.1 Principles of Transfer Learning

In conventional ML, one of the major assumptions is that the data for training and prediction must be in the same feature space and have the same distribution while in some cases, the prediction task is in one domain of interest, but sufficient training data is available in another domain of interest (Panigrahi et al., 2021). In such cases, TL can help to transfer and apply the prior knowledge from one domain and task to another domain and task (Gao et al., 2020). That is, in TL, the data for training ML models can come from a different domain. There are some terms related to TL that are described below according to Pinto et al. (2022).

- Domain (D), which forms the input space of the ML model, consists of two components: (i) the feature space X, and (ii) the marginal probability distribution P(X) where X= {x1 , …, xn} ∈ X. The feature space is the space of all influencing variables (input features) and xi represents the ith influencing variable. The marginal probability distribution represents the probability distribution of each input feature to be contained in the feature space.
- Task (T), which forms the output space of the model, consists of two components: (i) a label space Y and (ii) an objective predictive function $f(\cdot)$, denoted as T={Y, $f(\cdot)$}. f(.) is not observed but learned from the training data, represented as {xi , yi } , where xi ∈ X and yi ∈ Y. f(.) can be used to approximate the conditional probability P(y|x) and predict the corresponding label of a new instance x.
- Transfer Learning is a process of improving the learning of the target predictive function f(.) in Target Domain (DT) and Target Task (TT) by using the knowledge in Source Domain (DS) and Source Task (TS), where DS≠DT, or TS≠TT.

In the context of the building energy prediction models, the target building is the one with limited data in the Target Domain (D_T), and we want to predict its energy consumption. The source building is the one with sufficient historical data in its domain (D_S), and we want to use it to improve the prediction accuracy for the target building. The task of energy prediction for the target building is Target Task (T_T), and the task of energy prediction for the source building is Source Task (T_S). As shown in Figure 1, first, a ML model, called "Base Model" is created from the data of the source building available in the Source Domain (D_S) for performing Source Task (T_S), which is the prediction of the source building's energy consumption. Next, TL principles are used to transfer the knowledge from the source building to the target building and create a "TL Model" from the Base Model for performing Target Task (T_T), which is the prediction of the target building's energy consumption from the data of the target building available in the Target Domain (D_T).

2.2 Transfer Learning applications in Building Energy Predictions

Over the past few years, several studies have applied TL learning to enhance building energy predictions. Some of these studies have attempted to demonstrate the effectiveness of the TL method over the direct learning (i.e., conventional ML methods without TL) in various scenarios. For instance, Fan et al. (2020) performed statistical assessment to investigate the potentials of TL and validate its value for short-term building energy demand predictions. Their experiments demonstrated that the TL-based methodology could reduce approximately 15% to 78% of prediction errors compared with standalone models. In another study, Hooshmand and Sharma (2019) used TL to improve the accuracy of a convolutional neural network (CNN) model for short-term electricity load forecasting (i.e., forecasting the next 24 hours of electricity demand based on the energy consumption of the past four weeks).Through a case study, they demonstrated that the TL model could enhance the accuracy of the CNN model, which was trained only using the target load data without employing a public dataset and pre-training step, by 17%. To consider the effects of seasonality within domains, and predict cross-building energy consumption, Ribeiro et al. (2018) proposed a TL method using time-series multi-feature regression with seasonal and trend adjustments. Their proposed method merged data from similar buildings with different distributions and different seasonal profiles, and improved energy prediction accuracy for building with small historical datasets such as new buildings or those with newly installed meters. In their case study, it was shown that the proposed TL method could improve accuracy of the prediction by %11.2 compared to a direct learning model.

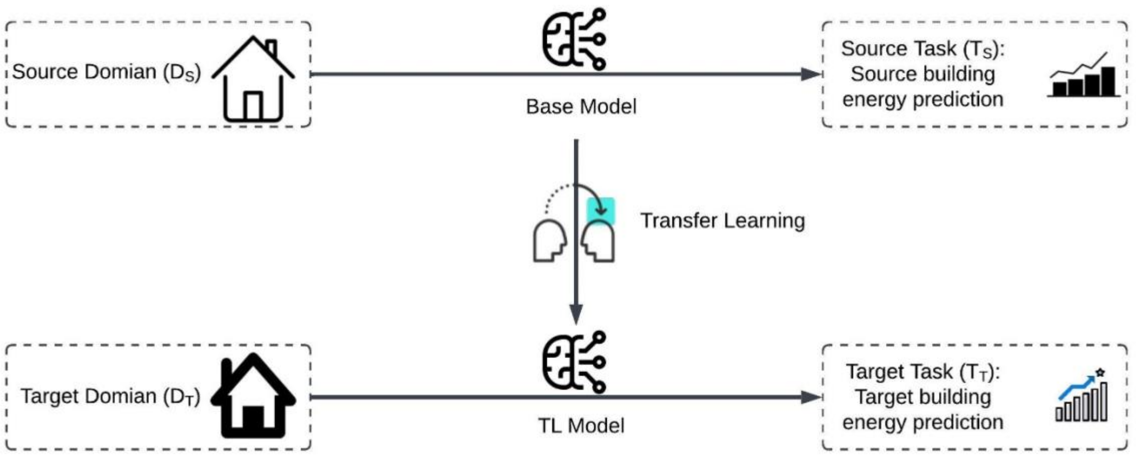
Source Domian (D_S)
Base Model
Source Task (T_S): Source building energy prediction
Transfer Learning
Target Domian (D_T)
TL Model
Target Task (T_T): Target building energy prediction

Figure 1. Overview of the transfer learning technique for predicting building energy.

Some other studies related to TL focused on exploring novel approaches for improving TL models' performance in energy predictions. For instance, one study proposed a novel approach, called Similarity-Based Chained Transfer Learning (SBCTL), for creating neural network-based models for energy forecasting (Tian et al., 2019). In their approach, first, a model is trained in a traditional way, and the following models transfer knowledge from the previous models in a chain-like manner based on similarities between their energy consumption profiles. The objective of this approach was to reduce training time, not to improve accuracy. In an experiment, SBCTL could reduce training time by 97% while achieving accuracy comparable to traditional models. For improving the training process, Wang et al. (2019) pre-trained the models for TL in a supervised way and then incorporated unlabelled test data into the training process in a semi-supervised way by introducing an adaptive layer. This method could shrink the distribution discrepancy from different energy sources and lower the demand of a high amount of labelled data for training purposes. For automating feature extraction and domain adaptation, Fang et al. (2021) proposed a Long Short-Term Memory (LSTM) based feature extractor, and a Domain Adversarial Neural Network (DANN) approach identifying domain invariant features between the source and target domains. In their proposed approach, the trained model with data from source building is applied directly to energy prediction of the Target building without prediction performance degradation caused by domain shifts. In another study, for selecting an optimum source building for TL, Lu et al. (2021) defined a similarity measurement index (SMI), and integrated it into their proposed TL-based framework for predicting thermal load of buildings. They implemented the proposed framework in a number of cases and showed prediction stability in their models while reducing the prediction errors by 0.6%~15.26% compared with direct learning and 1.81%~5.65% compared with TL without the selection of source tasks.

These studies demonstrated how TL can address data scarcity in building energy predictions. However, they considered only static data in their model and overlooked the adaptability of the models (i.e., model scalability), which is an important factor due to the dynamics of building energy and the need for adapting the predictive models to changing conditions that affect the model performance. To the best knowledge of the authors, only one recent study has attempted to address this drawback by proposing a novel reinforcing TL approach, which dynamically incorporates the newly generated building data into the model to improve its accuracy (Mahamedi et al., 2024). Their proposed approach could support the use of real-time DT data for building energy prediction when limited DT data is available.

For building energy prediction using direct learning, some studies have considered the adaptability of ML models. For instance, Yang et al. (2020) developed an adaptive ML model for improving building automation and control applications by updating the ML model every day using online building operation data. In another study, Wang et al. (2018) considered adapting ML models predicting building energy use by incorporating new building data into the models every two weeks. These studies showed that adapting ML models can continuously improve accuracy of the models as more new data is incorporated in the models. However, no study has considered the adaptivity of TL models using dynamic DT data.

Another challenge for developing TL models using DT data is data scalability because the models need to handle an increasing amount of data continuously generated by DT. Cloud computing has been adopted in some recent studies as a robust technology that can address data scalability. For instance, ElArwady et al. (2024) developed a cloud platform to use DT data for ML models predicting indoor thermal comfort in buildings. In another study, cloud computing was used for connecting IoT devices with the DT model and predicting CO_2 emissions of buildings (Arsiwala et al., 2023). Despite the contributions of these studies to develop cloud-enabled ML models, none of them considered TL for prediction. Therefore, this research intends to address the scalability (i.e., both model scalability and data scalability) of TL models for predicting energy consumption of buildings by leveraging cloud computing and DT data.

3. METHODOLOGY

In this research, the chosen research methodology is Design Science Research (DSR) method. DSR is a suitable method for this research because it offers a systematic approach for clearly defining the problem, which is addressing scalability of TL models for building energy prediction from limited DT data, and developing a problem-driven solution that meets the defined objectives of the solution. The solution is this research is a framework that leverages cloud computing to develop scalable TL-based models. DSR can generate knowledge of how things can and should be constructed (Vom Brocke et al., 2020), and encompasses six main steps including: 1) problem identification and motivation, 2) define the objectives for a solution, 3) design and development, 4) demonstration 5) evaluation, and 6) communication (Peffers et al., 2007). If needed, some of these steps can be iterated to refine objectives and/or the design and development. These iterations are very beneficial to this research because developing a scalable cloud-based TL framework requires several iterations for development, testing and refinements. The six steps of DSR are followed in this research as described below:

1. Defining the problem: The problem is how to develop scalable TL models using limited DT data for predicting energy consumption of buildings. The problem was identified through a semi-systematic review and by conducting thematic analysis of the literature (Snyder 2019) on data scarcity as a barrier of building energy predictions using ML, and the use of DT and TL as potential solutions to overcome this barrier. The summary of the literature review findings was presented in the introduction and literature review section of this paper. The selected studies for review were from 2020 to 2024.

2. Defining objectives and requirements of a solution: The main objectives of the solutions are to adopt cloud computing for 1) transmitting building data seamlessly from DT to TL models, and 2) developing TL models, which includes data preparation, model training and execution, and data visualisation on a single cloud platform. The requirements of the solution are i) addressing model scalability to adapt the TL models to the changing conditions by incorporating newly generated DT data into the models, and ii) addressing data scalability for handling an increasing amount of DT data. The objectives and requirements of a solution were identified through the semi-systematic review and thematic analysis, as described in the step 1.

3. Design and development: Design and development of two frameworks using Microsoft Azure and Amazon AWS as common cloud platforms that have the required services and power for handling an increasing amount of IoT data from DT models and developing scalable machine learning models. These two platforms are selected because they have services for DT and ML that enable the development of all required functions for the frameworks such as creating a data pipeline for transferring data, data preparation modules, and ML modules.

4. Demonstration: Conduct a case study to test and validate the practicality and applicability of the developed frameworks in predicting energy consumption of a building from limited DT data in a scalable manner. The case study is a commercial building as a Target building with limited historical data in London in 2021. A similar building with two years of historical data is selected as a Source building. For prediction of the energy consumption, four features were used: (1) temperature, (2) wind direction, (3) wind speed, and (4) dew point temperature.
5. Evaluation: Analyse the results of the case study to assess the frameworks by comparing the performance of used cloud platforms. For the comparative analysis, multiple performance indicators such as Mean Absolute Percentage Error, average data loading time, average CPU usage, average RAM usage, average throughput, training loss, validation loss, and training time were utilized. If needed, the previous steps are repeated to refine the framework and ensure that it can address the identified problem and meet the objectives and requirements.
6. Communication: Publish this paper to share the research outcomes to scholars and practitioners.

4. PROPOSED FRAMEWORKS

To develop the intended framework, first, a conceptual model as shown in Figure 2 is created to identify the key components and specify their functionalities. The conceptual model consists of five modules including:

- M-1: “Source building’s data transfer and data preparation” module and M-2) “Base Model development and training module”, which are for developing a Base Model from the historical data of the Source building,
- M-3: “Target building’s data transfer and data preparation” module, and M-4) “TL model development and fine-tuning” module, which are for developing TL models from the DT data of the Target building, and
- M-5: “TL model deployment and interface” module for deploying the TL model and visualise the data.

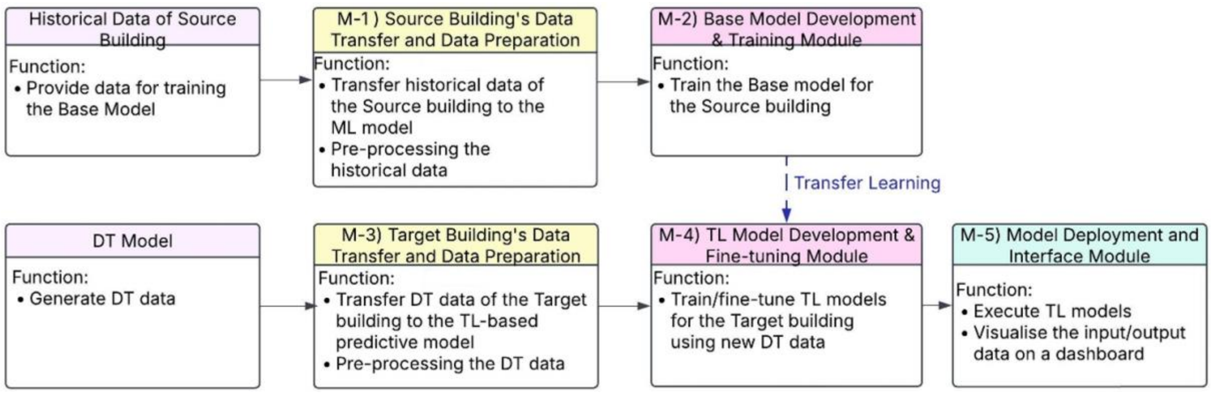
Historical Data of Source Building
Function:
• Provide data for training the Base Model
M-1) Source Building's Data Transfer and Data Preparation
Function:
• Transfer historical data of the Source building to the ML model
• Pre-processing the historical data
M-2) Base Model Development & Training Module
Function:
• Train the Base model for the Source building
Transfer Learning
DT Model
Function:
• Generate DT data
M-3) Target Building's Data Transfer and Data Preparation
Function:
• Transfer DT data of the Target building to the TL-based predictive model
• Pre-processing the DT data
M-4) TL Model Development & Fine-tuning Module
Function:
• Train/fine-tune TL models for the Target building using new DT data
M-5) Model Deployment and Interface Module
Function:
• Execute TL models
• Visualise the input/output data on a dashboard

Figure 2: A Conceptual model for developing the framework.

As discussed in Section 2, the historical data of the Source building is used to develop a Base Model using an ML algorithm. To this end, first, M-1 undertakes the task of transferring the historical data of the Source building to the ML model, and pre-processing the data as a preparation prior to training the Base Model. Then, M-2 develops and train the Base Model using a suitable ML algorithm from the pre-processed historical data of the Source building. The developed and trained Base Model will be used for TL to predict the energy consumption of the Target building.

In this paper, the assumption is that DT of the Target building already exists. Therefore, the focus for the Target building is on transferring DT data to the TL model for prediction. M-3 undertakes the task of transferring and pre-processing the DT data to prepare it for the TL model. Then, the processed data is used in M -4 for training the TL model. The TL model is also fine-tuned in this module using newly generated DT data. This function addresses the model scalability requirement of the framework by adapting the TL model to the changing conditions of the Target building.

Finally, M-5 executes the TL models and visualises the model input and output data on a dashboard. Developing all these modules on a single cloud platform provides a seamless data pipeline transmitting the data to predictive models, training the models, and visualising the data, which addresses the requirement of data scalability to handle an increasing amount of DT data. Based on the conceptual model, two frameworks are developed on Microsoft Azure and Amazon AWS. The proposed frameworks are described in the following sections.

4.1 Proposed Azure-based Framework

The Azure-based framework is shown in Figure 3. The framework has five modules as per the conceptual framework. The description of each module is provided in the following subsections.

4.2 Source building's data transfer and data preparation module in Azure

The process in this module begins with transferring historical data collected from the Source building that to Azure Blob Storage, which stores the data sets. Azure Blob Storage is a massively scalable object storage solution for text and binary data, which provides support for big data analytics (Microsft, 2025b). The stored data is then routed to Azure Databricks, which is a unified analytics platform for building, deploying, sharing, and maintaining data analytics at scale (Microsft, 2025c). Azure Databricks pre-processes the Source building's raw data to make it suitable for developing ML models, by cleaning the data, handling any missing values, and reshaping it into structured formats. In addition, the data can be enriched through feature engineering techniques such as generating time-based features such as day of week, month, and quarter to prepare the data for effective ML model training. After processing the data in M-1, it is then transferred to M-2.

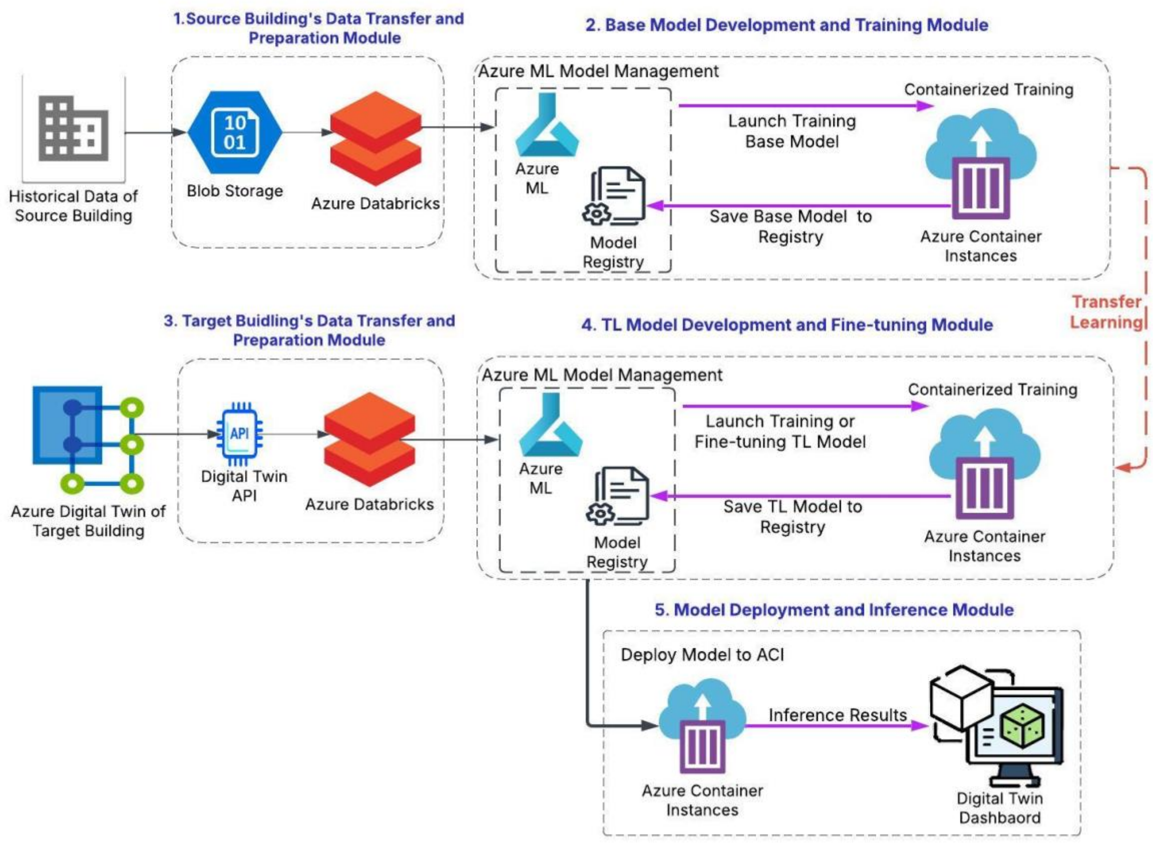
1.Source Building's Data Transfer and Preparation Module
2. Base Model Development and Training Module
Historical Data of Source Building
Blob Storage
Azure Databricks
Azure ML Model Management
Azure ML
Model Registry
Containerized Training
Launch Training Base Model
Save Base Model to Registry
Azure Container Instances
Transfer Learning
3. Target Buidling's Data Transfer and Preparation Module
4. TL Model Development and Fine-tuning Module
Azure Digital Twin of Target Building
API
Digital Twin API
Azure Databricks
Azure ML Model Management
Azure ML
Model Registry
Containerized Training
Launch Training or Fine-tuning TL Model
Save TL Model to Registry
Azure Container Instances
5. Model Deployment and Inference Module
Deploy Model to ACI
Inference Results
Azure Container Instances
Digital Twin Dashbaord

Figure 3: Proposed Azure-based framework for transfer learning.

4.2.1 Base Model development and training module in Azure

In this module, the Azure ML Model Management environment, which consists of Azure Machine Learning (Azure ML) and the Model Registry, receives the pre-processed Source building's data. Azure ML develops the Base Model from the received data. Azure ML offers a space for creating, testing and analysing ML models, which simplifies managing ML processes (Microsoft, 2024c), and enhances scalability and effectiveness of the models (Microsoft, 2024a). The Base Model is trained utilising a suitable ML algorithm that is efficient for sequential and temporal data. In this paper, Gated Recurrent Unit (GRU), which is a type of such as Recurrent Neural Network (RNN) algorithm, is used for training the Base Model. The description of GRU is provided in Section 4.2. To train the Base Model, the workflow transitions from the ML Model Management to Azure Container Instances (ACI), which is a serverless, on-demand container hosting service that is designed to run containerized workloads in a lightweight, cost-effective, and isolated environment (Microsoft, 2024b). These features are crucial for efficiently training the Base Model that requires handling a large dataset and complicated calculations during training. After training, the Base Model is saved in the Azure Model Registry, which is a central repository for storing ML models, enabling organised version control. Then, the Base Model is used to develop the TL model in M-4.

4.2.2 Target building's data transfer and data preparation module in Azure

This module creates a data pipeline transferring DT data of the Target building, including real-time readings of IoT devices, from Azure Digital Twins (Azure DT) for data pre-processing. Azure DT provides a powerful query API with extensive search conditions, including property values, relationships, relationship properties, model information to extract insights from the DT model (Microsoft, 2025b). In this module, the DT data is transferred from Azure DT to Azure Databricks through the API for pre-processing and make the data suitable for ML models. This pre-processing is an essential step prior to creating the ML model because DT data captured by IoT devices often contains noise or missing values that can degrade the performance of ML models. Azure Databricks can transform large amounts of raw data to data with quality, structured, sanitised, and enriched features (Microsoft, 2025a) through some tools that can perform data cleaning, data normalisation, and feature engineering. After pre-processing, the data is transferred to Azure ML for creating the TL model.

4.2.3 TL Model development and fine-tuning module in Azure

In this module, the TL model is developed using the Base Model received from M-2 as described in Section 2.2. Azure ML and ACI are used to develop and train the TL model. The developed TL model is then stored in the Model Registry. Version control and governance is a necessary step in the ML lifecycle as it ensures that the workflow is consistent and traceable across various other steps (Microsoft, 2025c). Since DT data is constantly generated and incorporated in the TL model for prediction, the TL model is fine-tuned with the new DT data and the TL model is continuously adapted to the changes. Therefore, the version control by Model Register is important throughout this process to track the version of the models and provide the right and the latest version of the TL model to be deployed. The trained or fine-tuned TL model is then transferred to M-5 for deployment and inference.

4.2.4 TL model deployment and inference module in Azure

In this module, the registered TL model is deployed into Azure ACI for inference. ACI can operate in isolated containers, eliminating the need for orchestration, and run event-driven applications, enabling quick deployment of the container development pipelines, and efficient data processing (Microsft, 2025a). Therefore, ACI provides a light, scalable, and effective containerised environment to perform inference tasks (Microsoft, 2024b). Through the processing of inference requests, the predictions of the Target building energy consumption are made, and the results are streamed to Azure Digital Twin Dashboard for visualisation. The dashboard provides insights into the building energy performance and enables data informed decision making for enhancing building operation strategies.

4.3 Proposed AWS-based Framework

The proposed AWS-based is presented in Figure 4. Similar to the Azure-based framework, this framework has five modules with the same functionalities that were specified in the conceptual model. The following subsections describe these five modules implemented in AWS.

4.3.1 Source building's data transfer and data preparation module in AWS

In this module, the Source building's historical data is transferred and stored in Amazon Simple Storage Service (S3). AWS S3 is a scalable data storage that is fully elastic, automatically growing and shrinking as the data is added and removed (Amazon, 2025a). The stored data is then routed to Amazon SageMaker Processing, which is capable of data pre-processing and feature engineering (Amazon, 2025e). Similar to Azure Databricks, Amazon SageMaker Processing pre-processes the Source building's data and makes it ready for the ML models through data cleaning and feature engineering techniques. Then, the pre-processed data is transferred to M-2.

4.3.2 Base Model development and training module in AWS

The pre-processed data in M-1 is used in this module to develop the Base Model. In this module, the AWS ML Model Management environment, which consists of Amazon SageMaker Studio Lab and SageMaker Registry, receives the Source building's data. Amazon SageMaker Studio Lab, which is an integrated development environment that provides tools for building, training, and deploying ML models (Amazon, 2019), develops the Base Model. In this paper, the same algorithm (i.e., GRU) is used for developing the Base Model in Azure and AWS. The developed Base Model then transitions from the ML Model Management to AWS SageMaker Endpoint, which helps train models and then deploy them into a production-ready hosted environment (Amazon, 2025b). After model training is complete, the model is registered in SageMaker Model Registry. With the Amazon SageMaker Model Registry, models are organised for version control and easy deployment to production. Finally, the trained Base Model is transferred to M-4 for TL.

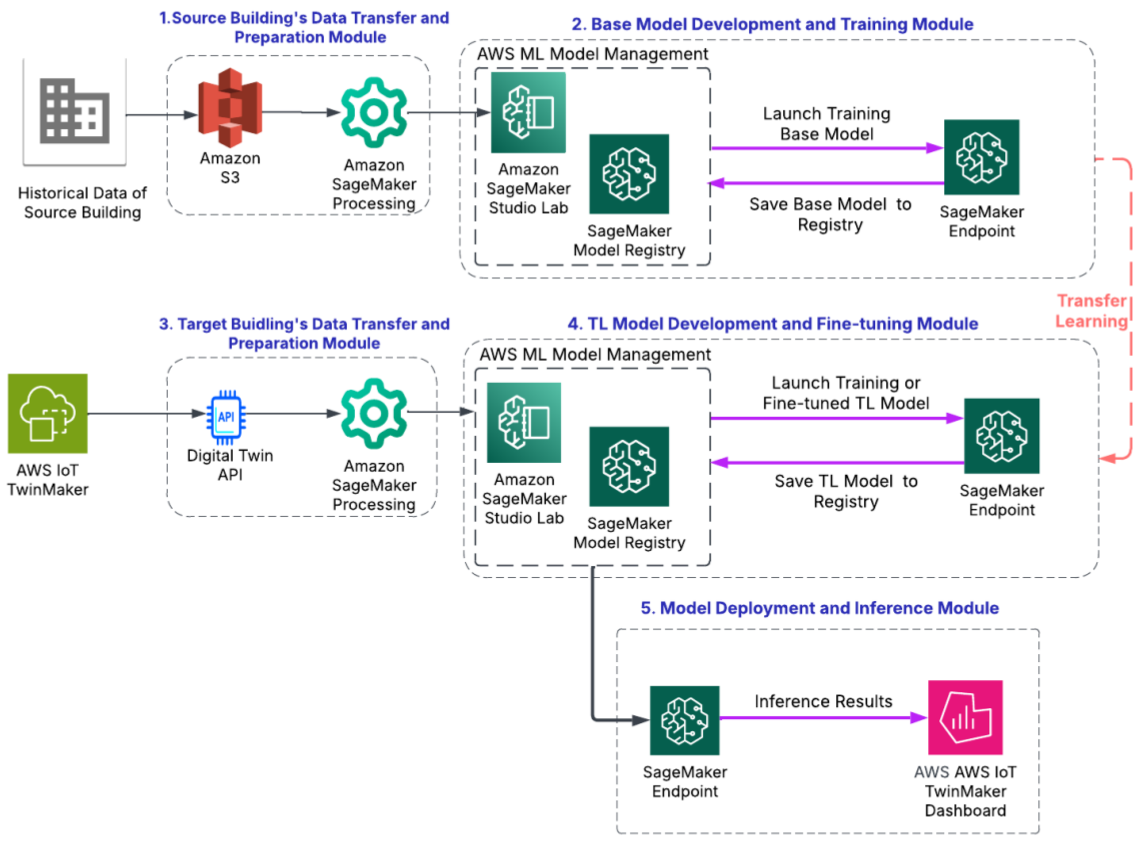
1.Source Building's Data Transfer and Preparation Module
2. Base Model Development and Training Module
AWS ML Model Management
Historical Data of Source Building
Amazon S3
Amazon SageMaker Processing
Amazon SageMaker Studio Lab
SageMaker Model Registry
Launch Training Base Model
Save Base Model to Registry
SageMaker Endpoint
Transfer Learning
3. Target Buidling's Data Transfer and Preparation Module
4. TL Model Development and Fine-tuning Module
AWS ML Model Management
AWS IoT TwinMaker
API
Digital Twin API
Amazon SageMaker Processing
Amazon SageMaker Studio Lab
SageMaker Model Registry
Launch Training or Fine-tuned TL Model
Save TL Model to Registry
SageMaker Endpoint
5. Model Deployment and Inference Module
SageMaker Endpoint
Inference Results
AWS AWS IoT TwinMaker Dashboard

Figure 4: Proposed AWS-based framework for transfer learning.

4.3.3 Target building's data transfer and data preparation module in AWS

In this module, a data pipeline is created to transfer DT data of the Target building Azure from AWS IoT TwinMaker to Amazon SageMaker Processing. AWS IoT TwinMaker provides the tools for creating DT, which enables the use of existing data from multiple sources to create virtual representations of any physical environment (Amazon, 2025d). The required DT data for building energy prediction is accessed through an API and transferred to Amazon SageMaker Processing for pre-processing. As discussed earlier, this step is crucial for addressing potential noise or missing values in the DT data of the Target building. The pre-processed data through data cleaning and feature engineering techniques in SageMaker Processing is then transferred to M-4 for creating the TL model.

4.3.4 TL Model development and fine-tuning module in AWS

In this module, the received Base Model from M-2 and the Target building's data from M-3 are used to develop the TL model. Amazon SageMaker Studio Lab and AWS SageMaker Endpoint are adopted to develop and train the TL model. SageMaker Model Registry then stores the developed TL model with a proper version control. By incorporating newly generated DT data of the Target building to the TL model, the models are fine-tuned and adapted to the changes. SageMaker Model Registry traces the version of the models throughout this updating process to provide the right and the latest version of the TL model to be deployed. Finally, the developed TL model is transferred to M-5 for deployment and inference.

4.3.5 TL model deployment and inference module in AWS

In this module, the registered TL model is deployed into SageMaker Endpoint, which serves as the entry point for real-time inference requests (Amazon, 2025f) and can provide features to manage resources and optimise inference performance when deploying ML models (Amazon, 2025g). The deployed model then predicts the Target building energy consumption, and the results are streamed to AWS IoT TwinMaker Dashboard for visualisation, which provides users with an intuitive interface to interpret and make decisions upon the insights derived.

5. CASE STUDY

The proposed frameworks are implemented in a case study of a commercial building to predict its energy consumption utilising the historical data of a similar building in London UK. We used Azure Machine Learning Studio for development and Python 3.10 as a programming language for the Azure-based framework. We also used AWS SageMaker Studio for development and Python 3.11 as a programming language for the AWS-based framework. The following subsections outline the implementation for each module of the frameworks.

5.1 Implementation of the Source building's data transfer and data preparation module

In this module, the data pipeline is created to transfer and pre-process Source building's data to make the data suitable for developing the Base Model. Data pre-processing involves data cleaning, normalising, and data engineering. In this case study, two years of historical data from the Source building is used with four features: (1) temperature, (2) wind direction, (3) wind speed, and (4) dew point temperature, and the output for the prediction is the daily energy (electricity) consumption of the building. For feature engineering, time-based features including quarter, month, week of year, day of year, day of week, a day, and weekend/weekday are created because all the used data is time-dependent, and these features could help the predictive models understand seasonal and periodic patterns that influence energy usage. To substitute missing values in a dataset (i.e., data imputation) (Yuan, 2010), the linear interpolation technique, which estimates the missing values by drawing a straight line between the known data before and after the missing data is used. This technique is suitable for weather data imputation because the selected features generally change gradually. While the prediction is made for daily energy consumption, the historical data has been collected on an hourly basis. Therefore, the historical data is aggregated to calculate their daily average to be used in the next module for developing the Base Model.

5.2 Implementation of the Base Model development and training module

In this module, the Base Model is developed and trained using the GRU algorithm, which is a type of RNN. GRU is a suitable ML algorithm for predicting energy consumption because it can solve the problem of error caused by the spatiotemporal evolution of energy consumption (Liu et al., 2017). The architecture of the model consists of four stacked GRU layers with decreasing units (i.e., number of neurons in each layer): 400, 300, 200, and 150 to progressively distil information. Each GRU layer is followed by Batch Normalisation and Dropout layers. Batch Normalisation standardises the inputs to each layer, which stabilises and accelerates training by reducing internal covariate shift. Dropout layers help prevent overfitting by randomly deactivating a fraction of neurons during training. After the GRU layers, the model includes two Dense layers with 100 and 50 units, respectively, both using ReLU activation functions. The output of the last Dense layer is a single value representing the predicted energy consumption. After training, the Base ML model is registered and safely stored to be used for developing the TL model.

5.3 Implementation of the Target building's data transfer and data preparation module

In this module, the generated DT data from the Target building is transmitted from the DT model through the API, and pre-processed. In this case study, 90 days of data from the Target building is used for the experiment. The pre-processing task in this module is similar to the one for the Source building data as described in Section 4.1. That is, the same features are selected, and the same data imputation technique and feature engineering approach are used in this module to prepare the data for developing the TL model.

5.4 Implementation of the TL Model development and fine-tuning module

In this module, the TL model is developed using the Base Model and the DT data of the Target building. The developed TL model is then registered and stored with a proper version control to be deployed in the next module. As new DT data from the Target building is generated, this process is repeated periodically by fine tuning the TL model to adapt the model to the changes.

5.5 Implementation of the TL model deployment and inference module

Finally, the registered TL model is deployed for inference and predicting the Target building energy consumption. The results are visualised on a Dashboard to provide insights into the energy consumption and support decision making.

The results of the case study are presented in the next section.

6. RESULTS

For executing the developed models, first, the compute instance for each platform should be selected. The choice of compute instances affects both performance and cost. Azure and AWS offer various specifications and pricing for their compute instance, allowing flexibility to match resources with project needs and budgets. The selected compute instances from Azure and AWS were similar in terms of the Central Processing Unit (CPU) type and Random Access Memory (RAM). In this case study, the following compute instances were used:

- Azure: Standard_E4ds_v4 with 4 vCPUs, 32GiB RAM
- AWS: AWS r5.xlarge with 4 vCPUs, 32 GiB RAM.

These instances were chosen because of their suitability for ML tasks and cost-effectiveness as they start with megabyte-sized datasets and scale them to gigabyte-sized data if needed. As of April 2025, the cost of the selected compute instance from Azure and ASW is approximately \$0.288 per hour (CloudOptimo, 2025), and \$0.302 per hour (Amazon, 2025c), respectively.

Figure 5 and Figure 6 show the results of the predictions in comparison with the actual values over the 90 days of the experiment on Azure and AWS, respectively. The errors of the predictions were measured using Mean Absolute Percentage Error (MAPE) calculated as the following formula:

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

where y_{true} represents the actual value, y_{pred} represents the predicted value, i is an observation, and n is the total number of observations.

MAPE for the models are presented in Table 1. As seen in this table, the AWS-based TL model has slightly lower MAPE than the Azure-based TL model. Since a lower MAPE indicates a more accurate model, the ASW model has a better performance in predicting the energy consumption of the Target building.

Table 1: MAPE of the TL models for the 90-day prediction.

	MAPE
Azure-based TL model	23.6%
AWS-based TL model	22.8%

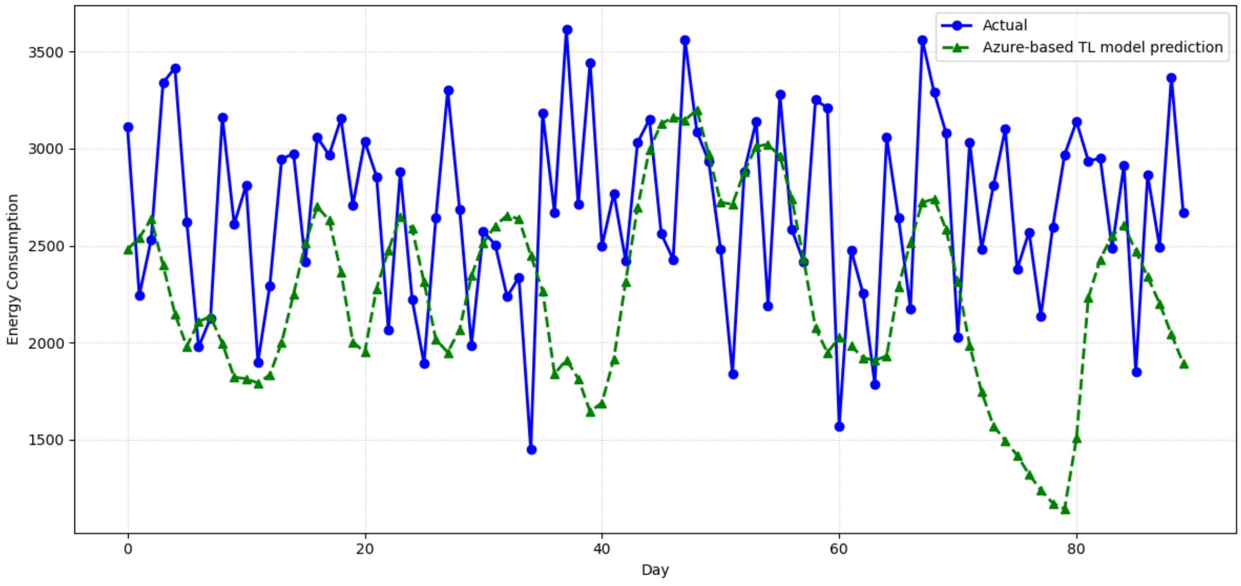

Figure 5: Azure-based TL model predictions versus actual values over the 90-day test.

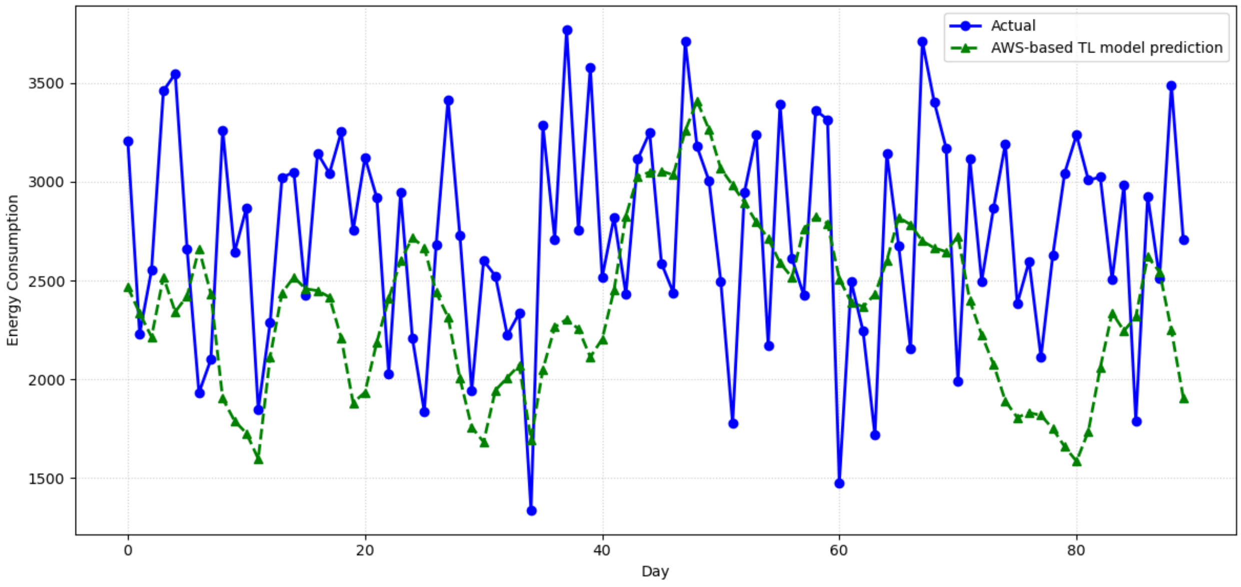

Figure 6: ASW-based TL model predictions versus actual values over the 90-day test.

7. DISCUSSION

In this section, a comparative analysis is performed to evaluate the performance of the Azure-based model and AWS-based model. In addition, the performance of the TL model is compared with the direct learning approach to underscore the value of TL for energy performance prediction with limited DT data. Finally, scalability of the frameworks is also discussed.

7.1 Comparison of data pre-processing performance

During data preprocessing, the performance of Azure and AWS can be compared using some measures such as average data loading time, average CPU usage, average RAM usage, and average throughput. Table 2 shows the details of this comparison. On average, AWS processed the data faster, taking only 0.74 seconds, while Azure took 2.25 seconds, which is a noticeable difference in speed. CPU usage was much higher on AWS (12.55%) compared to Azure (0.57%). This shows AWS used more processing power to finish the task faster while Azure used very little CPU, but it took longer. Azure used more RAM (357.80 MB) than AWS (267.29 MB). So, AWS was more efficient than Azure in terms of RAM usage. Finally, looking at throughput, which indicates how much data was processed per second, AWS showed a better performance. It had an average throughput of 0.10 MB/s, compared to 0.03 MB/s for Azure.

Overall, while both Azure and AWS offer similar computational power specifications, AWS was faster and more efficient in data loading and had higher throughput, but it used more CPU resources to achieve this better performance. Azure, on the other hand, used less CPU but took longer and used more RAM for pre-processing. The choice between the two may depend on whether the priority is the speed or resource utilisation.

Table 2: Comparison of data pre-processing performance between Azure and AWS models.

	Average data loading time (s)	**Average CPU usage (%)**	**Average RAM usage (MB)**	**Average throughput (MB/s)**
Azure	2.25	0.57	357.80	0.03
AWS	0.74	12.55	267.29	0.10

7.2 Training loss and validation loss comparison

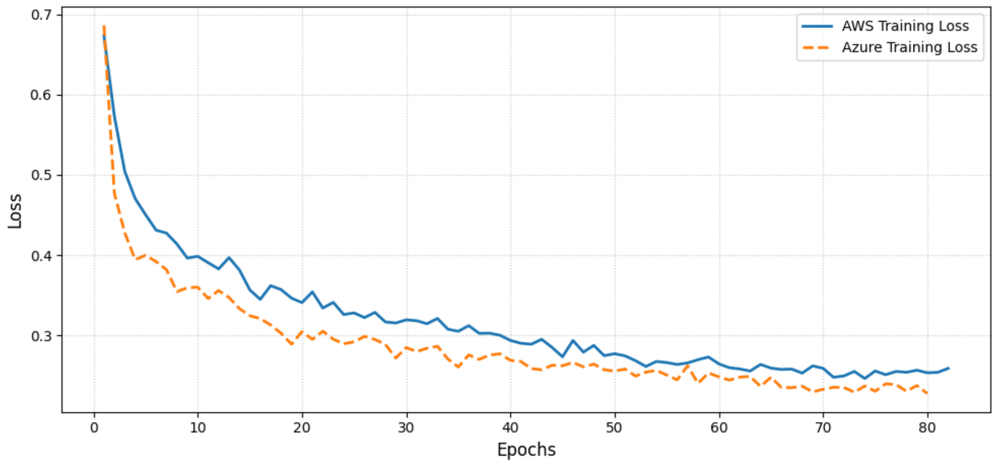

Figure 7: Training Loss During Base Model Training: Azure versus AWS.

Throughout training the Base Model, the average training loss (which refers to the error the model makes on the data it learns from), and average validation loss (which illustrates the error that the model makes on unseen data) were monitored in each complete training iteration (which is called an epoch) to evaluate the learning curve of the ML model and ensure that the training process is healthy. The training loss and the validation loss are shown in Figure 7 and Figure 8, respectively. As shown in these two figures, the loss values have decreasing trends over the epochs. However, there are some differences between the two platforms. On the training loss curve, Azure consistently maintains a slightly lower loss compared to AWS. On the validation loss curve, Azure's line is mostly lower (particularly at the beginning) and generally smoother than that of AWS, which shows more fluctuation, especially in the earlier epochs. This observation is interesting because both models were trained under similar conditions (e.g., CPU type and memory configuration) using the exact same architecture and the same input data, but Azure performed slightly better. This could be related to the way that Azure and AWS handle data loading, memory allocation, or internal Input/Output operations. The operating system and the configuration of the background processes could also affect training speed and consistency. These small differences can have a noticeable impact when training deep learning models, even on CPU-based machines. The consistency of Azure in lowering both training and validation data suggests that system-level optimisations on Azure could be support a more efficient training.

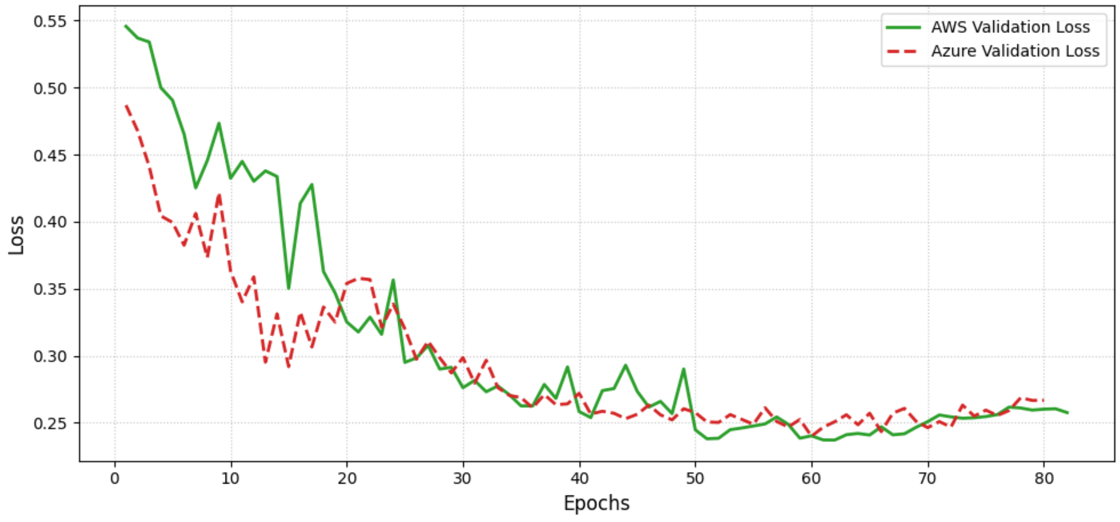

Figure 8: Validation Loss During Base Model Training: Azure versus AWS.

Throughout training the TL Model, the training loss and validation loss had a different trend as shown in Figure 9 and Figure 10. As seen in Figure 9, the training loss of Azure started with a higher loss of about 0.166 while that of AWS started around 0.150. However, the Azure line drops smoothly and steadily across epochs, eventually reaching about 0.126. In contrast, AWS's training loss fluctuated throughout the 15 epochs, staying mostly between 0.135 and 0.153, without a strong downward trend. This suggests Azure's system allowed the model to learn more consistently during training. The smoother behaviour might be influenced by background system load or how the compute handles optimization under the hood. However, for the validation loss, AWS consistently performs better than Azure as shown in Figure 10. AWS's validation loss stays around 0.135 to 0.142 for most epochs. Azure's validation loss starts at around 0.188 and slowly decreases but only reaches about 0.157 by the final epoch. Since validation loss indicates how well the model performs on unseen data, the lower values from AWS suggest better generalisation ability.

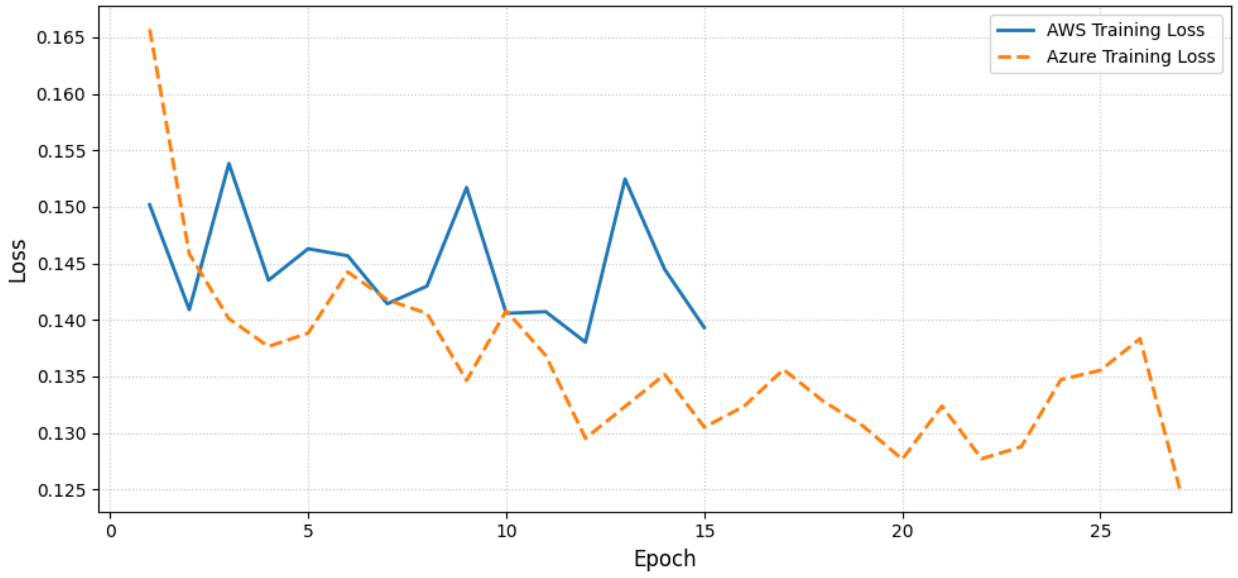

Figure 9: Training loss during training the TL model: Azure versus AWS.

For further investigation, MAPE, which is a more interpretable metric for understanding prediction errors, was plotted over Epochs for the TL model training as presented in Figure 11. As seen in this figure, AWS starts with an MAPE of around 31.8%, fluctuating slightly between 27.2% and 31.8%. Azure starts with a higher MAPE, close to 37.1%, but improves it gradually. By the end of the training, Azure reaches a final MAPE of about 25.5%, which is better than AWS's ending value of around 29%. That means Azure ended with more accurate predictions than AWS even though it started with less accuracy.

Overall, Azure demonstrated smoother training and slightly better final prediction accuracy than AWS. Meanwhile, AWS produced lower validation loss, suggesting better generalisation on unseen data. These results illustrate that even when cloud platforms use the same model and data, small differences in system performance and environment behaviour can affect how well models are trained and performed.

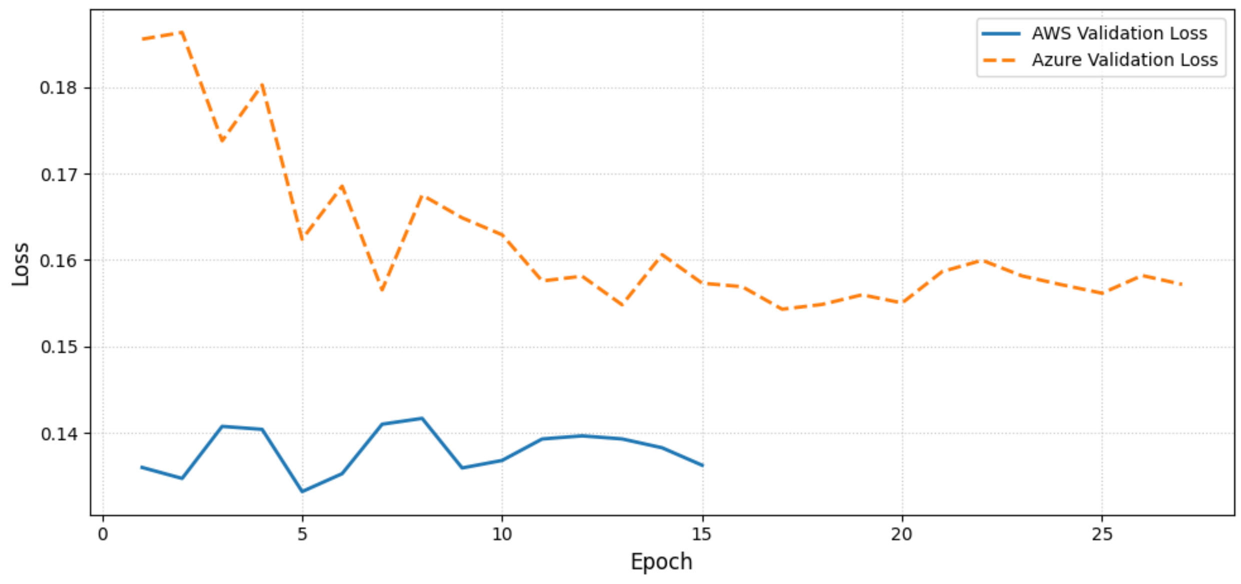

Figure 10: Validation loss during training the TL model: Azure versus AWS

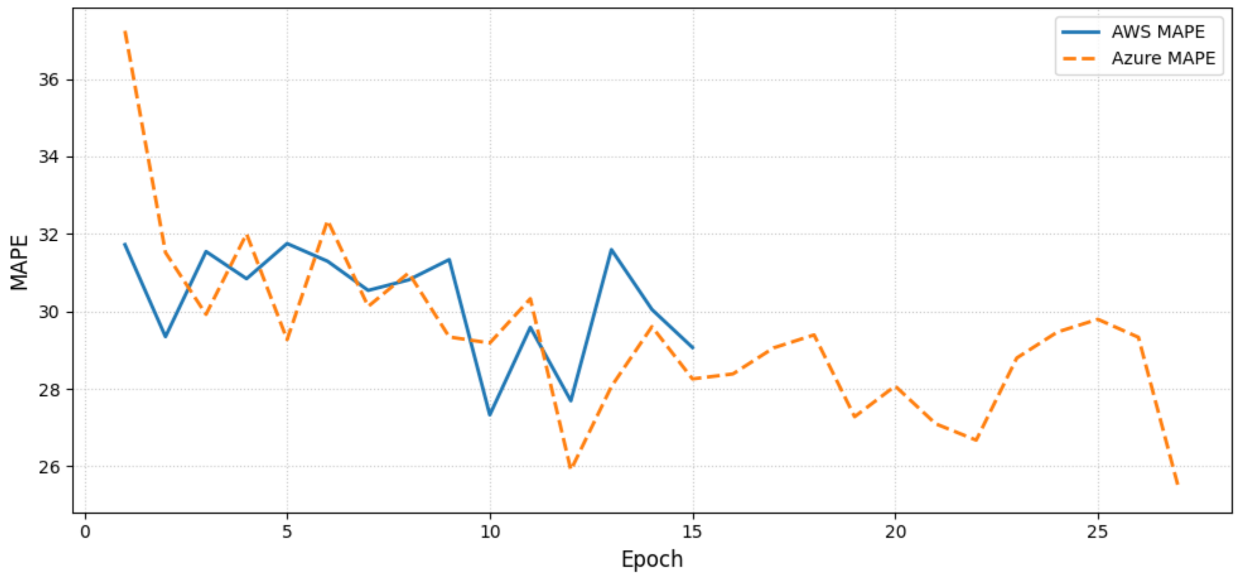

Figure 11: MAPE over epochs for the TL model training: Azure versus AWS.

7.3 Comparison of training time

The training time is another measure for comparing Azure and AWS. The time needed to train the Base Model can be time-consuming as it involves developing a new ML model from a relatively large historical dataset. The time needed for training the TL model and fine-tuning it could be less time-consuming as it requires only updating the models with a relatively small dataset. The time spent on training both the Base Model and the TL model was measured as presented in Table 3. As seen in this table, Azure is approximately 33% faster than AWS in training the Base Model and 42% faster in training the TL model.

Table 3: Training time comparison between Azure and AWS models.

	Training time for the Base Model (s)	**Training time for the TL Model (s)**
Azure-based model	398	46
AWS-based model	593	79

7.4 Comparison of CPU and RAM usage ratios

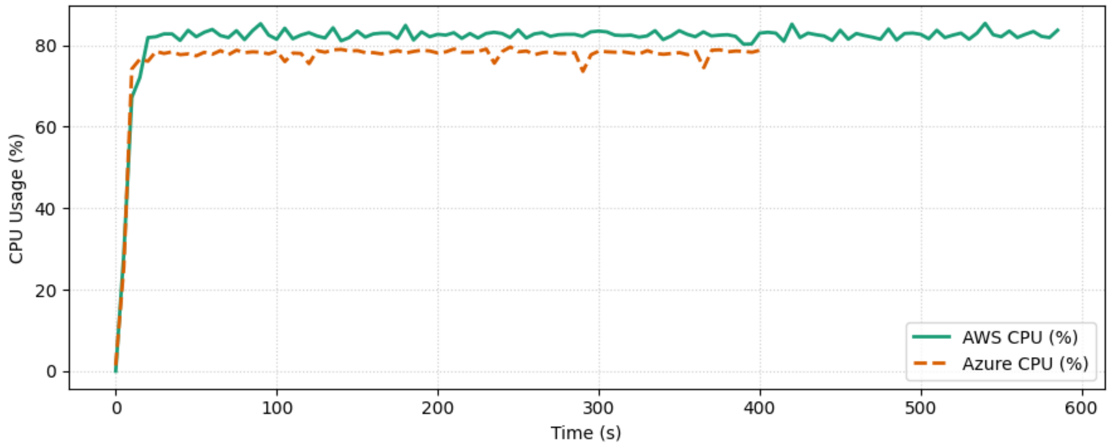

Figure 12: CPU usage during the Base Model training: AWS versus Azure.

CPU and RAM usage ratios during training models are useful measures for evaluating resource utilisation by cloud platforms. Figure 12 and Figure 13 show the CPU and RAM usage ratios during the Base Model training, respectively. These figures reveal an interesting contrast considering the fact that both platforms used similar computational configurations. AWS consistently showed higher CPU usage, staying above 80% for most of the training period. Its RAM usage was also relatively high, around 24%. In contrast, Azure used slightly less CPU but significantly less RAM (about 7.5%) than AWS. Despite the fact that AWS utilised more resources, it didn't lead to faster training. Azure completed the task in 398 seconds, while AWS took 593 seconds to complete it. This suggests that Azure utilises not only fewer resources but also uses them more efficiently. There could be a few possible technical reasons why Azure performed better. For instance, Azure could have lower system overhead, which allowed more resources to go directly to the training task. It is also possible that disk speed, memory bandwidth, or input/output handling was more optimised in Azure for this kind of workload.

The CPU and RAM usage ratios during the TL Model training are shown in Figure 14 and Figure 15, respectively. As seen in these figures, the general trends of CPU and RAM usage ratios during the TL Model training are similar to those observed during the Base Model training. AWS maintained higher CPU usage throughout the training, hovering steadily around 70–80%. Azure's CPU usage was slightly lower, closer to 70%, and showed a few brief dips. RAM usage for AWS stayed between 24% and 25%, while Azure consistently used much less, between 8% and 9%. Similar to the Base Model training, Azure completed the TL Model training tasks in 46 seconds, whereas AWS took 79 seconds to complete them.

Overall, Azure could deliver the results faster with fewer resources, showing more efficiency in this specific model training process.

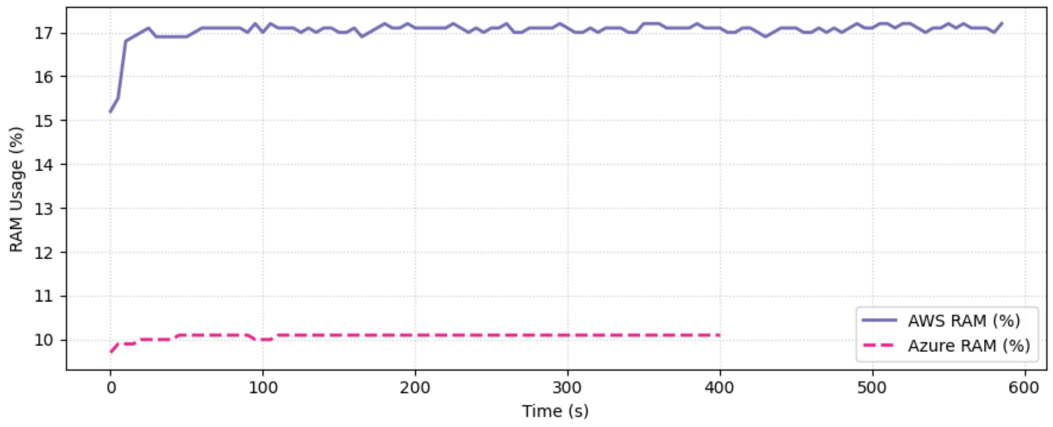

Figure 13: RAM usage during the Base model training: AWS versus Azure.

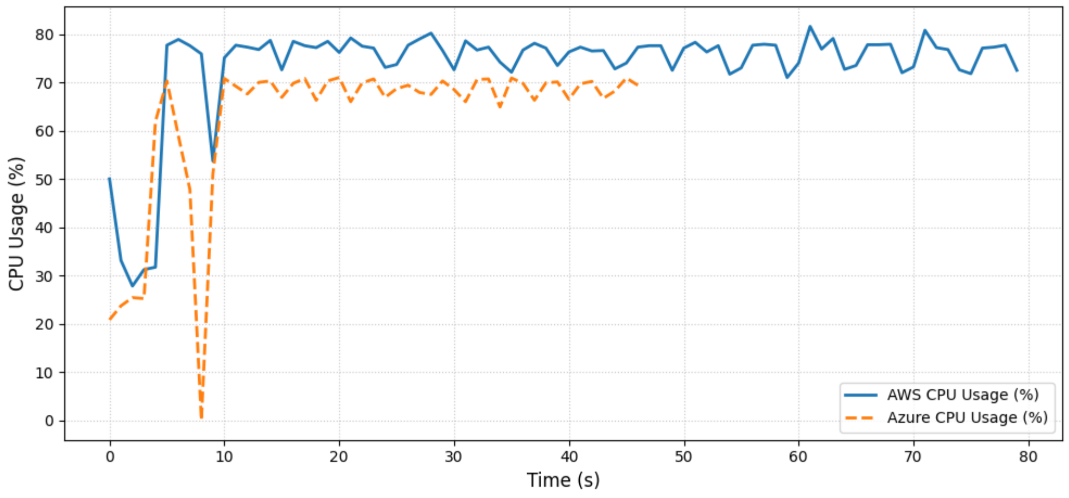

Figure 14: CPU usage during the TL Model training: Azure versus AWS.

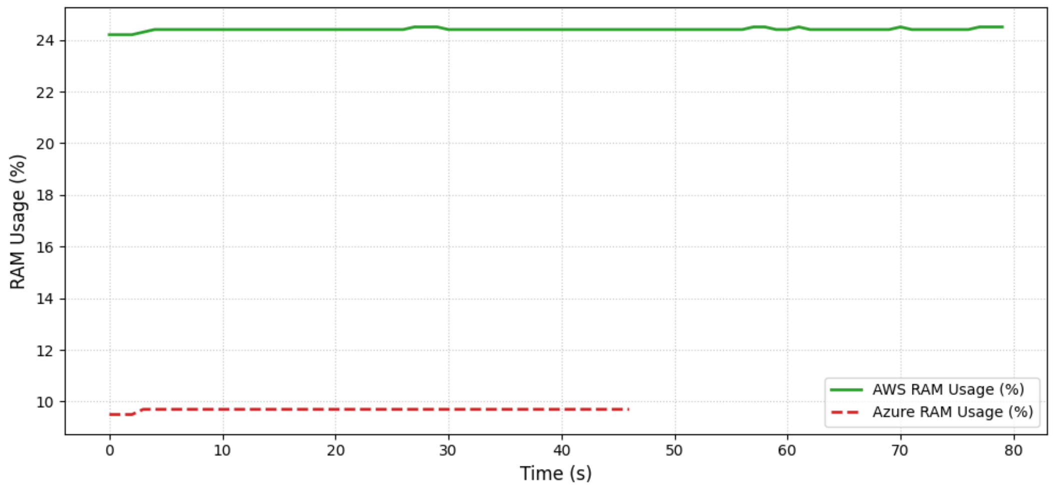

Figure 15: RAM usage during the TL model training: Azure versus AWS.

7.5 Comparison between TL and direct learning

To demonstrate the effectiveness of TL in improving ML model prediction compared to direct learning when DT data is limited, an ML model with the same algorithm was created using the 60 days of the data for training and 30 days of the data for testing. The results of the experiment on predicting 30 days of the data using TL and direct learning models are shown in Table 4. These results indicate that TL could reduce prediction errors by approximately 8.5 percentage points in both Azure-based and AWS-based models. This finding aligns with existing research (e.g., (Lu et al., 2021) and (Mahamedi et al., 2024)) demonstrating the effectiveness of TL in addressing data scarcity for developing ML models and enhancing accuracy of the models compared to direct learning approaches.

Table 4: MAPE of the TL models and direct learning models for the 30-day prediction.

	MAPE
Azure-TL model	28.42%
AWS-based TL model	26.21%
Azure-based direct learning model	36.88%
AWS-based direct learning model	34.83%

7.6 Data scalability and model scalability of the proposed frameworks

The proposed frameworks were implemented on cloud platforms, which streamlined the data flows from historical data of the Source building to the Base Model, and DT data of the Target building to the TL model through data pipelines. All the data pipelines and the ML models were created on a single cloud platform, which enhanced data scalability through seamless integration of all required services. From the computational perspective, cloud computing is flexible for scaling resources up and down based on the computational demand. For instance, Blob Storage and Amazon S3 are scalable solutions for data storage allowing for storing large datasets from the Source and Target buildings. In addition, Azure and AWS offers various compute instances to be selected based on the required computational needs, which is very critical for handling real-time DT data. Hence, the user has access to a variety of high-computational resources only when needed. Particularly in this case that the TL model needs to be fine tuned in certain intervals, cloud computing is more cost effective than expensive on-premises hardware because the cost of cloud services is based on the usage.

The proposed frameworks also consider model scalability by continuously incorporating dynamic DT data into the TL model and fine tuning the TL models based on the newly generated DT data. In this way, the models are adapted it to the changing conditions of the building energy performance, which are reflected in the DT data. In summary, these findings collectively demonstrate that cloud-based TL frameworks can effectively address the scalability challenges of building energy prediction when DT data is limited.

8. CONCLUSION

This paper focused on developing scalable TL frameworks for predicting building energy consumption from limited DT data using cloud computing (i.e., Azure and AWS). The frameworks were implemented and tested in a case study, demonstrating their applicability and practicality in handling large computational workloads and improving prediction accuracy. Based on the comparison made between the TL models and direct learning models, it was shown that the developed TL model could reduce the prediction errors (i.e., MAPE) by 8.5 percentage points. Through the conducted comparative analysis between the Azure-based models and AWS-based models, it was revealed that Azure surpassed AWS in some metrics while AWS outperformed Azure in others. Azure is approximately 33% faster than AWS in training the Base Model and 42% faster in training the TL model. In addition, the CPU and RAM usage in Azure was lower than in AWS, showing a more efficient training process in Azure. On the other hand, AWS had less data preprocessing time (0.74s versus 2.25s) and slightly lower MAPE than Azure (22.8% versus 23.6%). The findings of these analyses can help users select the most suitable cloud platform based on their priority factors.

The main contribution of this research is the development of frameworks that meet several requirements for employing ML models to predict building energy consumption, including:

- Addressing the challenge of data scarcity for developing ML models from limited DT data by adopting TL
- Integrating dynamic DT data into the TL process for energy consumption prediction
- Addressing both model scalability and data scalability of TL models by employing cloud computing
- Guiding researchers and practitioners on capabilities of two leading cloud platforms (i.e., Azure and AWS) by testing and analysing their performance in a case study

This research can further advance the application of DT in building energy predictions in both research and practice as DT is still in its early stage of adoption in the building sector, and DT data availability is limited for many buildings.

There are some limitations in this paper that could be addressed in future research. In this paper, the unit cost of using the selected compute instances in Azure and AWS was provided as a reference. It should be noted that executing the developed TL models entails additional costs for other services, such as data storage and data pre-processing. Due to the complexity of estimating the total cost for all used services, the cost metric was not included in the comparative analysis. Additionally, there could be other decision factors for selecting the most suitable cloud platform, such as compatibility with existing IT infrastructures. Future research can investigate the cost of adopting cloud platforms for predicting building energy consumption, as well as identifying a more comprehensive list of decision factors for selecting the most suitable cloud platform.

This paper assumed that the DT model of the Target building was already in place, and its data was available to be streamed into the TL model. Future research could extend the developed frameworks to incorporate DT model development into the workflow. The findings of the comparative analysis presented in this paper depend on selected compute instances. Further experiments with various computing configurations could be carried out to generalise the results. In this paper, only one ML algorithm (i.e., GRU) was used for developing the base model. While the framework is independent of the algorithm type, future research could evaluate the impact of different ML algorithms on the performance of cloud platforms and their prediction accuracy.

ACKNOWLEDGEMENT

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

REFERENCES

Amasyali, K., & El-Gohary, N. M. (2021). Real data-driven occupant-behavior optimization for reduced energy consumption and improved comfort. Applied Energy, 302, 117276. https://doi.org/10.1016/j.apenergy.2021.117276

Amazon. (2019). Introducing Amazon SageMaker Studio – the first integrated development environment (IDE) for machine learning. Retrieved April 19, 2025 from https://aws.amazon.com/about-aws/whats-new/2019/12/introducing-amazon-sagemaker-studio-the-first-integrated-development-environment-ide-for-machine-learning/

Amazon. (2025a). Amazon S3. Retrieved April 19, 2025 from https://aws.amazon.com/s3/

Amazon. (2025b). Amazon SageMaker AI endpoints. Retrieved April 19, 2025 from https://docs.aws.amazon.com/prescriptive-guidance/latest/image-classification/sagemaker.html#:~:text=Amazon%20SageMaker%20AI%20is%20a,SageMaker%20multi%2Dlabel%20image%20classification

Amazon. (2025c). Amazon SageMaker AI pricing. Retrieved April 19, 2025 from https://aws.amazon.com/sagemaker-ai/pricing/

Amazon. (2025d). AWS IoT TwinMaker. Retrieved April 19, 2025 from https://aws.amazon.com/iot-twinmaker/

Amazon. (2025e). Data transformation workloads with SageMaker Processing. Retrieved April 19, 2025 from https://docs.aws.amazon.com/sagemaker/latest/dg/processing-job.html

Amazon. (2025f). Invoke models for real-time inference. Retrieved April 19, 2025 from https://docs.aws.amazon.com/sagemaker/latest/dg/realtime-endpoints-test-endpoints.html

Amazon. (2025g). Model deployment options in Amazon SageMaker AI. Retrieved April 19, 2025 from https://docs.aws.amazon.com/sagemaker/latest/dg/how-it-works-deployment.html#:~:text=After%20you%20train%20your%20machine,inference%20with%20Amazon%20SageMaker%20AI

Arsiwala, A., Elghaish, F., & Zoher, M. (2023). Digital twin with Machine learning for predictive monitoring of CO2 equivalent from existing buildings. Energy and buildings, 284, 112851. https://doi.org/10.1016/j.enbuild.2023.112851

Awan, A., Kocoglu, M., Subhan, M., Utepkaliyeva, K., & Hossain, M. E. (2025). Assessing energy efficiency in the built environment: A quantile regression analysis of CO2 emissions from buildings and manufacturing sector. Energy and Buildings, 338, 115733. https://doi.org/10.1016/j.enbuild.2025.115733

Chen, Y., Tong, Z., Zheng, Y., Samuelson, H., & Norford, L. (2020). Transfer learning with deep neural networks for model predictive control of HVAC and natural ventilation in smart buildings. Journal of Cleaner Production, 254, 119866. https://doi.org/10.1016/j.jclepro.2019.119866

CloudOptimo. (2025). VM - Standard-E4ds-v4 Pricing. Retrieved April 19, 2025 from https://costcalc.cloudoptimo.com/azure-pricing-calculator/vm/Standard-E4ds-v4

De Simone, M. (2022). Modeling occupant behavior and quantifying its impact on building energy use. Frontiers in Sustainable Cities, 4, 905237. https://doi.org/10.3389/frsc.2022.905237

Digitemie, W. N., & Ekemezie, I. O. (2024). A comprehensive review of Building Energy Management Systems (BEMS) for improved efficiency. World Journal of Advanced Research and Reviews, 21(3), 829-841. https://doi.org/10.30574/wjarr.2024.21.3.0746

ElArwady, Z., Kandil, A., Afiffy, M., & Marzouk, M. (2024). Modeling Indoor Thermal Comfort in Buildings using Digital Twin and Machine Learning. Developments in the Built Environment, 100480. https://doi.org/10.1016/j.dibe.2024.100480

Elnour, M., Ahmad, A. M., Abdelkarim, S., Fadli, F., & Naji, K. (2024). Empowering smart cities with digital twins of buildings: Applications and implementation considerations of data-driven energy modelling in building management. Building Services Engineering Research & Technology, 45(4), 475-498. https://doi.org/10.1177/01436244241239290

Elnour, M., Himeur, Y., Fadli, F., Mohammedsherif, H., Meskin, N., Ahmad, A. M., Petri, I., Rezgui, Y., & Hodorog, A. (2022). Neural network-based model predictive control system for optimizing building automation and management systems of sports facilities. Applied Energy, 318, 119153. https://doi.org/10.1016/j.apenergy.2022.119153

Energy Performance of Buildings Directive (2025). Retrieved December 30, 2025 from https://energy.ec.europa.eu/topics/energy-efficiency/energy-performance-buildings/energy-performance-buildings-directive_en

Fan, C., Sun, Y., Xiao, F., Ma, J., Lee, D., Wang, J., & Tseng, Y. C. (2020). Statistical investigations of transfer learning-based methodology for short-term building energy predictions. Applied Energy, 262, 114499. https://doi.org/10.1016/j.apenergy.2020.114499

Fang, X., Gong, G., Li, G., Chun, L., Li, W., & Peng, P. (2021). A hybrid deep transfer learning strategy for short term cross-building energy prediction. Energy, 215, 119208.

Gao, Y., Ruan, Y., Fang, C., & Yin, S. (2020). Deep learning and transfer learning models of energy consumption forecasting for a building with poor information data. Energy and buildings, 223, 110156. https://doi.org/10.1016/j.enbuild.2020.110156

Grieves, M. (2014). Digital twin: manufacturing excellence through virtual factory replication. White paper, 1(2014), 1-7. https://www.researchgate.net/publication/275211047_Digital_Twin_Manufacturing_Excellence_through_Virtual_Factory_Replication

Han, F., Du, F., Jiao, S., & Zou, K. (2024). Predictive Analysis of a Building's Power Consumption Based on Digital Twin Platforms. Energies (19961073), 17(15). https://doi.org/10.3390/en17153692

Hooshmand, A., & Sharma, R. (2019). Energy predictive models with limited data using transfer learning. Proceedings of the tenth ACM international conference on future energy systems. https://doi.org/10.1145/3307772.3328284.

Hussien, A., Maksoud, A., Al-Dahhan, A., Abdeen, A., & Baker, T. (2025). Machine learning model for predicting long-term energy consumption in buildings. Discover Internet of Things, 5(1), 1-24. https://doi.org/10.1007/s43926-025-00115-7

Jradi, M., Madsen, B. E., & Kaiser, J. H. (2023). DanRETwin: A digital twin solution for optimal energy retrofit decision-making and decarbonization of the Danish building stock. Applied Sciences, 13(17), 9778. https://doi.org/10.3390/app13179778

Kapoor, N. R., Kumar, A., Kumar, A., Kumar, A., Mohammed, M. A., Kumar, K., Kadry, S., & Lim, S. (2022). Machine learning-based CO2 prediction for office room: a pilot study. Wireless communications and mobile computing, 2022(1), 9404807. https://doi.org/10.1155/2022/9404807

Kawa, B., & Borkowski, P. (2023). Integration of Machine Learning Solutions in the Building Automation System. Energies, 16(11), 4504. https://doi.org/10.3390/en16114504

Li, A., Xiao, F., Fan, C., & Hu, M. (2021). Development of an ANN-based building energy model for information-poor buildings using transfer learning. Building simulation. https://doi.org/10.1007/s12273-020-0711-5

Li, G., Wang, Z., Gao, J., Xu, C., Guo, Y., Sun, D., & Fang, X. (2024). Performance assessment of cross office building energy prediction in the same region using the domain adversarial transfer learning strategy. Applied Thermal Engineering, 241, 122357. https://doi.org/10.1016/j.applthermaleng.2024.122357

Liu, B., Fu, C., Bielefield, A., & Liu, Y. Q. (2017). Forecasting of Chinese primary energy consumption in 2021 with GRU artificial neural network. Energies, 10(10), 1453. https://doi.org/10.3390/en10101453

Lu, Y., Tian, Z., Zhou, R., & Liu, W. (2021). A general transfer learning-based framework for thermal load prediction in regional energy system. Energy, 217, 119322. https://doi.org/10.1016/j.energy.2020.119322

Mahamedi, E., Wonders, M., Gerami Seresht, N., Woo, W. L., & Kassem, M. (2024). A reinforcing transfer learning approach to predict buildings energy performance. Construction Innovation, 24(1), 242-255. https://doi.org/10.1108/CI-12-2022-0333

Mahamedi, E., Suliman, A., & Wonders, M. (2025). A Cloud-Based Framework for Creating Scalable Machine Learning Models Predicting Building Energy Consumption from Digital Twin Data. Architecture, 5(2), 29. https://doi.org/10.3390/architecture5020029

Microsft. (2025a). Azure Container Instances documentation. Retrieved April 19, 2025 from https://learn.microsoft.com/en-us/azure/databricks/introduction/

Microsft. (2025b). Introduction to Azure Storage. Retrieved April 19, 2025 from https://learn.microsoft.com/en-us/azure/storage/common/storage-introduction

Microsft. (2025c). What is Azure Databricks? Retrieved April 19, 2025 from https://learn.microsoft.com/en-us/azure/databricks/introduction/

Microsoft. (2024a). What is automated machine learning (AutoML)? Retrieved Jan 13, 2025 from https://learn.microsoft.com/en-us/azure/machine-learning/concept-automated-ml?view=azureml-api-2

Microsoft. (2024b). What is Azure Container Instances? Retrieved Jan 13, 2025 from https://learn.microsoft.com/en-us/azure/container-instances/container-instances-overview

Microsoft. (2024c). What is Azure Machine Learning? Retrieved Jan 13, 2025 from https://learn.microsoft.com/en-us/azure/machine-learning/overview-what-is-azure-machine-learning?view=azureml-api-2

Microsoft. (2025a). What is Azure Databricks? Retrieved Feb 10, 2025 from https://learn.microsoft.com/en-us/azure/databricks/introduction/

Microsoft. (2025b). What is Azure Digital Twins? Retrieved April 19, 2025 from https://learn.microsoft.com/en-us/azure/digital-twins/overview

Microsoft. (2025c). Work with registered models in Azure Machine Learning. Retrieved Jan 13, 2025 from https://learn.microsoft.com/en-us/azure/machine-learning/how-to-manage-models?view=azureml-api-2&tabs=python

Mishra, A. (2024). Scalable AI and Design Patterns: Design, Develop, and Deploy Scalable AI Solutions. Springer Nature. https://doi.org/10.1007/979-8-8688-0158-7

Moudgil, V., Sadiq, R., Bakhtavar, E., Paudel, A., & Hewage, K. (2024). A cloud-oriented data-analysis framework to analyze peak demand dynamics in institutional building clusters. Sustainable Cities and Society, 111, 105553. https://doi.org/10.1016/j.scs.2024.105553

NASA (2025). Why does the world (and NASA) need digital twins? Retrieved December 30, 2025 from https://science.nasa.gov/biological-physical/why-does-the-world-and-nasa-need-digital-twins/

Ni, Z., Liu, Y., Karlsson, M., & Gong, S. (2021). A sensing system based on public cloud to monitor indoor environment of historic buildings. Sensors, 21(16), 5266. https://doi.org/10.3390/s21165266

Olu-Ajayi, R., Alaka, H., Sulaimon, I., Sunmola, F., & Ajayi, S. (2022). Building energy consumption prediction for residential buildings using deep learning and other machine learning techniques. Journal of Building Engineering, 45, 103406. https://doi.org/10.1016/j.jobe.2021.103406

Panigrahi, S., Nanda, A., & Swarnkar, T. (2021). A survey on transfer learning. Intelligent and Cloud Computing: Proceedings of ICICC 2019, Volume 1. https://doi.org/10.1007/978-981-15-5971-6_83

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. Journal of management information systems, 24(3), 45-77. https://doi.org/10.2753/MIS0742-1222240302

Pham, A.-D., Ngo, N.-T., Truong, T. T. H., Huynh, N.-T., & Truong, N.-S. (2020). Predicting energy consumption in multiple buildings using machine learning for improving energy efficiency and sustainability. Journal of Cleaner Production, 260, 121082. https://doi.org/10.1016/j.jclepro.2020.121082.

Pinto, G., Wang, Z., Roy, A., Hong, T., & Capozzoli, A. (2022). Transfer learning for smart buildings: A critical review of algorithms, applications, and future perspectives. Advances in Applied Energy, 5, 100084. https://doi.org/10.1016/j.adapen.2022.100084

Ribeiro, M., Grolinger, K., ElYamany, H. F., Higashino, W. A., & Capretz, M. A. (2018). Transfer learning with seasonal and trend adjustment for cross-building energy forecasting. Energy and buildings, 165, 352-363. https://doi.org/10.1016/j.enbuild.2018.01.034

Snyder, H. (2019). Literature review as a research methodology: An overview and guidelines. Journal of business research, 104, 333-339. https://doi.org/10.1016/j.jbusres.2019.07.039

Tahmasebinia, F., Lin, L., Wu, S., Kang, Y., & Sepasgozar, S. (2023). Exploring the benefits and limitations of digital twin technology in building energy. Applied Sciences, 13(15), 8814. https://doi.org/10.3390/app13158814

Tian, Y., Sehovac, L., & Grolinger, K. (2019). Similarity-based chained transfer learning for energy forecasting with big data. IEEE Access, 7, 139895-139908. https://doi.org/ 10.1109/ACCESS.2019.2943752

Vom Brocke, J., Hevner, A., & Maedche, A. (2020). Introduction to design science research. Design science research. Cases, 1-13. https://doi.org/10.1007/978-3-030-46781-4_1

Wang, L., Kubichek, R., & Zhou, X. (2018). Adaptive learning based data-driven models for predicting hourly building energy use. Energy and buildings, 159, 454-461. https://doi.org/10.1016/j.enbuild.2017.10.054

Wang, S., Du, L., & Zhou, Q. (2019). A semi-supervised deep transfer learning architecture for energy disaggregation. 2019 IEEE power & energy society general meeting (PESGM). https://doi.org/10.1109/PESGM40551.2019.8973556

Xing, Z., Pan, Y., Yang, Y., Yuan, X., Liang, Y., & Huang, Z. (2024). Transfer learning integrating similarity analysis for short-term and long-term building energy consumption prediction. Applied Energy, 365, 123276. https://doi.org/10.1016/j.apenergy.2024.123276.

Yang, S., Wan, M. P., Chen, W., Ng, B. F., & Dubey, S. (2020). Model predictive control with adaptive machine-learning-based model for building energy efficiency and comfort optimization. Applied Energy, 271, 115147. https://doi.org/10.1016/j.apenergy.2020.115147.

Yuan, Y. C. (2010). Multiple imputation for missing data: Concepts and new development (Version 9.0). SAS Institute Inc, Rockville, MD, 49(1-11), 12. http://facweb.cdm.depaul.edu/sjost/csc423/documents/multipleimputation.pdf

Zhang, S., Yao, R., Toftum, J., Essah, E., & Li, B. (2024). Machine learning-based approach to predict thermal comfort in mixed-mode buildings: Incorporating adaptive behaviors. Journal of Building Engineering, 87, 108877. https://doi.org/10.1016/j.jobe.2024.108877

Zheng, L., Mueller, M., Luo, C., & Yan, X. (2024). Predicting whole-life carbon emissions for buildings using different machine learning algorithms: A case study on typical residential properties in Cornwall, UK. Applied Energy, 357, 122472. https://doi.org/10.1016/j.apenergy.2023.122472