Chapter 4

Modern Machine Learning Solution for Electricity Consumption Management in Smart Buildings

This chapter focuses on the first contribution of this thesis detail in Section. Sectin 4.1: We provide an introduction and HDL used in Section 4.1. Section 4.2: We describe the models used in this chapter 4.2. Section 4.3: Demonstrates the proposed work for EM 4.3. Section 4.4: The experimental setup and result obtained using the proposed technique 4.4.Section 4.5: Summary of the work done 4.5.

4.1 Introduction

Energy efficiency entails using energy resources in the most optimal manner to avoid waste [57]. With approximately 40% of the world's energy consumed by buildings—primarily due to lighting, HVAC systems, and electronics—inefficient usage not only increases operational costs but also causes buildings to surpass transportation as a major source of carbon emissions. Accurate prediction of energy consumption is therefore vital for smarter EM and for reducing environmental impact. SBs are equipped with advanced technologies that maximize energy usage, thereby minimizing waste and lowering carbon emissions efficiently [57]. By leveraging IoT technologies, these buildings automatically monitor and control energy usage, promoting efficiency without compromising occupant comfort [58]. Intelligent systems rely on accurate load forecasting to manage energy demands effectively and maintain equilibrium between supply and consumption. Such forecasts bridge the gap between energy generation and usage, thus improving power system stability. With real-time monitoring and predictive analytics, SBs can optimize energy use while maintaining indoor comfort, representing a modern approach to sustainable operations and a resilient energy infrastructure. As early as 1984, a UK study reported that domestic buildings were losing approximately £10 million annually due to poor energy estimation practices [56]. With buildings projected to consume over 65% of global energy by 2050 and CO_2 emissions potentially reaching 1.3 billion metric tons by 2030 [59], the importance of accurate load forecasting has become more pressing. In alignment with the evolution of smart cities, SBs are increasingly leveraging natural resources and data-driven insights to improve forecasting accuracy. Accurate energy predictions play a crucial role in unlocking the full potential of SBs by enabling efficient resource utilization, minimizing ecological impact, and supporting sustainable urban development. Time series forecasting, a key method used to predict future values based on historical trends, assumes continuity of patterns and has found widespread application in energy analytics. Both statistical methods such as regression and ARIMA (Auto Regressive Integrated Moving Average) [59] \cite{paper7}, and ML approaches have shown effectiveness in such forecasting tasks. Regression models identify mathematical relationships between dependent and independent variables, while DL models—particularly CNNs and RNNs have emerged as state-of-the-art techniques. CNNs excel in detecting spatial patterns through convolution operations, whereas RNNs are designed to capture temporal dependencies in sequential data using internal memory mechanisms \cite{paper8}. These DL approaches have demonstrated superior accuracy and adaptability in handling the complex dynamics of time series prediction for smart building energy demand. This research offers a framework of smart building EF using the latest technologies. The performance of the proposed model is compared to those of different ML, statistical, and DL models.

The key contributions of this study are outlined below:

1. A novel hybrid DL model combining Temporal Convolutional Neural Networks (TCN) and Gated Recurrent Neural Networks (GRU) is proposed for forecasting energy consumption in SBs.

2. The model is assessed for LTEF using two RNN variants—Temporal Convolutional Neural Networks (TCN) and Bi- directional LSTM (Bi-LSTM).

3. Performance of the proposed model is evaluated using a publicly available dataset and compared with other existing methods based on error metrics such as MAE, RMSE, and R^2 Score.

4.2 Theoretical Background

4.2.1 Temporal Convolutional Network (TCN)

A Temporal Convolutional Network (TCN) is a DL architecture designed for sequence modeling tasks. Unlike RNNs, TCNs use one-dimensional convolutional layers to process sequential data. The core concept is to apply stacked 1D convolutional layers with **causal convolutions**, ensuring that the output at any time step depends only on the current and previous inputs, thus preserving temporal order.

Key features of TCNs include:

- **Dilated Convolutions:** These allow the network to achieve a large receptive field without increasing depth, by skipping input values at defined intervals (e.g., every second or fourth value). This enables efficient modeling of long-range dependencies.

- **Residual Blocks:** TCNs incorporate residual connections, similar to those in ResNet, which help train deeper models and alleviate the vanishing gradient problem.
- **Normalization and Activation:** Each convolutional layer is followed by normalization techniques (e.g., batch normalization) and non-linear activation functions (e.g., ReLU), enhancing model stability and learning dynamics.
- **Causality:** Causal convolutions ensure that the model respects sequence order by avoiding the use of future time steps in predictions.

These features make TCNs highly effective for capturing complex temporal structures in sequential data, enabling stable training and robust forecasting performance.

A TCN is a type of NN architecture that utilizes 1D convolutional layers to process sequential data. TCNs operate by applying a sequence of 1D convolutions to the input sequence. Typically, a normalization and activation layer follow each convolution operation. Moreover, the convolutional layers often incorporate dilation, meaning they skip over some input time steps to expand the network's receptive field [60].

The activation matrix A at a given location t with filter F and dilation factor df is given as follows:

$$A(t) = \sum_{i=0}^{i-1} F(k)x(t - df.i) \tag{4.1}$$

More compactly Eq. 5.1 expressed as:

$$A(t) = (x *_{df} F)(t) \tag{4.2}$$

Here, i represents the size of the F, and the F size and df remain consistent within the module. Employing a limited number of layers and stacking multiple dilated convolutions assists in capturing temporal LTD through large receptive fields.

The residual block of a TCN comprises a rectified non-linear unit (ReLU), two operations of batch normalization, and two layers of dilated causal convolutions [60]. This block enhances the network's stability and learning efficiency when handling long time series with complex, stochastic characteristics.

4.2.2 Gated Recurrent Unit (GRU)

A Gated Recurrent Unit (GRU) is a type of Recurrent Neural Network (RNN) specifically designed to handle sequence data and capture temporal dependencies. GRUs introduce gating mechanisms that control information flow through the network, effectively addressing the vanishing gradient problem commonly associated with traditional RNNs.

The main components of a GRU include:

- **Update Gate:** Determines how much of the previous hidden state should be retained and passed to the current time step.
- **Reset Gate:** Controls how much of the past information should be forgotten when computing the new candidate activation.
- **Candidate Activation:** Combines the current input and the reset-modulated previous hidden state to propose a new value for the hidden state.
- **Hidden State Update:** The new hidden state is calculated as a weighted combination of the previous state and the candidate activation, controlled by the update gate.

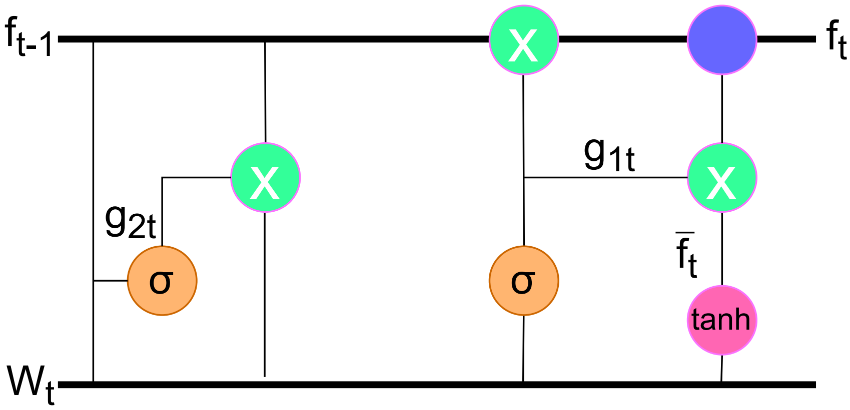


FIGURE 4.1: The block diagram of GRU.

Although GRUs are simpler than LSTMs due to having fewer parameters, they often achieve comparable performance. Their efficiency and effectiveness make them ideal for real-time prediction tasks involving irregular or noisy sequential data.

GRUs are a special kind of neural network made for working with sequences, like text or time-series data. They use clever mechanisms called **reset and update gates** to manage how information flows, helping them remember important patterns over long stretches of time.Figure 4.1 illustrates the overall structure of GRU.

Compared to LSTMs (their more complex cousins), GRUs are simpler and faster but still pack a similar punch in learning from sequences. They work step by step, keeping track of an internal "memory" as they go [61–64]. In the PM, the feature extraction process of TCNs yields a sequence $K = \{k_1, ..., k_n\}$, which is fed into the GRU. At time t, this process can be expressed using a non-linear function φ and GRU states h as follows:

$$h(t) = \varphi(h_{t-1}, k_t) \tag{4.3}$$

The following is an equation of the GRU's $\mathbf{g_1}_t$ and $\mathbf{g_2}_t$:

$$\mathbf{g_1}_t = \sigma(\mathbb{W}_{g_1}\mathbf{k}_t + \mathbb{U}_{g_1}\mathbf{f}_{t-1} + \mathbf{b}_{g_1}) \tag{4.4}$$

$$\mathbf{g_2}_t = \sigma(\mathbb{W}_{g_2}\mathbf{k}_t + \mathbb{U}_{g_2}\mathbf{f}_{t-1} + \mathbf{b}_{g_2}) \tag{4.5}$$

$$\tilde{\mathbf{f}}_t = \tanh(\mathbb{W}_f\mathbf{k}_t + \mathbb{U}_f(\mathbf{g_2}_t \odot \mathbf{f}_{t-1}) + \mathbf{b}_f) \tag{4.6}$$

$$\mathbf{f}_t = (1 - \mathbf{g_1}_t) \odot \mathbf{f}_{t-1} + \mathbf{g_1}_t \odot \tilde{\mathbf{f}}_t \tag{4.7}$$

where σ represents the sigmoid activation function, $\odot$ represents element-wise multiplication, $\mathbb{W}_{g_1}, \mathbb{W}_{g_2}$, and $\mathbb{W}_f$ is the input weight matrices for the $\mathbf{g_1}_t$, $\mathbf{g_2}_t$, and candidate activation, respectively. $\mathbb{U}_f$, $\mathbb{U}_{g_1}$, and $\mathbb{U}_{g_2}$ are the weight matrices for the previous hidden state, and $\mathbf{b}_{g_1}, \mathbf{b}_{g_2}$, and $\mathbf{b}_f$ is the bias vectors for the $\mathbf{g_1}_t$, $\mathbf{g_1}_t$, and candidate activation, respectively.

4.3 Proposed Methodology

The proposed approaches is inspired by the HM discussed in related works and the specific characteristics of the EC dataset. We aim to effectively leverage HLFI-based to enhance EC prediction outcomes. Hence, we propose a HM-based on the TCN and GRU. The advantages of both classifiers—TCN for feature extraction and GRU for LTD capture—are combined in this HM.

Sequential TS data modeling is widely employed across various disciplines. Therefore, we utilize data for model development in our approach. Let $X = \{x_1, ..., x_n\}$ represent an dataset with a time span of T between two consecutive x_i and x_j. This dataset X is fed into TCN to generate features for the data, and subsequently, GRU is employed to predict the $\widehat{y}$ value, which is then compared with the actual value y. In proposed model, we utilize data up to n samples to predict the value at $n + 1$.

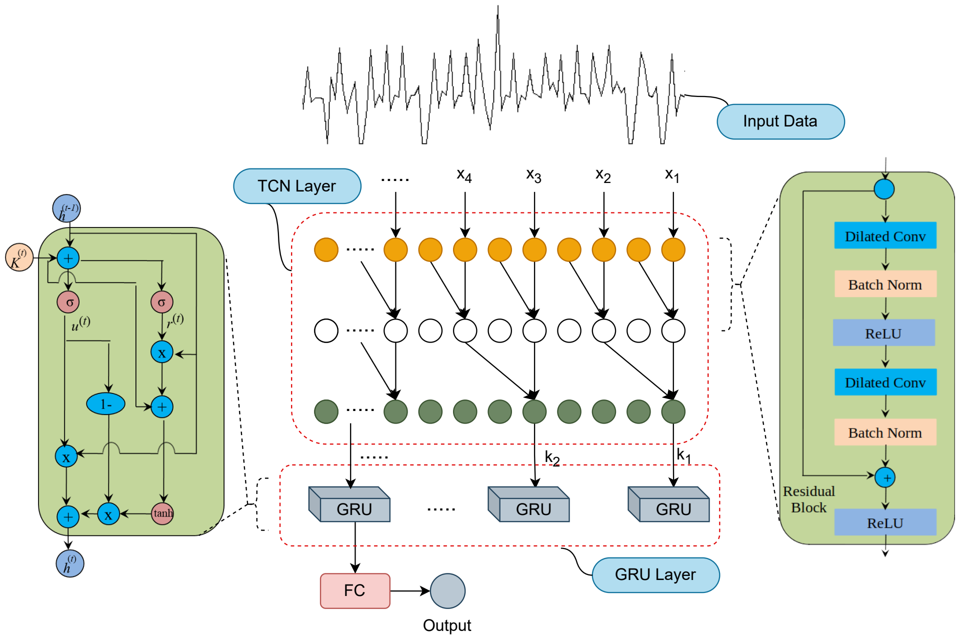


FIGURE 4.2: Proposed hybrid DL model for EC prediction.

4.4 Experimental Setup and Result Discussion

This section confirm the hybrid TCN-GRU model outperforms conventional methods, effectively capturing complex patterns and LTD, thus enhancing the accuracy and reliability of EC forecasting in SBs.

4.4.1 Results Evaluation

The suggested approach uses time series analysis to forecast new data based on prior data. Time series analysis is applied in this research on both multivariate and univariate data to extract features and study the nonlinear correlations observed in the CU-BEMS dataset. The proposed CNN-RNN model predicts the hourly energy consumption of intelligent buildings and is contrasted with other ML, statistical, and deep learning models like RNN, LSTM, and GRU, as explained in this section. Figure 4.3 illustrates the energy consumption patterns of the building over 12 months. We use two resolutions of data: total building consumption summed from minute-resolution daily recordings, and floor level consumption generated directly from minute-resolution data. To address data gaps resulting in abrupt consumption dips, we applied preprocessing based on hourly average plug loads and weekly consumption profiles. Training of the model on hourly data identified clear weekly patterns - significantly lower energy consumption on weekends versus weekdays. Week day data was especially useful for prediction modeling because it had a regular, workday-like pattern of consumption. At these times, current hour energy consumption was the strongest predictive variable, which mirrored the rhythms of the building's operation. This temporal analysis method allows for more precise EF by reflecting both the macro-level consumption patterns and micro-level usage habits within various building zones.

This study offers a novel method for hourly energy consumption forecasting based on the design and validation of sophisticated time series models. The research involves identifying energy usage patterns from past data via univariate time series methodology to derive essential characteristics and reveal intricate nonlinear relationships between building energy consumption patterns. We introduce a new Temporal Convolutional Network-Gated Recurrent Unit (TCN-GRU) hybrid model tailored for smart building hourly energy consumption prediction that leverages the temporal processing capability of convolutional networks and sequential modeling ability of recurrent networks.

To stringently evaluate the performance of our considered model, we perform thorough comparative evaluations with other state-of-the-art deep neural network architectures routinely used in EF scenarios. The evaluation criterion utilizes several statistical measures such as MAE for assessing average prediction error deviations, R-squared (R^2) for determining quality of model fits, and RMSE for predicting accuracy measurements. These comparative assessments are reported consistently in Table 3.3 of the work.

The choice of benchmark models for comparison was guided by a thorough survey of recent literature [65, 66], with a view to DL architectures that have proven useful

TABLE 4.1: Comparison of various models with lag values.

Models	**Lag**				**Lag**				**Lag**			
	1hr	3hr	5hr	7hr	1hr	3hr	5hr	7hr	1hr	3hr	5hr	7hr
	MAE				R^2				**RMSE**			
DL models												
RNN	26.4812	22.5859	20.6686	24.8361	0.9449	0.951	0.9608	0.9463	38.0802	43.8174	37.0673	37.2911
LSTM	19.2864	17.0511	16.5322	19.5125	0.9660	0.9698	0.9707	0.9686	37.4651	35.3006	34.7824	36.0371
Bi-LSTM	19.1978	16.9246	17.0047	17.9249	0.9657	0.9697	0.9623	0.9602	37.6343	33.0691	33.8403	33.9047
GRU	19.1564	16.7523	**15.6934**	17.1493	0.9672	0.9697	**0.9791**	0.9691	37.4343	34.0567	**31.8484**	33.9473
TCN	19.3895	**16.4941**	17.9853	16.9378	0.9627	**0.9725**	0.9689	0.9704	39.2281	**33.5567**	35.8351	33.6691
Hybrid DL models												
CNN-RNN	23.2822	29.2368	24.5888	23.228	0.9509	0.9349	0.9472	0.9525	45.0346	51.8873	46.7127	44.2947
CNN-LSTM	25.8177	28.1079	24.0721	21.9569	0.9505	0.9379	0.9535	0.955	45.2209	50.6549	43.8433	43.0147
CNN-Bi-LSTM	22.3561	29.8469	23.1679	22.8387	0.9517	0.9368	0.9532	0.9553	44.6727	51.1254	43.9547	42.9621
CNN-GRU	23.5764	21.5623	26.2557	23.7094	0.9504	0.9573	0.9528	0.9548	45.2678	42.0084	44.1411	43.2277
CNN-TCN	25.1159	28.4134	23.1394	22.5039	0.9503	0.9382	0.9538	0.9553	45.3235	50.5461	43.6990	42.9940
TCN-RNN	18.4919	17.9783	20.4270	20.9887	0.9700	0.9676	0.9672	0.9663	35.1958	36.5801	36.8035	37.3311
TCN-LSTM	17.6472	16.9559	16.9458	17.2117	0.9712	0.971	0.9714	0.9719	34.4742	34.6204	34.3358	34.0659
TCN-Bi-LSTM	18.5853	18.2173	17.8256	19.0715	0.9707	0.9703	0.9695	0.9692	34.801	34.9896	35.4884	35.6515
TCN-GRU	16.8845	16.8846	**15.1422**	17.3518	0.9710	0.9705	**0.9808**	0.9707	35.7210	34.9040	**31.8484**	34.2659
Architecture details												
CNN: CNN layer = (Filter = 64, Kernel size = 1), Maxpooling layer = (Pool size = 3, Padding = Same)												
RNN: RNN Layer = (Hidden nodes = 125, Dropout = 0.2)							LSTM: LSTM Layer = (Hidden nodes = 100, Dropout = 0.2)					
Bi-LSTM: Bi-LSTM Layer = (Hidden nodes = 50, Dropout = 0.2)							GRU: GRU Layer = (Hidden nodes = 100, Dropout = 0.2)					
TCN: TCN Layer = (Filter =32, Kernel size = 5, Dilation = [1,2,4,8,16,32])							Other parameters: Learning Rate: 0.001, Optimizer: Adam, Loss: MSE					
Simulation envriornment												
All models were trained on a Google Colab[1] with Python3 and backend GPU provided by Google Compute Engine.												
The configuration includes a single-core Intel(R) Xeon(R) CPU @ 2.20 GHz.												

in analogous EF tasks. The comparative framework guarantees that our assessment yields significant observations regarding the relative performance strengths of the proposed TCN-GRU model in the context of available solutions for smart building EM.

This research shows the better performance of Temporal Convolutional Network (TCN)-based hybrid models (HMs) than the traditional DL methods for predicting building energy consumption. The developed TCN structure outperforms sequential DL models throughout, at a minimum 6.0% decrease in MAE. Compared to CNN-based hybrid models specifically, the TCN design offers even greater improvements, lowering MAE by at least 20.0%. These findings emphasize the model's increased ability to process HLFI as well as efficiently capture LTD in energy consumption trends.

A close inspection of model performance across various lag intervals shows consistent accuracy. At pivotal lag values of 3.0 and 5.0 hours, all TCN-based hybrid models (except TCN-RNN) retain superior performance, recording $\mathbf{R^2}$ scores averaging around 0.97. The TCN-GRU model is found to be the best performer with a maximum R^2 of 0.9808 at a 5.0-hour lag. Comparative results indicate consistent error bounds, and deviations in MAE and RMSE between TCN-based and sequential DL counterparts are close to 2.0 and 4.0, respectively, for all the lag values tried.

The predictive accuracy of the model is also attested to by the high correlation between real and predicted energy consumption values Figure 4.3. Interestingly, the TCN-based architecture has high precision even during times of high consumption volatility, correctly capturing both sudden spikes and declines in energy usage. This high performance under changing conditions is a testament to the reliability of the model over current methods.

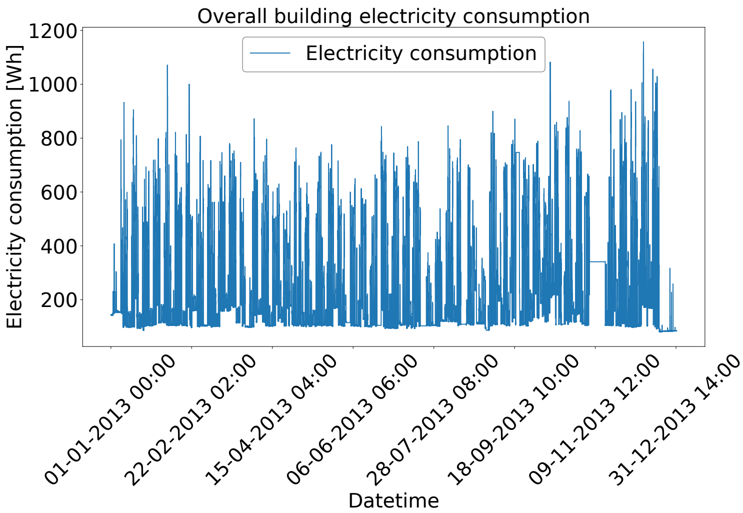
Still, the research indicates significant factors of consideration for deployment in real-world applications. Construction attributes, including size and spatial complexity, determine model scalability substantially. Successful deployment necessitates responsiveness to various architectural configurations, factoring in elements such as:

- Zone-specific usage patterns
- Equipment usage profiles
- Thermal phenomena in varying sections of a building

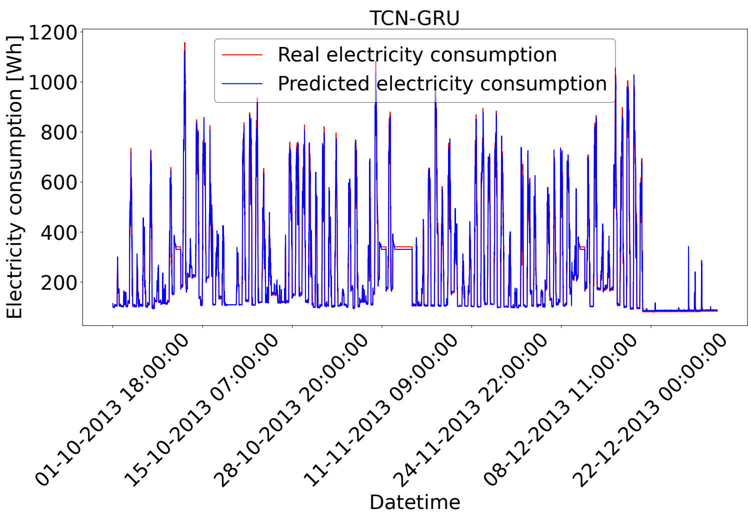
Geographical position also comes into play, since models need to factor in regional climatic patterns, local energy policies, and seasonal changes. These results indicate that although TCN-based models provide better baseline performance, the best accuracy for real-world use comes from customization to individual building conditions and environmental parameters. Future studies should concentrate on creating adaptive models that can automatically adapt to these changing parameters while retaining the model's essential predictive benefits.

4.4.2 Managerial Implications

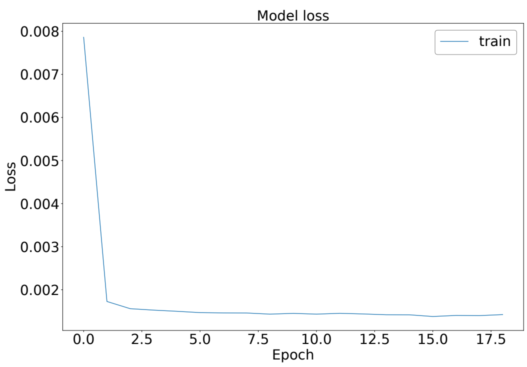
In the US, about 30.0% electricity used by residential appliances, producing associated 12.0% of CO_2 emissions. Artificial lighting and HVAC system accounts



(a)



(b)



(c)

FIGURE 4.3: (a) Overall building EC. (b) Actual v/s projected values using PM. (c) Loss per epoch graph of the PM.

for around 6.0% and 54.0% of energy usage in residential buildings. Similarly, HVAC and lighting systems use 40.0% and 15.0% of commercial buildings' energy [67]. Thus, for building managers and legislators, putting the suggested PM into practice in SB has several advantages:

- Precise EC predictions enhance resource allocation and LT planning.
- Demand response strategy become more promising, balancing demand and supply.
- Real-time EC forecasting allow occupant comfort, optimizing electricity efficiency and timely adjustments.
- Merging renewable energy sources is facilitated, reducing reliance on non-renewable.
- Enhanced EC management lowers costs and strengthens finances.
- Energy usage optimization, valuable insights and operational efficiency are achieved.
- CO_2 emissions and environmental impact are reduced, supporting sustainability goals and regulatory compliance.

Overall, the proposed PM empowers building managers to make data-driven decisions, optimize energy use, and support sustainable operations, balancing environmental responsibility and economic prosperity.

4.5 Summary

Sophisticated prediction models like the hybrid TCN-GRU approach revolutionize EM in SBs by enabling accurate electricity demand forecasts. These models optimize resource allocation, reduce costs, and maintain occupant comfort while capturing complex energy patterns for proactive decision-making. Their real-time prediction capabilities enhance demand response strategies, crucial for balancing supply amid growing renewable energy integration.

By improving energy consumption forecasts, these models significantly reduce CO_2 emissions, supporting global climate action. They empower building managers with data-driven insights for energy efficiency, cost savings, and regulatory compliance while facilitating renewable energy adoption. Given buildings' substantial energy footprint, such ML solutions are vital for sustainable urban development.

Future research should focus on geographic and climate-specific adaptations, incorporating environmental variables like temperature and daylight to boost accuracy. Intelligent forecasting tools will be pivotal in achieving sustainable operations, delivering both environmental and economic benefits for a greener future.