Journal of Operation and Automation in Power Engineering

vol. 14, no. Special Issue, Dec. 2026, Pages: 1-14

http://joape.uma.ac.ir

Research Paper

A Forecast-Driven AI–BC Framework for Intelligent Energy Trading in Sustainable SGs

Ilham Albakri and *Haider Th. Salim ALRikabi* *

Department of Electrical Engineering, College of Engineering, University of Wasit, Kut, Iraq.

Abstract— *The increasing deployment of Smart Grids (SG) has generated large volumes of real-time energy data, creating new challenges for accurate energy forecasting, dynamic pricing, and secure energy trading. Existing approaches often rely on predictive intelligence or static centralized trading mechanisms independently, limiting their ability to support adaptive and autonomous market operations. To address the above limitations, this paper proposes a multi-layer AI–Blockchain (BC) framework for intelligent energy trading in SGs. The proposed framework integrates the forecasting of three market parameters electricity demand, market sensitivity, and generator price cap with automated BC-based pricing and P2P energy trading through an oracle-enabled architecture. The proposed framework consists of three integrated layers: an IoT-enabled SG interaction layer, an off-chain intelligence layer, and an on-chain execution layer. A set of Deep Learning (DL) models is employed to predict electricity demand, market sensitivity coefficient (β), and generator price cap parameters using the SGRT-LMD dataset, which contains approximately 50,000 SG records. Experimental results demonstrate high forecasting accuracy. For electricity demand prediction, the CNN–MHA–BiLSTM model achieved an R^2 of 0.9925 and an MAE of 0.2097. For β, the same model achieved an R^2 of 0.9654 and an MAE of 0.2955, while the TCN–GRU model achieved the best generator price cap prediction performance with an R^2 of 0.9808 and an MAE of 0.0123. BC evaluation demonstrated efficient execution, with the maximum observed gas consumption reaching approximately 224,150 gas units.*

Keywords—Blockchain, smart grid, deep learning, dynamic pricing, energy trading, smart contracts.

1. INTRODUCTION

Contemporary SGs are changing into networked and informed energy systems, in which various forms of generation and renewable energy sources as well as active consumers constantly affect the operation of the system and the development of markets This change leads to new opportunities in the field of energy management but also to the need for accurate forecasting, adaptive pricing, and secure trading processes [1, 2].

1.1. Research motivation

The increasing integration of renewable energy sources and IoT-enabled technologies into SGs has introduced challenges in balancing electricity supply and demand, requiring intelligent energy management and adaptive pricing mechanisms [3]. DL techniques enable accurate forecasting of dynamic market conditions by capturing complex temporal patterns in energy data. However, forecasting alone cannot ensure effective market operations without mechanisms that translate predictions into pricing and trading decisions [4] Meanwhile, centralized energy trading systems face challenges related to transparency, trust, and intermediary dependence [5]. BC technology offers decentralized and automated transaction execution through smart contracts, but lacks the predictive intelligence required for adaptive market decisions [6]. Therefore, this study proposes integrating DL-based market forecasting with BC-enabled dynamic pricing and decentralized energy trading to establish a unified, automated decision-making framework.

1.2. Literature review

Agrawal [7] an open benchmarking framework that uses the BQEB-Data v1 dataset for assessing AI forecasting models in Smart Grids is introduced. Three AI models (namely Linear Regression, Random Forest, and Gradient Boosting) are compared for electricity load prediction in the next hour and for electricity price prediction for the next day according to a pre-defined chronological evaluation framework. In the experiments, Linear Regression each time yielded the minimum RMSE values measured in both tasks with R^2 values corresponding to 0.9288 and 0.2078 that means it is harder to predict prices compared to electricity loading in the conducted analysis. Wen *et al.* [8] introduced a hybrid DL model that integrates Gated Recurrent Units (GRU), Temporal Convolutional Networks (TCN), and an attention mechanism specifically for electricity load forecasting in the short term. The system was tested using four different datasets including GEFCom2014, ERCOT, AEMO, and NYISO and it demonstrated great forecasting accuracy, processing speed, and reliability. The main drawback of the work is that the authors concentrated only on electricity load forecasting and did not integrate forecast data into BC-based dynamic pricing or decentralized energy trading solutions. Kotla *et al.* [9] put forward a smart energy management technique that incorporates opponent reinforcement learning along

Received: 16 Oct. 2026
Revised: 20 Nov. 2026
Accepted: 21 Nov. 2026
*Corresponding author:
E-mail: hdhiyab@uowasit.edu.iq (H.Th.S. ALRikabi)
DOI: 10.22098/joape.2026.21310.2668

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.
Copyright © 2026 University of Mohaghegh Ardabili.

with the Dynamic Grey Wolf Optimization. It has been observed that the development coordinated electric vehicle charging their use of renewable energy and energy exchange thus helping achieve higher efficiency in work and economic performance in renewable-integrated SGs. However, the main focus of the work was on energy efficiency and optimization rather than the incorporation of several predicted market indicators into BC-based dynamic pricing. Shirke *et al.* [10] created a system for energy trade that included the application of Nash equilibrium principles as well as Hopfield neural networks. Their objective was to design a fair energy allocation system and optimize energy distribution among energy suppliers and users in the market. Even though their work dealt with questions of efficiency in the market and fair distribution of energy, they did not create an integrated approach based on the use of multi-parameter forecast systems and BC trading implementation expertise. Umar *et al.* [11] proposed a decentralized community energy management framework employing BC smart contracts to support demand response and peer-to-peer (P2P) energy trading. Their approach incorporated supply–demand-based pricing and automated transaction execution to improve market participation and coordination among distributed energy resources. Nevertheless, the pricing mechanism was not based on the combined DL prediction of multiple market parameters. Dong *et al.* [12] introduced a peer-to-peer cryptocurrency energy trading model that uses smart contracts with the Supply-Demand Ratio (SDR) and game theory as the basis for dynamic pricing of electricity. His technique was meant to optimize the use of electricity, omit the intermediaries, and yield economic profits to market players. Nonetheless, the main drawback of this method is that it uses fixed rules instead of using DL-predicted demand, market responsiveness, and generator price limits.

Su *et al.* [13] created a power trading system based on blockchain technology used in conjunction with Ethereum smart contracts, Power Order Matching Algorithm (POMA), and Proof of Credit Scores (PoCS) consensus method. The goal of this approach was to make transaction security, order matching, and trading efficient in distributed electricity markets better. Experimental findings confirm that the experimental system is good for transaction processing and efficient energy consumption. However, the framework primarily addressed BC trading and consensus mechanisms without incorporating DL-based multi-parameter forecasting into adaptive electricity pricing. Badar and Al-Khalidy [14] looked into the use of BC and machine learning (ML) technologies to enhance safe energy transactions in smart grids. Their system was based on intelligent data-driven approaches and decentralized transaction systems in order to improve reliability, security, and trust in energy systems. Nevertheless, the authors of this study did not propose a unique pricing scheme that would rely on the simultaneous prediction of electricity demand, market sensitivity, and price limits of power producers. Uddin *et al.* [15] proposed an IoT-enabled framework for virtual power plants (VPPs) that integrates edge–fog computing, blockchain (BC) technology, and AI-based energy management. The proposed framework supports coordinated electricity supply, demand response, secure communication, and participation in electricity markets. However, although the study integrates intelligent energy management with BC-based coordination, it does not incorporate an oracle-mediated dynamic pricing mechanism that utilizes forecasted market parameters. Singh *et al.* [16] proposed a BC-integrated AI framework to improve energy efficiency and sustainability in SGs. The proposed approach employed AI-based predictive analytics for electricity demand forecasting and energy distribution, while BC technology supported secure and decentralized energy management through smart contracts. Although the framework combined predictive intelligence with decentralized energy management, it did not establish a specific multi-parameter DL forecasting mechanism directly connected to on-chain dynamic electricity pricing. Sun and Weingärtner [17] proposed a decentralized energy trading architecture integrating AI-based forecasting, BC technology, and smart contracts to enhance power grid stability. Their framework introduced a Proof of Prediction mechanism to encourage accurate forecasting and improve energy management during periods of high demand. However, its primary emphasis was on forecasting incentives and grid stability rather than integrating the joint prediction of electricity demand, market sensitivity, and generator price caps into an oracle-mediated dynamic pricing mechanism. Kothuru *et al.* [18] presented a hierarchical multi-agent framework combining graph transformer networks, forecasting, multi-agent reinforcement learning, and BC-based coordination for decentralized demand response and P2P energy trading. The proposed architecture integrated predictive intelligence, operational decision-making, and secure transaction execution to improve energy coordination and system efficiency.

However, unlike the proposed work, it did not specifically incorporate the joint prediction of electricity demand, market sensitivity coefficient, and generator price cap into an on-chain dynamic pricing mechanism. Collectively, these studies demonstrate significant progress in intelligent forecasting, decentralized energy management, dynamic pricing, and BC-based electricity trading. However, the reviewed approaches differ in their forecasting targets, pricing mechanisms, and integration of predictive intelligence with transaction execution. These distinctions motivate the development of a unified framework that connects DL-based market parameter forecasting with oracle-mediated dynamic pricing and smart-contract-based P2P energy trading.

1.3. Research gap

The integration of DL-based forecasting and BC-based energy trading has received increasing attention. But continues to face several hurdles in the energy segment of SG [5, 19]. As highlighted in the presented literature overview, various techniques exist for forecasting the market, optimizing it, and executing decentralized transactions [8, 20]. However, the fact that forecasting is accurate is not enough for effective decision-making in the market, while the execution of transactions through the use of BC cannot bring in information that is necessary for pricing adjustment. There are three interrelated gaps that provide rationale for this research. The first one is the joint forecasting of multiple pricing-related variables, which is still challenging. Electricity demand D(t), market sensitivity β(t) and generator price limits P_{gen}(t) shape the process of dynamic pricing but their joint forecasting and integration into a unified pricing and trading mechanism remain insufficiently explored in the reviewed literature [20, 21]. Second, the relationship between the off-chain forecasting and on-chain execution requires adequate synchronization. The absence of an oracle that would transmit forecasting results to smart contracts prevents from fully utilizing the possibilities of DL-based market intelligence in automated pricing and trading processes. Finally, forecasting, dynamic pricing and BC-based P2P trading are not always performed as a single integrated process, which limits the application of predictive intelligence in transparent and automated transactions. To address these research gaps, this study proposes an integrated AI–BC framework that connects multi-parameter DL forecasting with oracle-mediated dynamic pricing and smart-contract-based P2P energy trading [5, 22]. Fig. 1 illustrates the identified research gaps and their relationship to the proposed framework, while Table 1 compares the proposed approach with the reviewed studies.

1.4. Novelty and main contributions

This paper proposes an integrated AI–BC framework for intelligent energy trading in SGs, addressing the limitations of multi-parameter market forecasting and centralized energy trading. Unlike existing approaches that integrate AI and BC primarily for energy forecasting, optimization, or transaction coordination, the proposed framework directly incorporates three DL-predicted

Table 1. Comparative analysis of related studies and the proposed AI–BC framework.

Ref.	Main research focus	AI / Forecasting / Decision mechanism	BC-based execution
[9]	EV charging and renewable-energy trading	Adversarial RL and evolutionary optimization	Not implemented; proposed for future integration
[14]	SG stability and secure energy transactions	ML-based grid stability prediction	Conceptual BC integration; on-chain trading not demonstrated
[11]	Decentralized community energy management	Demand response and SDR-based dynamic pricing	Smart-contract-based energy management and trading
[15]	IoT-enabled virtual power plants	AI-assisted market bidding and energy optimization	BC-secured communication and coordination
[18]	Hierarchical multi-agent energy trading	Graph-transformer forecasting and multi-agent RL	BC-coordinated P2P trading and settlement
The proposed work	**Forecast-driven decentralized energy trading**	**DL prediction of D(t), β(t), and**	**Oracle-mediated on-chain pricing and smart-contract trading**

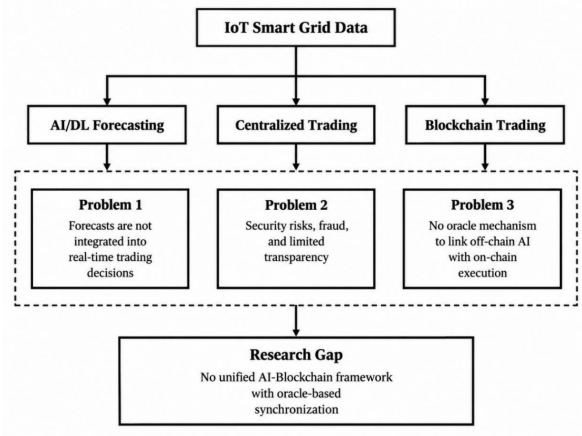

Fig. 1. Research gap and key limitations identified in existing SG energy trading systems.

market parameters into an on-chain dynamic pricing mechanism for decentralized P2P energy trading. The main contributions are summarized as follows:

- A DL-based forecasting solution that independently predicts electricity demand D(t), market sensitivity coefficient β(t), and generator price cap P_{gen}(t). Ten DL architectures are evaluated across the three forecasting tasks, and the selected models provide the predicted parameters required for adaptive dynamic pricing.
- A BC solution on the premise of having two smart contracts that are connected with each other, Pricing Contract and the Energy Trading Management Contract. The Pricing Contract works by computing the electricity pricing based on the anticipated market conditions as well as the available supply, while the Energy Trading Management Contract will carry out the functions of purpose of participant's verification; orders, the trade process and payment transactions circumventing a centralized trade.
- An architecture consisting of three layers that connects the IoT-enabled SG data acquisition and off-chain DL forecasting with on-chain market execution via an authorized oracle. The essence of the technical contribution is in linking the three market parameters predicted by the model into a dynamic pricing mechanism with a defined price ceiling for the generators that facilitates pricing based on the forecast and automated P2P trading.

1.5. Organization of the paper

The remainder of this paper is organized as follows: Section 2 presents the proposed AI–BC framework, including its three-layer architecture and dynamic pricing mechanism. Section 3 describes the dataset, DL model configurations, training procedures, smart contract implementation, and experimental setup. Section 4 presents the forecasting results, BC performance evaluation, and scenario-based analysis of dynamic pricing and energy trading. Finally, Section 5 concludes the paper by summarizing the main findings, limitations, and future research directions.

2. PROPOSED FRAMEWORK AND ARCHITECTURE

The proposed research presents an integrated AI and BC architecture for intelligent, decentralized, and adaptive energy management in IoT-enabled SG systems. The proposed architecture combines DL-based time-series forecasting with BC-based transaction management, thereby linking data-driven market prediction with decentralized pricing and trade execution. The architecture comprises three closely interconnected layers:

- An IoT-enabled SG interaction layer,
- an off-chain intelligence layer, and
- an on-chain execution layer.

This hierarchical structure ensures modularity, scalability, and data consistency, spanning from raw measurements and predictive models to final market execution via BC.

2.1. System architecture overview

The AI–BC framework suggested has three interconnected layers as depicted in Fig. 2. Each trading period (t), the IoT-based layer gathers operational SG data and gets energy supply offers and demand requests from market players. The off-chain intelligence layer analyzes this data and uses DL models to forecast the market parameters needed for dynamic pricing. Next, the predicted parameters and the pertinent market information are communicated to the on-chain execution layer where smart contracts carry out the functions of dynamic pricing, trading energy, and executing transactions.

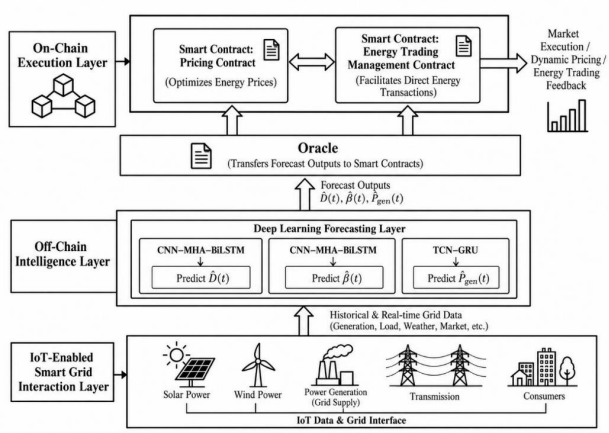

Fig. 2. Overall architecture of the proposed AI–BC framework for intelligent energy trading in IoT-enabled SGs.

A) IoT-enabled SG interaction layer

The IoT-supported interaction layer of the smart grid acts as a connector to the physical smart grid along with market actors and energy trading systems that rely on BC technology. The interaction layer collects real-time data on smart meters, green energy generators as well as other IoT-enabled devices, including electricity consumption data, generation data, and information related to electricity market conditions and environment. These measurements are transmitted to the off-chain intelligence layer

for preprocessing, feature engineering, and DL-based forecasting of electricity demand, market sensitivity coefficient β(t), and generator price cap.

Simultaneously, producers and prosumers submit energy supply offers, while consumers submit demand requests for each trading interval. This market information is forwarded to the on-chain execution layer, where smart contracts enforce participant verification, pricing, and transaction rules [8]. Formally, the SG input sequence at time slot t is represented as:

$$X_t = \{x_{t-w+1}, x_{t-w+2}, \ ..., \ x_t\} \tag{1}$$

where X_t historical input sequence available at time t, w is the length of the sliding window, defining the observation horizon, and x_t is the multivariate feature vector with all the SG measurements available at time period t. Participant market interaction at time slot t is represented by supply bids E_{it} from producers and prosumers and demand requests R_{jt} from consumers.

B) Off-chain intelligence layer

The off-chain intelligence layer is the analysis layer of the framework given above. In this mathematical formulation of the proposed framework, DL models are used to analyze data from the SG and predict the market conditions needed for the pricing mechanism to operate. Specifically, the off-chain intelligence layer is used to calculate the values of D(t), β(t), and P_{gen} based on the time-series data coming from the IoT-enabled SG interaction layer. The predictions obtained from this process are then used as inputs for the on-chain execution layer. Following prediction, the generated outputs are organized into a structured data object and transmitted to the on-chain execution layer via an authorized oracle mechanism. This division facilitates the off-chain DL inference process while permitting the on-chain integration of the resultant market predictions into pricing and trading logic. The specific forecasting models used for the three prediction tasks are outlined in the paragraphs below.

C) On-chain execution layer

Blockchain's execution layer takes into account the parameters of the market predicted by the artificial intelligence layer and combines them with various inputs relating to the demand and supply of the product to generate smart contracts and price the energy. Additionally, it validates the eligibility of its users and checks transaction details according to the previously established procedure. All transactions are written on the Blockchain ledger, which provides the necessary level of transparency, traceability and protection against forgery of the information. To maintain the required level of security of the layer, governments and management organizations employ access control systems that give rights to authorized users only. This layer works with two smart contracts, which are connected to each other, namely – the Pricing Contract and the Energy Trading Management Contract.

- **Pricing contract:** Is the pricing part of the On-Chain Execution Layer. It takes in three market parameters computed by DL models via an authorized oracle: electricity demand, market sensitivity coefficient, and price cap of a generator. These parameters along with the existing electricity supply and the fundamental electricity price will allow computing the dynamic price of the electrical unit in any particular trading interval. The contract uses owner-based access control for administrative operations and market data updates. Thus, its main functions are setWriter for setting authorized data writers, setSlotInputs for updating market data, setSlotPgenCap for setting the expected generator price cap, getUnitPrice for obtaining the calculated electricity price, and getTotalPriceWei to get the value of the whole transaction based on the amount of electricity used.
- **Energy trading management contract:** The energy trading management contract governs the processes of participant registration, verification, order placement, trade execution, and settlement in the proposed P2P energy trading system. This contract connects with the pricing contract to obtain the changing electricity price for each eligible transaction. The access to the contract is administratively limited through authorized accounts that are used for registration and verification of participants and diversion of the contract operations. The contract has its own registration functions for consumers, producers, and prosumers, whereas the functions 'setVerified' and 'blacklist' are responsible for monitoring participant status. Participants send requests for electricity consumption and supply bids for the corresponding trading period with the help of the submission functions supported by the contract. The functions 'placeOrder' and 'acceptOrder' are responsible for matching orders and processing the transactions while settlement functions are responsible for dealing with energy trade payments and records of completed transactions. The oracle mechanism is used to update the price data on a periodic basis and therefore integrate the pricing information created by the off-chain intelligence layer into the on-chain trading process. Fig. 3 illustrates the interaction between the Pricing Contract and the Energy Trading Management Contract. The oracle transfers the forecasted demand, market sensitivity coefficient, and generator price cap to the Pricing Contract. The Energy Trading Management Contract provides the aggregated market supply, which is used alongside the predicted parameters to calculate the dynamic electricity price. The final price is constrained by the predicted generator price cap and returned to the Energy Trading Management Contract for trade execution and payment settlement.

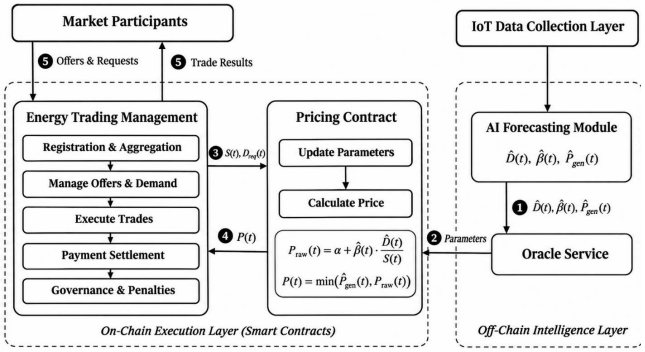

Fig. 3. Interaction between the energy trading management contract and the pricing contract.

2.2. System model

A) Electricity demand forecasting model

The problem of forecasting electricity demand is treated as a supervised regression issue defined as a task of predicting the future electricity consumption on the basis of historical multivariate SG data. During the forecasting process, different DL models are used that can reveal intricate nonlinear correlations and patterns taken from various electrical, environmental, and operational parameters, such as historical energy consumption, renewable energy production, and information about voltage, current, temperature, and humidity. By using these historical measurements, these modeling techniques enable capturing different variations in electricity demand under diverse grid conditions.

In the proposed framework, electricity demand is represented by D(t), with the Power Consumption variable serving as the prediction target. The DL model analyzes historical time-series observations to generate the forecasted demand $\hat{D}$(t) for the corresponding trading interval. This predicted value is subsequently

transmitted through the authorized Oracle to the blockchain-based Pricing Contract, where it is combined with the predicted market sensitivity coefficient $\hat{\beta}$(t), actual available energy supply S(t), and predefined base price α to calculate the dynamic electricity price. Therefore, demand forecasting serves as an essential component of the proposed framework by providing predictive information that supports adaptive pricing and decentralized P2P energy trading [2].

B) Market price sensitivity prediction

The β(t), represents the responsiveness of the electricity price to prevailing supply and demand conditions. Instead of treating β(t) as a constant parameter throughout the simulation, the proposed framework represents it as a time-varying quantity derived from the pricing formula and relevant market state variables.

The historical β(t) is stated to be derived from analysis of the electricity price in the market, demand for electricity, available supply in total as well as the base price. The values obtained using this method are included in the dataset for the market sensitivity model. By means of historical data regarding SG, the model deduces the time-dependent relations linked with variation of market sensitivity under different operational conditions. Thanks to the application of this method, establishing the price becomes conditional on the changing supply and demand conditions, and no fixed value of sensitivity is to be used. The next stages of the analysis involve passing the value for $\hat{\beta}$(t) which is treated as an updated version of β(t) further to the on-chain pricing method and other forecasts on market parameters.

C) Generator price cap prediction

The cap on the generator price sets a limit on the dynamic electricity price, which prevents undue increases in price due to the different market conditions. The cap is in effect when the calculated raw electricity price goes over the anticipated cap for the generator price, no matter whether there is a shortage or surplus in the market. Within the new approach, the cap of generator price is viewed independently so that the DL model could forecast the value with the use of necessary historical data as well as other relevant elements of the market. Once the cap is forecasted, it will be sent to the practical layer of the execution on the blockchain through the use of the oracle process. The approach helps to adapt to changes of prices in the market while giving an assurance that the final selling price will not go beyond the forecasted price cap for the generator.

2.3. Mathematical model of the pricing mechanism

The framework that has been suggested makes use of BC and DL methodologies to calculate the necessary market variables needed to achieve adaptive electricity pricing and decentralized trading in energy. For each trading time interval t, the *off-chain* intelligent layer processes the SG multivariate input stream X_t and creates forecasts for electricity demand, market price responsiveness, and the generator pricing limit. Then the predicted data is sent to the *on-chain* executing layer so that the pricing contract can utilize it in making price decisions for electricity. The electricity demand forecasting model is formulated as:

$$\hat{D}(t) = f_D(X_t) \tag{2}$$

where $\hat{D}$(t) is the predicted electricity demand at time interval t, $f_D(.)$ denotes the pre-trained DL-based demand forecasting model, and X_t is the SG multivariate input sequence.

1) Historical construction of the market sensitivity target

The model proposed here does not apply a fixed sensitivity coefficient but instead defines price sensitivity as a dynamic value. The first step to create the target for this model is to rearrange the pricing equation with the help of past data on electricity prices, demand, and supply. Using this equation, the historical sensitivity value can thus be derived.

$$\beta_{\text{obs}}(t) = \frac{[P_{\text{obs}}(t) - \alpha]\, S(t)}{D_{\text{obs}}(t)} \tag{3}$$

P_{obs}(t) denotes the historical electricity price, D_{obs}(t) represents the observed power demand, and S(t) denotes the total available supply, calculated as the sum of solar, wind, and grid supply measurements. The base price α is defined as the 10th percentile of the electricity price distribution in the training dataset.

The suffix "obs" differentiates between historical observations used to develop the target and model predictions. The historical values of β are used as the target, while the current value of the Power consumption used to derive β is not included in the inputs of the model to obtain the proper results. The β model is then expressed as:

$$\hat{\beta}(t) = f_\beta(X_t) \tag{4}$$

where $\hat{\beta}$(t) is the predicted market price sensitivity, and $f_\beta(.)$ denotes the trained DL prediction model. During actual market operation, $\hat{\beta}$(t) rather than the historically derived $\beta_{obs}(t)$(t) equation value is fed into the *on-chain* pricing mechanism.

2) Generator price cap prediction

The generator price cap is introduced as an upper-limit parameter for the dynamic pricing mechanism under varying market conditions.

$$\hat{P}_{\text{gen}}(t) = f_P(X_t) \tag{5}$$

where $\hat{P}_{gen}$(t) denotes the predicted generator price cap and $f_P(.)$ equation represents the corresponding prediction model. Subsequently, the predicted cap is transmitted to the Pricing Contract via the oracle mechanism described in Section 3.6.

The base price parameter is obtained from the lower tail of the training price distribution, as:

$$\alpha = P_{10}(\text{price}_{\text{train}}) \tag{6}$$

Where $P_{10}(.)$ denotes the 10th percentile operator and $price_{train}$ equation represents the electricity price observations in the training dataset. Thus, α provides a data-derived reference price for price formulation, rather than being treated as an arbitrarily selected constant [4].

3) Dynamic electricity pricing

The total electricity demand presented in time interval t is calculated as:

$$D(t) = \sum_{j=1}^{M} R_j(t) \tag{7}$$

where R_j(t) is the electricity demand submitted by the consumer. j and M is the number of active consumers. The forecasted demand is used to determine the dynamic electricity price, whereas actual participant demand requests are used for order validation and trade execution. Individual orders are executed only when the requested quantity is available from the selected seller, while aggregate market demand may remain partially unfulfilled under supply shortages. Using the predicted market sensitivity and demand together with the available supply, the unconstrained price is calculated as:

$$P_{\text{raw}}(t) = \alpha + \hat{\beta}(t)\frac{\hat{D}(t)}{S(t)} \tag{8}$$

Eq. (8) establishes the principal supply–demand relationship of the proposed pricing mechanism. For a given base price and sensitivity, a higher predicted demand relative to available supply

results in a higher unconstrained price [4]. To prevent the final electricity price from exceeding the predicted generator price cap, the following upper-bound constraint is applied under all market conditions:

$$P(t) = \min\left\{\hat{P}_{\text{gen}}(t), P_{\text{raw}}(t)\right\} \tag{9}$$

4) Trading and settlement

Once the unit price has been determined, the cost associated with an accepted traded quantity. E_{trade} is:

$$C(t) = P(t)E_{\text{trade}}(t) \tag{10}$$

where C(t) is the transaction cost and E_{trade}(t) is the quantity of energy accepted for settlement at time slot t.

Following order acceptance, the remaining available supply is updated according to:

$$S_{\text{remaining}}(t) = S(t) - E_{\text{trade}}(t) \tag{11}$$

The corresponding seller payment is:

$$\text{Payment}_{\text{seller}}(t) = P(t)E_{\text{trade}}(t) \tag{12}$$

If the buyer initially transfers an amount Paid(t) exceeding the final settlement amount, the refundable amount is calculated as:

$$\text{Refund}(t) = \text{Paid}(t) - \text{Payment}_{\text{seller}}(t) \tag{13}$$

The connection between the dynamic unit price and the actual traded quantity and settlement process adopted in the Contract of Energy Trading Management is established by these equations. In the proposed system, the forecasted variables indirectly affect market clearing through the dynamic price of electric energy rather than making a direct impact on the process of energy allocation. In other words, an unconstrained price is produced based on expected demand and market sensitivity, and the forecasted generator price cap limits the ultimate unit price. The Energy Trading Management Contract subsequently uses this price to process eligible trading orders based on available seller capacity. Accepted quantities are settled at the calculated price, and the remaining supply is updated to prevent double allocation. Each order is executed in full only when the selected seller has sufficient available energy; otherwise, the order is rejected. Consequently, aggregate market demand may remain partially unfulfilled when total available supply is insufficient.

5) Enforcement state

The mechanism for enforcing rules is incorporated in Energy Trading Management Contract in order to control eligibility of participants and prevent unauthorized and restricted participants from carrying out transactions in energy. Each participant must be registered and verified, without any mention of the name in the blacklist to be eligible in P2P energy trading. The eligibility of a participant i at time t is given by the variable B_i(t).

$$A_i(t) = \begin{cases} 1, & \text{if participant } i \text{ is authorized to trade at } t, \\ 0, & \text{if participant } i \text{ is restricted at } t. \end{cases} \tag{14}$$

If a party breaches any market rules, the designated administrator has authority to execute the punishment by applying the function that blocks trading activity of the party in question. A participant subject to restrictions cannot carry out his trading activities until the restrictions have been removed by means of the clearBlacklist function. The mechanism for punishment operates independently of the calculation of pricing dynamics. Therefore, the issue of eligibility of the participants is crucial to their ability to execute trading operation, without any implications for the market indicators disclosure or calculation of the price of the electricity unit.

6) Penalty and enforcement mechanism

The Energy Trading Management Contract applies an enforcement mechanism that controls the eligibility of participants and forbids illegal trading. Participants must be defined, verified, and must not be banned if they intend to carry out P2P energy transactions. The value of a participant's eligibility status i at time t is denoted by B_i(t), which determines whether the participant has the right to trade. When a participant violates predefined market rules, an authorized administrator can invoke the punish function to impose trading restrictions. The clearBlacklist function allows an authorized administrator to remove these restrictions once the required administrative conditions are satisfied. This mechanism functions outside of the dynamic pricing process. Whether a transaction can occur is determined by who is eligible as a participant, while the electricity price is evaluated based on expected market variables all while taking the available energy into account. Hence, enforcement steps do not change the pricing formula or its anticipated parameters.

Algorithm 1: AI–BC pricing and energy trading cycle

Require: X_t (IoT data), S(t) (supply), α (base price), R_j(t) (demands), models (f,g,h)

Ensure: Dynamic price P(t), executed trades, ledger updates, and penalties

Phase 1: Off-Chain AI Parameter Prediction

1: Preprocess X_t via sliding-windo generation

2: Compute $\hat{D}$(t)←f(X_t), β(t)←g(X_t), and P_{gen}(t)←h(X_t)

3: Broadcast $\hat{D}$(t),β(t),P_{gen}(t) to the smart contract via the oracle

Phase 2: On-Chain Dynamic Pricing

4: Calculate raw price: P_{raw}(t)←α+β(t)×$\frac{\hat{D}(t)}{S(t)}$

5: Apply capping: P(t)←min(P_{gen}(t),P_{raw}(t))

Phase 3: P2P Trading and Blacklist Enforcement

6: **for** each participant order u with quantity E_{trade} **do**

7: **if** Banned(u)=1 **and** $T_{current}$≤BanUntil(u) **then**

8: Revert transaction // active ban penalty

9: **else**

10: Verify eligibility, grid conditions, and available supply S(t)

11: Deduct supply: S(t)←S(t)-E_{trade}

Phase 4: Settlement and Governance

12: $Payment_{seller}$←P(t)×E_{trade}

13: Refund←Paid-$Payment_{seller}$

14: **if** u violates trading conditions **then**

15: BanUntil(u)←$T_{current}$+$T_{penalty}$

16: Banned(u)←1

17: **end if**

18: **end if**

19: **end for**

20: Commit finalized P(t), trade states, and settlement logs to the ledger

3. IMPLEMENTATION AND EXPERIMENTAL SETUP

In this section, the implementation of the proposed AI-BC framework is described. The experimental setup, dataset description, data pre-processing pipeline, configurations of DL models, and BC deployment setup are described.

3.1. System implementation overview

The suggested framework distinguishes between the computationally intensive DL forecasting and the on-chain BC execution processes. The off-chain Python processes SG data and provides predicted market parameters while the Ganache shuts down a local Ethereum-compatible environment along with testing and deployment of Solidity smart contracts responsible for energy trading and dynamic pricing. The permissioned Python oracle has been created using Web3.py to transfer the parameters to deployed smart contracts through access control that allows only authorized accounts to perform slot-level updates. The hardware and software configurations used in the experiments are summarized in Table 2.

Table 2. Hardware and software specifications.

Category	Item	Specification
Hardware	Processor	Intel Core i9
Hardware	Memory (RAM)	32 GB
Hardware	Storage	1 TB SSD
Software	Programming language	Python 3.9+
Software	DL framework	TensorFlow 2.x
Software	Data processing libraries	NumPy, Pandas
Software	Visualization	Matplotlib, Seaborn
BC	Smart Contract Language	Solidity v0.8.19
BC	BC interaction	Web3.py v6.x
BC	Test environment	Ganache CLI
BC	Development framework	Truffle suite

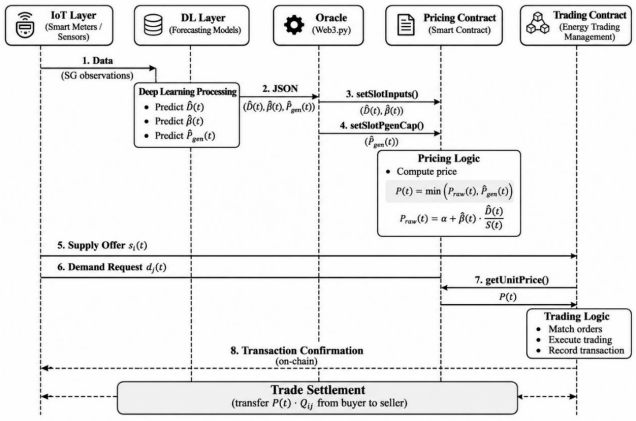

Fig. 4. Sequence diagram of the proposed AI–BC SG framework.

Fig. 4 illustrates the complete timeline of the proposed framework, starting with data generation in the SG environment. The collected data are processed by trained DL models to predict three parameters: D(t), β(t), and P_{gen}(t). These predictions are transferred through the oracle to the BC pricing contract, which also receives supply offers and demand requests from market participants. The pricing contract then calculates the electricity price and provides it to the trading contract, where energy buying, selling, and transaction settlement are executed.

3.2. Dataset description

The experiments were conducted using the publicly available SG Real-Time Load Monitoring (SGRT-LMD) data set [23, 24]. This data set has time-sequence records of the electrical, functional, environmental and market parameters, which are necessary for the prediction computations to be carried out in accordance with the proposed dynamic pricing model. A unified pre-processing and feature-engineering pathway was used for different forecasting problems, with the necessary site-specific modifications and measures taken in order to avoid leakage being beneficial.

The SGRT-LMD database consists of about 50,000 measurements recorded every 15 minutes. The list of available parameters includes: voltage, current, power consumption, reactive power, power factor, solar energy generation, wind energy generation, grid supply, voltage fluctuations, overload, faults in the transformer, temperature, humidity and electricity price.

3.3. Data preprocessing pipeline

As shown in Fig. 5, the entire experiment process, which runs from SG measurements to energy trading using BC technology, is complete. During this, many stages such as obtaining data, preprocessing it, feature engineering, DL model building, choosing the most suitable model, establishing communication based on the oracle, and execution of smart contracts are integrated to enable dynamic pricing and P2P energy trading. In the beginning, data related to electrical, operational, environmental, and market aspects must be derived from the SGRT-LMD dataset. Data must be organized chronologically concerning timestamps. Data must be cleaned during data processing so that all timestamps without valid information are deleted. Next, actual timestamps must be used to calculate meaningful parameters. Ultimately, data normalization must be performed using appropriate scaling methods reflecting only training data. After gathering all the data, feature engineering is performed using lagged observations and rolling statistics. In this way, a sliding window of 48 consecutive observations allowing to construct a sequence with 12 hours of past data with 15-minute accuracy is created. The length of the window means the number of observations per sequence but not the size of the initial dataset that comprises around 50,000 observations. Data is divided into training, validation, and testing according to a 64%, 16%, 20% ratio.

Table 3. Deep learning training hyperparameters.

Hyperparameter	Configuration
Optimizer	Adam
Loss function	Huber loss
Initial learning rate	0.0003 (3×10^{-4})
Batch size	64
Maximum epochs	100
Dropout Rate	0.20
Early stopping patience	20

The ten DL architectures (M01 to M10) have been developed and examined against three forecasting objectives: electricity demand, market sensitivity coefficient, and generator price cap. The optimal architecture for each forecast purpose has been determined for validation purposes. Therefore, the architecture CNN–MHA–BiLSTM (M09) has been chosen for forecasting electricity demand and market sensitivity and the architecture TCN–GRU (M06) for generator price cap forecasting. The two architectures yield the outputs of the forecast parameters D(t), β(t), and P_{gen}(t) to be applied in the proposed dynamic pricing solution. Table 3 consists of information about the training hyperparameters used in the evaluated deep learning models. The predicted results

are processed into a structured format and then passed to an authorized oracle based on Python with Web3.py.

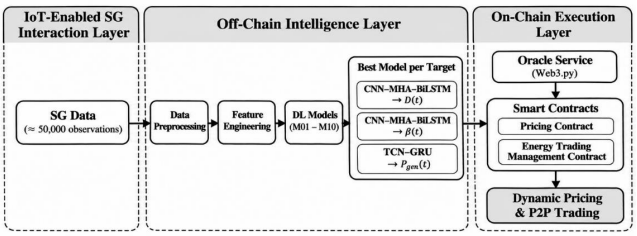

Fig. 5. Experimental implementation workflow of the proposed AI–BC framework, illustrating dataset preprocessing, feature engineering.

3.4. DL model configurations

A total of ten DL architectures (M01–M10) were trained and evaluated for three forecasting tasks: D(t), β(t), and P_{gen}(t). These architectures combine recurrent neural networks, convolutional layers, attention mechanisms, temporal convolutional networks, and Transformer-based techniques to capture nonlinear relationships and temporal patterns in SG data [8, 25–27].

The evaluated architectures include M01 (BiGRU–Transformer), which combines bidirectional gated recurrent units with Transformer-based attention to model temporal dependencies; M02 (BiLSTM–Attention), which integrates bidirectional long short-term memory networks with attention to capture temporal relationships; M03 (CNN–GRU) and M04 (CNN–LSTM), which employ convolutional layers for local feature extraction followed by recurrent units for temporal modeling; and M05 (PatchTST) [28], which employs patch-based time-series representations and Transformer-based attention to capture temporal dependencies. M06 (TCN–GRU) integrates temporal convolutional networks with gated recurrent units to capture temporal patterns, while M07 (TSMixer) employs MLP-based temporal and feature mixing to model time-series relationships [29]. M08 (GRU–MHA) uses multi-head attention in its GRU-based temporal modeling process. On the other hand, M09 (CNN–MHA–BiLSTM) applies convolution-based feature extraction and multi-head attention together with a bidirectional LSTM to learn local and long-range temporal dependencies. Lastly, M10 (Transformer Encoder) applies self-attention functions to the entire input signal without requiring recurrent processing.

The designs of all architectures have been done in TensorFlow 2.x in combination with Keras API. There has been an application of the identical source data set together with the feature engineering framework, the Min-Max normalization approach based on training data, and the chronological order of storing the data for training, validating, and testing only for the purpose of ensuring the appropriate experimental conditions. Besides, preprocessing for specific targets has also been employed wherever it has been needed. Dropout regularization with a rate of 0.20 was applied to selected network components, while early stopping monitored validation loss and restored the model weights associated with the best validation performance to reduce overfitting [30].

Forecasting performance was evaluated using two regression metrics: the coefficient of determination (R^2) and Mean Absolute Error (MAE). R^2 measures the proportion of variation in the actual target values explained by the model, with values closer to 1 indicating better predictive performance. MAE represents the average absolute difference between predicted and actual values, with lower values indicating smaller prediction errors. Hence, the architecture which obtained the best validation R^2 value was selected for every forecasting purpose. We used an independent test dataset to check the performances of the architectures selected. Random seeds were set in the process of training to allow the replicability of some stochastic operations. A summary of the ten DL architectures used in the forecasting of the three purposes is shown in Table 4.

Table 4. Deep learning model architectures and configurations.

ID	Model	Brief description
M01	BiGRU–Transformer	Combines bidirectional GRU with Transformer-based attention to capture temporal dependencies.
M02	BiLSTM–Attention	Models long-term temporal dependencies with attention-based weighting.
M03	CNN–GRU	Extracts local patterns using CNN and models sequential behavior using GRU.
M04	CNN–LSTM	Combines local feature extraction with long-term temporal modeling.
M05	PatchTST	Uses patch-based time-series representations and Transformer attention to capture temporal dependencies.
M06	TCN–GRU	Uses temporal convolutions followed by GRU to represent sequential patterns.
M07	TSMixer	Employs MLP-based temporal and feature mixing to capture time-series relationships.
M08	GRU–MHA	Integrates GRU with multi-head attention to capture diverse temporal relationships.
M09	CNN–MHA–BiLSTM	Combines local feature extraction, multi-head attention, and bidirectional temporal modeling.
M10	Transformer encoder	Uses self-attention to capture dependencies across the input sequence.

3.5. BC deployment and smart contract configuration

The BC component was implemented and evaluated in a local Ethereum-compatible network using Ganache CLI. Two smart contracts, the Pricing Contract and the Energy Trading Management Contract, were developed using Solidity v0.8.19, deployed through Truffle, and accessed via Web3.py [11]. After the deployment of the Pricing Contract and its registration with the Energy Trading Management Contract, the two contracts can function together. The Pricing Contract is an implementation designed to facilitate the process of managing dynamic pricing. As for the details, for each trade period, the Pricing Contract receives predictions made by the DL models on three parameters,These predictions are sent via a special oracle from the Python programming language, and are transformed into numbers used in Solidity. The amount of electricity available is found with the help of energy offers made by sellers and received through the Energy Trading Management Contract. After collecting the aforementioned data, the Pricing Contract starts calculating dynamic pricing by combining together predicted amounts, market sensitivity, available supply, and previously specified base price with the usage of maximum price cap determined by the generator. Moreover, access control is being applied, so that only authorized users can update the parameters based on predictions. Finally, Energy Trading Management Contract receives the unit price calculated.

The experiments took place on the controlled testbed Ganache, which allowed us to assess the execution of the smart contract and the gas cost involved. Nevertheless, this particular local environment does not replicate the situation of distributed consensus, network congestion, or network security conditions. In addition, we didn't implement the required transaction privacy protocols. As a result, what we have reported refers to some experimental results rather than demonstrating the security, scalability, and performance in practice.

3.6. Oracle mechanism and system integration

The oracle serves as a communication interface between off-chain DL models and on-chain smart contracts. During each trading interval, the oracle receives the predicted electricity demand, market sensitivity coefficient, and generator price cap, and transmits these parameters to the Pricing Contract through authorized Web3.py transactions [31]. The Pricing Contract combines forecasted demand and sensitivity with available supply in the market in order to assess the unconstrained price of electricity, which is subsequently limited by the forecasted capping price of the generator. The Energy Trading Management Contract takes the unit price it has obtained as well as orders from all participants for trading in order to establish their eligibility for trading, check the capability of the sellers to supply electricity, and execute the payment for accepted energy. Overall, the forecasted variables affect the dynamic price conditions, while the actual execution of orders depends on the eligibility of traders to perform transactions and the available supply offered by sellers. Thus, it is clear that off-chain predictive results have the ability to effect on-chain pricing and decentralized trading processes. The framework described above uses 15-minute trading intervals, meaning that this term indicates the resolution of scheduling, rather than verified execution time. The latency of DL inference, oracle communication, transaction processing, and completion of trading was also not calculated in this method. Although the oracle employs permissioned access control to restrict unauthorized updates, this mechanism does not independently guarantee the authenticity, integrity, or freshness of the off-chain forecasting data. Comprehensive oracle data verification, including timestamp validation and integrity-checking mechanisms, remains beyond the scope of the current implementation and is considered a direction for future work.

3.7. Simulation scenario design

A total of eight simulations were conducted to analyze whether the pricing mechanism meets its objectives. The simulations cover a range of supply and demand conditions, from excess supply and near-equilibrium situations to moderate, severe, and extreme cases of supply shortage. Every scenario has a fixed supply and demand. This information is included in the BC trading context for the purpose of studying pricing behavior, execution of transactions, and the impact of the generator cap price. The different scenarios and their respective operating conditions are presented in Table 5.

While scenarios S1 to S7 show the generic modes of functioning being explored in the conducted experiments, scenario S8 was specifically introduced to create a situation of stress-testing involving basic elements of supply shortage. These scenarios are to see if any reaction of the pricing model to changes in the market operation is evident, and whether the upper limit of the price for producers is effective in regulating the price under critical circumstances. Results acquired from running these scenarios on the BC platform are discussed in Section 4.5.

4. RESULTS AND DISCUSSION

The assessment of the suggested AI-BC system is done through three combined viewpoints: prediction correctness based on DL technology, BC efficiency and activity, and dynamic pricing characteristics in different markets. First of all, ten DL models were compared in terms of electricity demand, market reaction, and price limits for producers. After this procedure, the best-performing model according to the indicators mentioned above was chosen to be used in the proposed system. In the end, the last layer of the BC system was evaluated in terms of "gas" consumption in the smart contracts and a scenario-based evaluation of the results concerning dynamic pricing and energy market behavior was carried out.

4.1. DL model performance

Ten DL architectures were evaluated under identical conditions regarding dataset configuration, preprocessing procedures, data splitting, and training parameters. The models were assessed against three forecasting targets: D(t), β(t), and P_{gen}(t). To quantitatively measure forecasting performance, the R^2 , MAE were used. The full results are presented in Table 6.

The results in Table 6 indicate that forecasting performance varies across the three prediction targets. M09 (CNN–MHA–BiLSTM) achieved the best performance for D(t) and β, benefiting from CNN-based feature extraction, multi-head attention, and BiLSTM-based temporal modeling. In contrast, M06 (TCN–GRU) performed best for generator price cap prediction by combining temporal convolutional features with GRU-based sequential modeling. These findings highlight the importance of selecting suitable DL architectures for each prediction target rather than relying on a single model for all tasks.

A) Electricity demand prediction

For electricity demand forecasting, M09 (CNN–MHA–BiLSTM) achieved the highest R^2 (0.9925) and the lowest MAE (0.2097) among the evaluated architectures. M03 (CNN–GRU) demonstrated the second-best performance, with an R^2 of 0.9877 and an MAE of 0.2661, whereas M08 (GRU–MHA) exhibited the lowest predictive accuracy, with an R^2 of 0.9322 and an MAE of 0.6616. The superior performance of M09 may be associated with its combination of CNN-based local feature extraction, multi-head attention for capturing relevant temporal relationships, and BiLSTM for modeling sequential dependencies.

Fig. 6 presents the actual and predicted electricity demand values for the first 1,000 test samples, while Fig. 7 illustrates the training and validation curves of M09. The close agreement between the predicted and actual values provides visual support for the model's forecasting accuracy. Furthermore, dropout regularization and early stopping, as described in Section 3.4, were employed to reduce overfitting during training.

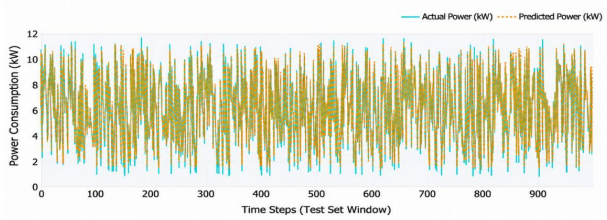

Fig. 6. Actual versus predicted power demand for the first 1000 test samples using the proposed CNN-MHA-BiLSTM model.

B) Market price sensitivity prediction

The β prediction exhibited greater performance variation across the evaluated architectures than electricity demand forecasting. As shown in Table 6, M09 (CNN–MHA–BiLSTM) achieved the highest R^2 (0.9654) and lowest MAE (0.2955), followed by M10 (Transformer Encoder) and M05 (PatchTST), with R^2 values of 0.9589 and 0.9522, respectively. These results support the selection of M09 for modeling the temporal variations in β.

Fig. 8 compares the actual and predicted β values over 300 test samples, showing close agreement with minor deviations during abrupt changes. Fig. 9 illustrates the training convergence of M09, with a minimum validation Huber loss of 0.0002 at epoch 53 and a minimum validation MAE of 0.0115 at epoch 65. The validation MAE of 0.0115 was obtained on the Min–Max normalized scale, whereas the test MAE of 0.2955 reported in Table 6 was calculated on the original β scale after inverse normalization. Overall, the results support the use of M09 for forecasting β within the proposed dynamic pricing mechanism.

C) Generator price cap prediction

As shown in Table 6, M06 (TCN–GRU) performed better than the evaluated architectures for predicting generator price caps with

Table 5. Simulation scenarios used for evaluating the proposed framework.

S	Condition	S	D	Description
S1	Supply shortage	600 kWh	818 kWh	Supply is lower than demand; cap activation is expected.
S2	Supply surplus	182 kWh	139 kWh	Supply slightly exceeds demand; stable pricing is expected.
S3	Severe shortage	213 kWh	626 kWh	Large deficit; unmet demand is predicted.
S4	Moderate shortage	87 kWh	237 kWh	Moderate deficit; partial fulfillment is anticipated.
S5	Mild shortage	112 kWh	125 kWh	Small deficit with high sensitivity; cap activation may occur.
S6	Near-balanced shortage	233 kWh	264 kWh	Demand slightly exceeds supply; moderate pressure is expected.
S7	Abundant supply	946 kWh	541 kWh	Supply surplus; low price pressure and full execution are expected.
S8	Extreme shortage	228 kWh	1332 kWh	Extreme demand pressure under severe supply deficit to evaluate system robustness and pricing stability.

Table 6. Performance comparison of DL models.

Model	Electricity demand		Market sensitivity (β)		Generator price cap	
Architecture	R^2	MAE	R^2	MAE	R^2	MAE
M01 BiGRU–Transformer	0.9373	0.6058	0.9357	0.3768	0.1021	0.09420
M02 BiLSTM–Attention	0.9382	0.5757	0.9056	0.3996	0.1378	0.09220
M03 CNN–GRU	0.9877	0.2661	0.9450	0.3297	−0.0063	0.09990
M04 CNN–LSTM	0.9871	0.2768	0.9518	0.3237	0.9465	0.01983
M05 PatchTST	0.9786	0.3520	0.9522	0.3604	0.9483	0.02042
M06 TCN–GRU	0.9729	0.4053	0.9437	0.4020	0.9808	0.01225
M07 TSMixer	0.9514	0.5593	0.9511	0.3738	0.8713	0.03651
M08 GRU–MHA	0.9322	0.6616	0.9263	0.3559	0.5489	0.06520
M09 CNN–MHA–BiLSTM	0.9925	0.2097	0.9654	0.2955	0.9680	0.01670
M10 Transformer encoder	0.9771	0.3731	0.9589	0.3610	0.9607	0.01864

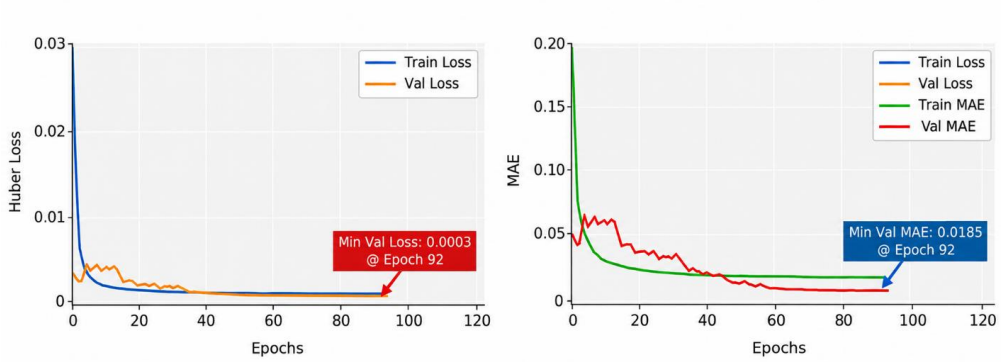

Fig. 7. Training dynamics of the proposed CNN-MHA-BiLSTM model for demand prediction: Loss (Huber) and MAE versus training epochs.

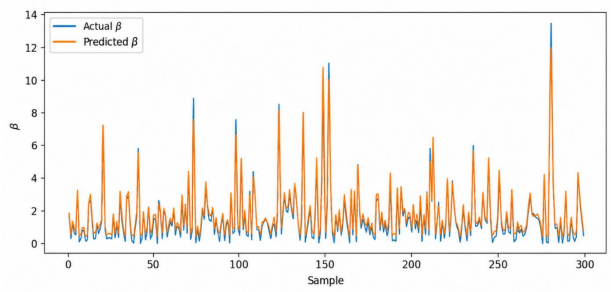

Fig. 8. Actual versus predicted historical β(t) values using the proposed CNN–MHA–BiLSTM model.

R^2 = 0.9808 and MAE = 0.0123. The use of temporal convolutions and GRU based sequential modeling might be able to capture the variations of the target variable. M09 (CNN–MHA–BiLSTM) also performed well (R^2 = 0.9680), while M02 and M03 had poor performance, with R^2 values of 0.1378 and -0.0063, respectively.

Table 7. Comparison with prior work on the SGRT-LMD dataset.

Study	Model	Dataset	R^2
[32]	BiTCSAtt–TCN	SGRT-LMD	0.9640
The proposed work	CNN–MHA–BiLSTM	SGRT-LMD	0.9925

These differences make the case for an independent forecasting architecture for each target even stronger.

Fig. 10 compares the actual and predicted generator price cap values for the first 1,000 test samples. The two values are close to each other with small deviations at the points of sharp transitions. The training convergence of M06 is shown in Fig. 11, where the minimum validation Huber loss (∼0.0009) and validation MAE (0.0315) are achieved at epoch 98. The convergence trends suggest stable training with no apparent visual evidence of significant overfitting. Overall, these findings support the use of M06 to generate the price cap predictions used in the proposed dynamic pricing mechanism.

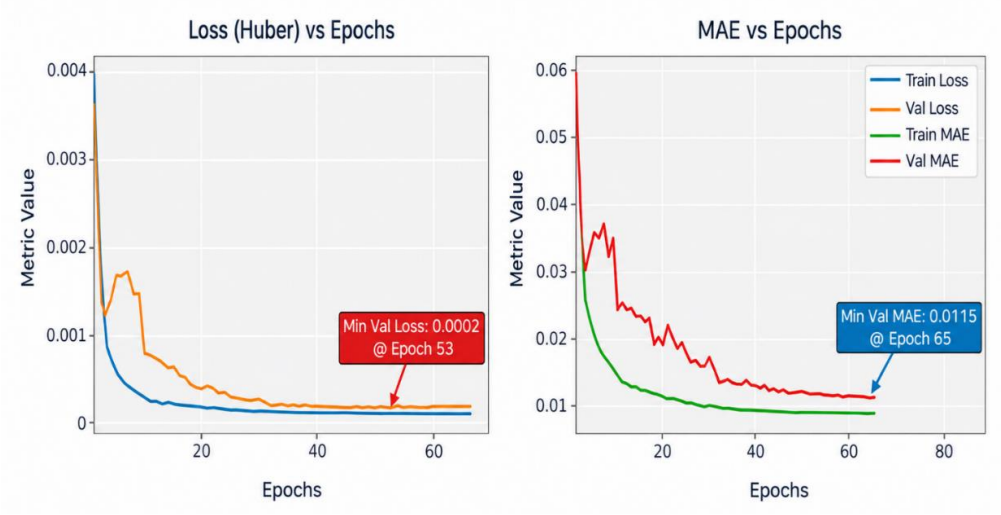

Fig. 9. Training dynamics of the proposed CNN–MHA–BiLSTM model for parameter β(t) prediction.

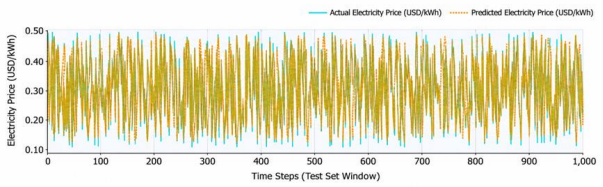

Fig. 10. Actual versus predicted P_{gen} values for the first 1000 test samples.

Table 8. Approximate Gas consumption of smart contract operations.

Operation	Approximate gas used	Interpretation
Participant registration	49,000	Lightweight; one-time per participant
Participant verification	31,000	Lowest cost; simple state update
Offer submission	50,000–117,000	Variable; depends on slot data size
Demand submission	50,000–80,000	Variable; depends on demand request size
Oracle slot update	138,000–158,000	Moderate cost; updates slot-level market inputs
Price computation	85,000	Deterministic pricing logic execution
Settlement execution	224,150	Highest cost; includes escrow and payment transfer

4.2. Comparison of DL performance with prior work

To evaluate forecasting performance within the context of existing research, Table 7 compares the proposed electricity demand model with a directly comparable study that utilized the SGRT-LMD dataset. This comparison is based on the R^2 value reported for the demand forecasting task.

An R^2 value of 0.9640 was reported for electrical load forecasting using the BiTCSAtt–TCN framework by [32]. The developed model CNN-MHA-BiLSTM obtained R^2 value of 0.9925 with difference of 0.0285. Note that both studies used the same datasets, so the comparison is a measure of the application of the forecasting method, but differences in preprocessing and partitioning and experimental design must be taken into account when interpreting the results.

4.3. BC execution performance

The gas consumption of selected key smart contract operations within the on-chain execution layer was measured, and the results are summarized in Table 8. Administrative operations consumed relatively less gas, while oracle updates, pricing, and settlement operations consumed more gas due to their computational and state-modification requirements. Oracle updates took about 138,000–158,000 gas units, price computation took around 85,000 gas units, and settlement execution was the most gas consuming at about 224,150 gas units. The results indicate that operations involving multiple state changes tend to incur higher on-chain execution overhead.

Table 9. Comparison of gas consumption for smart contract operations with previous studies.

Operation	Proposed framework	[33]	[34]
Participant registration	49,871	160,284	54,432–54,453
Order / Matching operation	210,061–224,073	354,520	84,065
Settlement / Energy trading	64,766–64,778	25,574	102,094

4.4. Comparison of BC performance with prior work

To establish a baseline for measuring the observed BC overhead, the maximum gas consumption of the proposed framework was compared against the benchmark [33] a summary of this comparison is presented in Table 9.

Table 9 presents a comparison of the gas consumption reported for selected smart contract operations in the proposed framework and two previous Ethereum-based P2P energy trading studies. reported gas consumption of 160,284 for participant registration, 354,520 for matching, and 25,574 for transfer operations. In [34], they reported approximately 54,432–54,453 gas for participant registration, 84,065 for prosumer–consumer matching, and 102,094 for energy trading.

In the proposed framework, participant registration required approximately 49,000 gas, while the highest reported consumption was 224,150 gas for settlement execution. These results provide a reference for comparing the on-chain resource requirements of different blockchain-based energy trading implementations.

Nonetheless, it is essential to note that this comparison should not be taken literally taking into account that the operation from one framework is different from the operation from another framework under this comparison. More specifically, pricing and order processes as well as payment processing and settlements take place in different ways in every implementation being compared. Therefore, the reported gas values are meant for the purpose of comparison and are not seen as the basis for assessing computational efficiency or scalability.

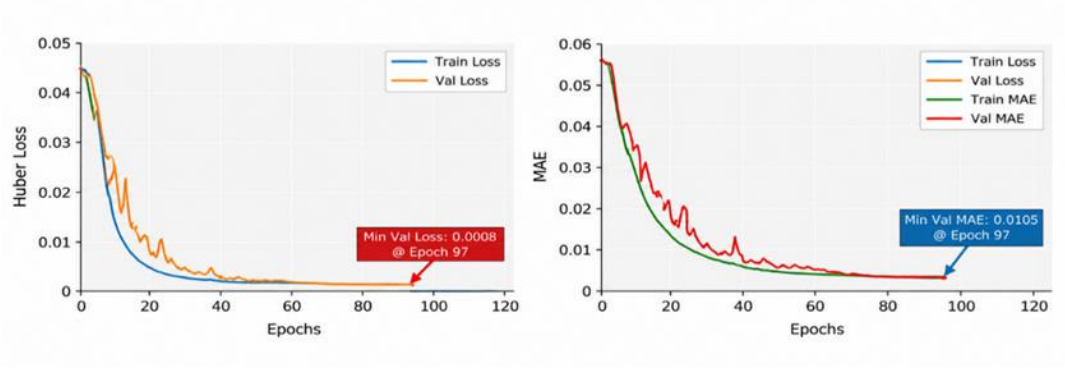

Fig. 11. Training dynamics of the proposed TCN–GRU model for P_{gen} prediction.

Table 10. Scenario-based market evaluation results.

Scenario	Market condition	Supply (kWh)	Demand (kWh)	β	P_{gen}	Unit price	Unmet demand (kWh)
1	Shortage	600	818	2.026	0.3329	0.3329	218
2	Supply Surplus	182	139	0.984	0.2874	0.2874	0
3	Severe Shortage	213	626	0.925	0.3103	0.3103	413
4	Moderate Shortage	87	237	0.574	0.3876	0.3368	150
5	Mild Shortage	112	125	3.958	0.3835	0.3835	13
6	Near-Balanced Shortage	233	264	1.710	0.3460	0.3460	31
7	Abundant Supply	946	541	1.970	0.3361	0.2914	0
8	Extreme Shortage	228	1332	3.472	0.2566	0.2566	1104

4.5. Dynamic pricing behavior and scenario-based analysis

The dynamic pricing mechanism was evaluated across eight simulation scenarios representing a range of supply demand conditions, from extreme shortage to significant surplus. The complete scenario results are shown in Table 10, followed by a detailed analysis of the pricing behavior and implementation outcomes observed in each case.

Results suggest that the pricing mechanism is sensitive to changes in demand, supply, and the market sensitivity coefficient. Under conditions of shortages, the amount of electricity demanded is greater than that available for sale. Consequently, aggregate market demand can only be partially fulfilled, although accepted individual orders are executed in full.

The volumes of unmet demand are 13 kWh for the case of the scenario S5 and 1,104 kWh for the case of S8 (scenario S8 includes the most critical shortage case). However, in both cases S2 and S7 transactions are performed with full execution as the supply is greater than demand. The generator price cap applies when the unconstrained electricity price is above the predicted cap. Hence, in scenarios S1, S2, S3, S5, S6 and S8, the final electricity price is equal to the predicted generator price cap $\hat{P}_{gen}$(t).

It means that price cap is applied in these particular cases. In the cases of S4 and S7, the price of the unit does not exceed the cap price and, thus, it indicates that price cap constraint is not applied here. At the same time, the β(t) varies among scenarios significantly from the point of 0.574 in S4 to that in S5 of 3.958, which illustrates the fact that the pricing mechanism can't be based on one locked-in value of sensitivity coefficient. Nonetheless, the results refer to scenarios created in simulations and under control rather than actual observations taken in the market. Thus, they indicate how the pricing mechanisms would behave in controlled conditions but do not serve as proofs of its functioning in the market conditions.

Scenario S8 offers another extreme stress test where the demand is 1,332 kWh and the supply is just 228 kWh. The unmet demand in this scenario equals 1,104 kWh. Even though the unit price remains constant at the producer price ceiling (0.2566), the fact that the unit price does not exceed the ceiling demonstrates that the ceiling is effective even under very extreme market conditions.

5. CONCLUSION

This article proposed an integrated AI and BC framework for dynamic energy pricing and decentralized trading within a SG environment. The proposed system bridges the gap between data-driven predictive intelligence and decentralized execution by combining DL models with BC-based smart contracts. Experimental results demonstrate high accuracy in predicting key market parameters, including electricity demand, market sensitivity β(t), and the generator price limit P_{gen}(t). The CNN–MHA–BiLSTM model achieved an R^2 of 0.9925 and an MAE of 0.2097 for demand prediction, and an R^2 of 0.9654 and an MAE of 0.2955 for β(t); meanwhile, the TCN–GRU model achieved an R^2 of 0.9808 and an MAE of 0.0123 for predicting the generator price limit. These findings yield predictive information for a more flexible pricing mechanism. The data indicates the transmission of these predictions into the BC execution layer via an oracle, allowing smart contracts to specify prices for electricity and execute P2P energy trades. The BC layer provides transparent transaction recording as well as contract execution automation. The behavior of the pricing mechanism is illustrated through a scenario analysis, which identifies several operating states of the system, including low supply, approximate balance, and high supply. The pricing mechanism responds to variations in supply and demand while enforcing the predicted generator price cap as an upper limit on the electricity unit price. The study showed a maximum gas consumption of about 224,150 units when using the BC. Nevertheless, the reliability of the proposed framework depends on the accuracy of the predicted market parameters. Further evaluation under extreme market conditions is recommended to assess the robustness of β(t) forecasting.

Further studies would be needed to address the issue of fair distribution of charges and large-scale implementation of the proposed approach. Planned future experimentations will be about making improvements to predictions via constrained learning and parameter estimation based on optimization. The introduction of the InterPlanetary File System (IPFS) for storing the appropriate information off-chain should allow for a reduction of chain storage space. Further developments will cover fairer market mechanisms, trading management system improvement,

and scaling in realistic conditions. Many possibilities for IoT, AI and distributed ledger technologies to allow for better and more flexible SG management can be fulfilled. In general, this research proves that the implementation of AI-based market prediction can get integrated with the energy pricing and trading based on BC in the course of experimental evaluation of the process.

Acknowledgement

The authors acknowledge the use of artificial intelligence tools (e.g., ChatGPT by OpenAI) for language editing and clarity improvement during the preparation of this manuscript. The authors are fully responsible for the scientific content, analysis, and conclusions.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Declaration of Generative AI in Scientific Writing

During the preparation of this work, the authors used ChatGPT to improve language quality, grammar, readability, and manuscript organization. The authors carefully reviewed, edited, and validated all generated content and take full responsibility for the content of this publication.

Data Availability Statement

The data used in this study are publicly available from the SGRT-LMD dataset source. The processed data and additional materials supporting the findings of this study are available from the corresponding author upon reasonable request.

CRediT Author Contributions

Ilham Albakri: Conceptualization, Methodology, Software, Data curation, Formal analysis, Investigation, Validation, Visualization, Project administration, Writing – original draft.

Haider Th. Salim ALRikabi: Supervision, Methodology, Validation, Resources, Writing – review and editing.

References

[1] J. Powell, A. McCafferty-Leroux, W. Hilal, and S. A. Gadsden, "Smart grids: A comprehensive survey of challenges, industry applications, and future trends," *Energy Rep.*, 2024.

[2] W. Hua, Y. Chen, M. Qadrdan, J. Jiang, H. Sun, and J. Wu, "Applications of blockchain and artificial intelligence technologies for enabling prosumers in smart grids: A review," *Renew. Sustain. Energy Rev.*, 2022.

[3] M. Takruri, M. Rabih, L. M. Dbeiss, H. Shall, S. A. Badawi, M. Al Atem, and M. Arnaout, "AI–IoT-enabled smart energy ecosystems: Architectures, security, and sustainability," *Eng.*, vol. 7, no. 7, p. 354, 2026.

[4] D. Kaur, S. N. Islam, M. A. Mahmud, M. E. Haque, and Z. Y. Dong, "Energy forecasting in smart grid systems: Recent advancements in probabilistic deep learning," *IET Gener. Transm. Distrib.*, 2022.

[5] M. Andoni, V. Robu, D. Flynn, S. Abram, D. Geach, D. Jenkins, P. McCallum, and A. Peacock, "Blockchain technology in the energy sector: A systematic review of challenges and opportunities," *Renew. Sustain. Energy Rev.*, 2019.

[6] Y. Guo, Z. Wan, and X. Cheng, "When Blockchain meets smart grids: A comprehensive survey," *High-Confidence Comput.*, vol. 2, no. 2, 2022.

[7] R. K. Agrawal, "BQEB forecastbench: Benchmarking AI models for smart grid forecasting using BQEB-Data v1," 2026.

[8] X. Wen, J. Liao, Q. Niu, N. Shen, and Y. Bao, "Deep learning-driven hybrid model for short-term load forecasting and smart grid information management," *Sci. Rep.*, vol. 14, no. 1, 2024.

[9] R. W. Kotla, S. R. Yarlagadda, and T. S. Devi, "Resilient adversarial–evolutionary multi-agent intelligence for real-time EV charging and energy trading in renewable-integrated smart grids," *Glob. Energy Interconnect.*, vol. 9, no. 3, pp. 621–638, 2026.

[10] S. Shirke, S. K. Gupta, and S. K. Jain, "Intelligent energy trading in smart microgrids using hopfield networks with Nash equilibrium fairness," *Iran. J. Sci. Technol. Trans. Electr. Eng.*, pp. 1–18, 2026.

[11] A. Umar, D. Kumar, T. Ghose, T. A. H. Alghamdi, and A. Y. Abdelaziz, "Decentralized community energy management: Enhancing demand response through smart contracts in a Blockchain network," *IEEE Access*, vol. 12, pp. 80781–80798, 2024.

[12] J. Dong, P. Ning, H. Zhao, and C. Song, "Decentralized peer-to-peer energy trading: A blockchain-enabled pricing paradigm," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 37, no. 1, 2025.

[13] K. Su, Y. Yu, and J. Zhang, "Blockchain-based smart grid power trading technology," *J. Eng. Appl. Sci.*, vol. 71, no. 1, 2024.

[14] M. Z. I. Badar and M. M. M. Al-Khalidy, "Integrating Blockchain and machine learning for secure energy transactions in smart grids," *Eng. Appl. Artif. Intell.*, vol. 165, no. 113419, p. 113419, 2026.

[15] F. Uddin, J. Hao, I. Alsaduni, S. Iqbal, M. Shafiullah, M. Aurangzeb, and S. Uddin, "IoT-enhanced virtual power plants with edge computing and Blockchain security for sustainable smart grid management," *Sci. Rep.*, vol. 16, no. 1, 2026.

[16] K. Singh, S. B. Goyal, A. S. Rajawat, and H. N. Waked, "A Blockchain-integrated AI framework for enhancing energy efficiency and sustainability in smart grids," in *Procedia Comput. Sci.*, pp. 2302–2311, Elsevier B.V., 2025.

[17] H. Sun and T. Weingärtner, "A Blockchain-based architecture for energy trading to enhance power grid stability," *Electron.*, vol. 14, no. 23, 2025.

[18] V. Kothuru, S. Dwivedi, K. Lingaraj, and A. S. L. G. Gupta, "Energy efficient hierarchical multi-agent graph transformer architecture for decentralized demand response and energy trading in Blockchain-coordinated smart grids," *Sustain. Comput. Inform. Syst.*, vol. 50, no. 101346, p. 101346, 2026.

[19] F. Ahsan, N. H. Dana, S. K. Sarker, L. Li, S. Muyeen, M. F. Ali, Z. Tasneem, M. M. Hasan, S. H. Abhi, and M. R. Islam, "Data-driven next-generation smart grid towards sustainable energy evolution: Techniques and technology review," *Prot. Control Mod. Power Syst.*, vol. 8, no. 1, pp. 1–42, 2023.

[20] G. Dudek and M. Blachnik, "Artificial intelligence and machine learning models for forecasting, optimization, and control in smart energy systems," *Energies*, vol. 19, no. 3, p. 768, 2026.

[21] M. F. Sulaima, A. K. Pradhan, M. Schmirler, and H. Namazi, "Physics-informed artificial intelligence architectures for carbon-neutral integrated energy systems," *Int. J. Energy Res.*, vol. 2026, no. 1, p. 5452111, 2026.

[22] T. Alladi, V. Chamola, J. J. P. C. Rodrigues, and S. A. Kozlov, "Blockchain in smart grids: A review on different use cases," *Sensors*, vol. 19, no. 22, p. 4862, 2019.

[23] "Smart grid real-time load monitoring dataset." https://www.kaggle.com/datasets/ziya07/smart-grid-real-time-load-monitoring-dataset/data.

[24] A. Rohini, P. Purnima, and M. Pradeep, "Deep learning-based fault prognosis in smart electrical networks using multivariate

time-series signal analysis," *Comput. Decis. Making: Int. J.*, vol. 3, no. 2S, pp. 354–364, 2026.

[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[26] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, "Short-term residential load forecasting based on LSTM recurrent neural network," *IEEE Trans. Smart Grid*, vol. 10, no. 1, pp. 841–851, 2019.

[27] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018.

[28] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," in *Int. Conf. Learning Representations*, 2023. Accessed: Oct. 20, 2026.

[29] S.-A. Chen, C.-L. Li, N. Yoder, S. O. Arik, and T. Pfister, "TSMixer: An all-MLP architecture for time series forecasting," 2023.

[30] I. D. Mienye and T. G. Swart, "A comprehensive review of deep learning: Architectures, recent advances, and applications," *Inf.*, vol. 15, no. 12, p. 755, 2024.

[31] K. Erdayandi, L. C. Cordeiro, and M. A. Mustafa, "Privacy-preserving and accountable billing in peer-to-peer energy trading markets with homomorphic encryption and Blockchain," *Sustain. Energy Grids Netw.*, vol. 41, p. 101568, 2025.

[32] Y. Sun and Y. Zhou, "BiTCSAtt-TCN: A hybrid model for efficient electricity load forecasting in smart grids," *Alexandria Eng. J.*, vol. 130, pp. 437–446, 2025.

[33] J. G. Song, E. S. Kang, H. W. Shin, and J. W. Jang, "A smart contract-based P2P energy trading system with dynamic pricing on Ethereum Blockchain," *Sensors*, vol. 21, no. 6, p. 1985, 2021.

[34] A. K. Vishwakarma, P. K. Patro, A. Acquaye, R. Jayaraman, and K. Salah, "Blockchain-based peer-to-peer renewable energy trading and traceability of transmission and distribution losses," *J. Oper. Res. Soc.*, vol. 77, no. 1, pp. 85–107, 2026.