Received 9 August 2023, accepted 19 August 2023, date of publication 23 August 2023, date of current version 30 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3308053

RESEARCH ARTICLE

A Robust Hybrid Classical and Quantum Model for Short-Term Wind Speed Forecasting

YING-YI HONG[1], **(Senior Member, IEEE), CHRISTINE JOY E. ARCE**[1], **AND TSUNG-WEI HUANG**[2]
[1]Department of Electrical Engineering, Chung Yuan Christian University, Taoyuan City 32023, Taiwan
[2]Quantum Information Center, Chung Yuan Christian University, Taoyuan City 32023, Taiwan

Corresponding author: Ying-Yi Hong (yyhong@ee.cycu.edu.tw)

This work was supported in part by the National Science and Technology Council, Taiwan, under Contract NSTC 112-2218-E-008-011 and Contract NSTC112-2119-M-033-001.

ABSTRACT Power scheduling by power utilities is more difficult than in the past decades because of a high penetration of renewable power generation, such as wind power generation, with highly uncertain and stochastic characteristics. To address this issue, a highly accurate technique for forecasting wind speed must be developed. In this work, a hybrid classical–quantum model is developed to exploit the advantages of two powerful models, a long short-term memory (LSTM) and a quantum neural network. Quantum neural networks, also known as parameterized quantum circuits, act like machine learning models but with greater expressive power. They comprise quantum gates that apply the principles of quantum mechanics in order to achieve quantum advantage. Additionally, to obtain a robust design that is insensitive to seasonal changes in the data, the Taguchi method is used to set up orthogonal experiments to set the hyperparameters of the proposed model. Historical data from seven sites in various countries (Taiwan, the Philippines, China, and South Korea) are used to forecast 24-hour-ahead wind speeds at the Fuhai wind farm near Taiwan. Comparative simulation results show that the proposed robust hybrid classical-quantum model outperforms current state-of-art models, such as classical nonlinear autoregressive network, random forest, extreme gradient boosting, support vector regression, and classical LSTM.

INDEX TERMS Deep learning model, quantum neural network, robust design, wind speed forecasting.

I. INTRODUCTION

A. IMPORTANCE OF WIND ENERGY FORECASTING

The 2030 Agenda for Sustainable Development, popularly called the '17 Sustainable Development Goals', officially came into force on January 1, 2016. Goal 7.2 concerns the importance of increasing renewable energy in the global energy mix. Over the years, the use of wind energy has grown to supply almost 5% of global electricity [1]. However, owing to its uncontrollable and intermittent nature, integrating wind turbines into energy systems poses stability and energy management problems. The volatility of wind power generation must be considered to ensure that the spinning reserve of power systems suffices for unit commitment (UC).

To take uncertain wind speeds into account, various forecasting methods are continuously being used and improved. Currently, the five classes of methods are as follows. (i) The persistence method assumes that the wind speed at time '$t+\Delta t$' equals that at time 't'. (ii) The physical approach [2] uses weather observations and a mathematical computer model of the atmosphere to generate forecasts. (iii) Statistical approaches [3], like autoregressive integrated moving average (ARIMA), estimate statistical relationships among input data. (iv) Artificial intelligence (AI) does not require predefined mathematical models. Examples include random forest (RF) [4], support vector machines [5], support vector regression (SVR) [6], extreme gradient boost (xGBoost) regression [7], nonlinear autoregressive neural networks (NAR) [8], and deep neural networks (DNN). The several types of DNN include autoencoders [9], the deep belief network (DBN) [10], the deep Boltzmann machine [11], the recurrent neural network (RNN) [12], long short-term memory (LSTM) [13], and the convolutional neural network (CNN) [14]. (v) Hybrid structures combine two or more

The associate editor coordinating the review of this manuscript and approving it for publication was Akshay Kumar Saha.

of the aforementioned and provide favorable outcomes by combining the advantages of two models. Some recent works with outstanding results have involved a combination of CNN and LightGBM [15], and a novel deep convolutional recurrent network to forecast wind power [16].

B. RISE OF QUANTUM MACHINE LEARNING

Recently, the field of quantum computing has been attracting increasing attention as research has demonstrated unparalleled quantum advantages over classical computing. These have led, in particular, to increased interest in Quantum Machine Learning (QML). Initially, the development of QML was mainly motivated by a desire to investigate quantum algorithms to accelerate classical training processes [17]. This paved the way to quantum equivalents of classical machine learning methods, such as (i) the Quantum Principal Component Analysis (QPCA) [18], in which a 4-qubit nuclear magnetic resonance (NMR) quantum processor was trained, resulting in the accurate recognition of all test images, (ii) the Quantum Support Vector Machine (QSVM) [19], in which IBM superconducting quantum computers were used to show that QSVM outperforms classical SVM for some datasets. Subsequently, with the continued rise of Noisy Intermediate-Scale Quantum (NISQ) processors, the field shifted towards Quantum Neural Networks (QNNs), which are the quantum versions of deep neural networks. In QNNs, parameterized quantum circuits (PQCs) or variational circuits act as neurons with parameters that are adapted to minimize the objective loss function. Killoran et al. [20] used the Strawberry Fields quantum simulator to perform binary classification using QNN. Results show a receiver operating characteristic (ROC) curve with an area of 0.945, opposed to an ideal value of 1. QNNs have also been applied to other classification tasks like the popular MNIST database, and to regression problems especially in the field of finance. Pistoia et al. [21], used Google's Cirq to simulate PQCs, and demonstrated that it outperformed classical BiLSTM neural networks whenever the noise coefficient was high, and was comparable otherwise.

C. HYBRID CLASSICAL-QUANTUM ALGORITHMS

Noisy quantum computers, with over 100 qubits, have recently been developed and shown to perform tasks better than current supercomputers [22]. While the present is an exciting time to investigate and explore quantum algorithms and other applications, noiseless quantum computers with thousands of qubits are required in order to fully exploit the advantages of major algorithms like Shor's algorithm and Grover's algorithm. Hence, researchers are leaning toward hybrid classical-quantum algorithms as applications of quantum computing to machine learning, with the general idea of combining quantum and classical computers. Endo et al. [23] used IBM's superconducting quantum computer to review the results for hybrid quantum-classical algorithms and quantum error mitigation techniques, and determined that future work on error mitigation would be extremely helpful since the use of NISQ devices is limited by large errors.

D. LIMITATIONS OF EXISTING METHODS

The aforementioned methods have one or more limitations with regard to wind speed forecasting and general system structure as follows.

(1) Physical methods use large mathematical models and require meteorological data on humidity, terrain structure, pressure, and other variables, to obtain accurate results. They are very costly and complicated [2] and are generally not used for short-term forecasting.

(2) Persistence and statistical methods yield accurate results but their accuracies quickly decline as the forecasting horizon increases [3].

(3) Many machine learning models are used for forecasting and may encounter difficulties when used with data with large standard deviations [4], [5], [6], [7], [8].

(4) Autoencoders and DBN require large amount of clean data to generate useful results [9], [10].

(5) Solving the model selection problem for (quantum) deep learning is time-consuming because the structure parameters and hyperparameters are obtained by trial-and-error [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21]; thus, the robustness of (quantum) deep learning models cannot be guaranteed.

E. MOTIVATIONS OF THIS WORK

Overcoming the above limitations would improve the accuracy of the predictions. This paper proposes a novel method, combining a deep learning LSTM model and QNN model, for 24h ahead wind speed forecasting using historical wind speeds at seven sites (in Taiwan, China, South Korea, and the Philippines). The motivation behind this combination is rooted in the potential to harness quantum computing capabilities to enhance and potentially revolutionize machine learning and neural network-based tasks. There are potential advantages associated with QNNs: (i) Quantum parallelism: QNN can process multiple computations simultaneously through superposition, allowing QNNs to potentially evaluate multiple input states in parallel, making QNNs attractive for certain machine learning tasks. (ii) Quantum feature mapping: quantum feature mapping allows for efficient encoding of classical data into quantum states, providing a more expressive representation of data. (iii) Quantum entanglement: Quantum entanglement enables strong correlations among qubits, which can be exploited in QNNs to capture complex relationships between features in the data. On the other hand, LSTM's ability to capture long-term dependencies, handle sequential data efficiently, and mitigate the vanishing gradient problem makes it a powerful choice for a wide range of sequential data processing tasks in machine learning. Based on these reasons, the proposed model integrates LSTM with QNN.

F. NOVELTIES AND CONTRIBUTIONS

The novelties of this paper are as follows:

(1) A hybrid classical and quantum model is proposed. The LSTM is augmented by a deep QNN, allowing for the seamless fusion of classical and quantum neural networks. This fusion facilitates the transfer of knowledge gained from the classical layers to the quantum layers, empowering the QNN to refine predictions through the principles of the quantum mechanism, resulting in highly accurate 24-hour ahead wind speed forecasts. Consequently, the limitations mentioned earlier (1) and (2) can be effectively overcome.

(2) The hybrid classical and quantum model leverages the complementary learning capabilities of classical and quantum layers. LSTM is known for its proficiency in handling time-series data, while the QNN extends the learning capacity through quantum entanglement. This unique combination enables the network to recognize more complex patterns, which could be challenging for classical models to achieve alone.

(3) Originally, QML was applied mainly to classification problems. This paper aims to extend its application to regression problems, specifically wind speed forecasting.

The contributions of this paper are as follows.

(1) Historical wind speed data from the other ancillary sites are used to forecast wind speeds at a target location in Taiwan. Spearman correlation analysis is used to determine the time lags to be inputted to the LSTM. This aims to overcome limitations (3) and (4) listed above.

(2) The structure hyperparameters of the LSTM and QNN are determined without trial-and-error but using a robust design, specifically, the Taguchi method. The LSTM hyperparameters, considered as design factors, are the number of neurons, number of hidden LSTM, and dropout rate. The QNN hyperparameters are the type of embedding and the circuit depth. This aims to improve limitation (5).

(3) The hybrid LSTM-QNN offers a practical solution for wind speed forecasting, as it harnesses quantum advantages without necessitating fully quantum algorithms. Its seamless integration with the existing classical infrastructure makes it more feasible to implement in real-world scenarios. This contribution opens up new possibilities for leveraging cutting-edge technologies to optimize renewable energy resources and decision-making processes.

This paper is organized as follows. Section II will discuss the background of this work with respect to quantum computing and the Taguchi method. Section III will describe the proposed hybrid LSTM and QNN, based on robust design. Section IV will present simulation results. Section V will draw conclusions.

II. BACKGROUND

A. QUANTUM COMPUTERS, SIMULATORS, AND ANNEALERS

The various approaches to quantum computation include general-purpose quantum computation, quantum simulation, quantum annealing, and digital annealing. Ultimately, a quantum computer performs calculations that are intractable on any classical supercomputer, effectively achieving a quantum advantage.

The first kind of quantum hardware is called the general-purpose quantum computer or universal gate model computer, which is constructed with processors that use quantum bits (qubits) instead of classical bits. Qubits exploit three main ideas from quantum mechanics, which are superposition, entanglement, and interference. Examples are publicly accessible IBM Quantum Systems [24] and the privately-accessible Google Quantum Computing Service, both accessed through the cloud.

Since NISQ computers are still being developed to become more stable and available for general and public use, quantum simulators are gaining popularity. Quantum simulators allow classical computers to execute quantum circuits as if they were being run on a quantum computer. Simulators vastly help in developing algorithms, debugging code, and more. Available simulators include run-it-yourself simulators, packaged with open-source tools such as IBM quantum simulators [24] and Google Cirq simulators [25], which are advanced cloud-based classical emulators of quantum systems,or hardware-optimized packages like NVIDIA cuQuantum, which allows any quantum enthusiast to evaluate quantum circuits.

Another approach involves the use of quantum annealers, such as the D-Wave annealer, mainly to solve optimization problems [26]. The physical concept is based on the phenomenon of annealing. Quantum annealers are based on the same principle: an optimization problem is specified as an entangled state, and qubits are then allowed to settle into the lowest-energy state or the globally optimal state. Although not as widely applicable as universal gate model computers, annealers are less affected by noise and therefore easier to build.

Finally, a set of variations on quantum annealers are known as digital annealers. These are regarded as enhanced annealers for solving combinatorial optimization problems. They are quantum-inspired computers that emulate qubits in a digital circuit. Unlike quantum computers, digital annealers operate at room temperature, saving much energy. Also, they have a fully connected architecture that allows the total interaction of signals across bits. Therefore, large-scale problems that are impossible to solve using classical computers, or require much computing time can be solved in near-real time. Recently, Fujitsu released pioneering digital annealers [27] to solve problems for businesses in real-world scenarios.

B. PENNYLANE

PennyLane is a Python library and software framework for optimization and machine learning of quantum and hybrid quantum-classical computations. PennyLane's core feature is computing the gradients of variational quantum circuits in a way that is compatible with classical techniques like backpropagation. It essentially extends the popular machine learning libraries like TensorFlow, PyTorch, and autograd, to handle modules of quantum computation.

PennyLane manages the execution of quantum computations, such as evaluation of circuits, and the transformation of quantum information back to the classical output of a quantum node. Thus, seamlessly forming quantum to classical connection. It can also handle more complex tasks such as training a hybrid quantum-classical machine learning model the same way a classical neural network is trained. Moreover, by using plugins, this framework can be interfaced with any gate-based quantum hardware or simulator to evaluate the hybrid models on publicly accessible quantum computers.

Another extremely useful feature of PennyLane is its library of templates of circuit architectures from recent quantum machine learning literature. Formally, these templates are called ansatzes. An ansatz consists of a series of quantum gates to be applied to qubits. With PennyLane, readily coded ansatz can be used to build/evaluate models depending on the problem at hand [28].

C. QUANTUM GATES

Mathematical operations on qubits are carried out by applying quantum gates. These quantum gates are mathematically represented as matrices, and qubits are mathematically represented as column vectors. Thus, qubit operations follow the rules of linear algebra and matrix multiplication. This section will describe some of the most commonly used quantum gates

1) ROTATION GATES

Rotation gates perform rotations about the three mutually perpendicular axes of the Bloch sphere [29]. These rotation operators are described by the equations that were provided by M. A. Nielsen and I.L. Chuang in [29]:

$$R_X(\alpha) = \begin{pmatrix} \cos\left(\frac{\alpha}{2}\right) & -isin\left(\frac{\alpha}{2}\right) \\ -isin\left(\frac{\alpha}{2}\right) & \cos\left(\frac{\alpha}{2}\right) \end{pmatrix} \tag{1}$$

$$R_Y(\alpha) = \begin{pmatrix} \cos\left(\frac{\alpha}{2}\right) & -sin\left(\frac{\alpha}{2}\right) \\ \sin\left(\frac{\alpha}{2}\right) & \cos\left(\frac{\alpha}{2}\right) \end{pmatrix} \tag{2}$$

$$R_Z(\alpha) = \begin{pmatrix} e^{-\frac{i\alpha}{2}} & 0 \\ 0 & e^{\frac{i\alpha}{2}} \end{pmatrix} \tag{3}$$

where α is the angle in the initial qubit state, and i is the imaginary number $\sqrt{-1}$.

To demonstrate the effect of R_Z gate when applied to an arbitrary single qubit state, $|\psi >$ with θ and ϕ as initial coordinates on the Bloch sphere, a mathematical solution is shown below.

$$|\psi >= \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) \\ e^{i\phi}\sin\left(\frac{\theta}{2}\right) \end{pmatrix}. \tag{4}$$

The rotation around the z-axis is as follows.

$$R_Z(\alpha)|\psi > = \begin{pmatrix} e^{-\frac{i\alpha}{2}} & 0 \\ 0 & e^{\frac{i\alpha}{2}} \end{pmatrix}\begin{pmatrix} \cos\left(\frac{\theta}{2}\right) \\ e^{i\phi}\sin\left(\frac{\theta}{2}\right) \end{pmatrix} \tag{5}$$

$$= \begin{pmatrix} e^{-\frac{i\alpha}{2}}\cos\left(\frac{\theta}{2}\right) \\ e^{\frac{i\alpha}{2}}e^{i\phi}\sin\left(\frac{\theta}{2}\right) \end{pmatrix} \tag{6}$$

$$= e^{-\frac{i\alpha}{2}}\cos\left(\frac{\theta}{2}\right)|0 > +e^{\frac{i\alpha}{2}}e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1 > \tag{7}$$

and to finalize the operation, a global phase factor can be freely multiplied to the quantum state shown in (7). In quantum computing, $|\psi >$ and $e^{i\gamma}|\psi >$ are mutually indistinguishable. So, multiplying the current qubit state by a global phase factor of $e^{i\alpha/2}$ yields the quantum state,

$$R_Z(\alpha)|\psi >= \cos\left(\frac{\theta}{2}\right)|0 > +e^{i(\phi+\alpha)}\sin\left(\frac{\theta}{2}\right)|1 > \tag{8}$$

Therefore, applying an R_z gate advances the angle ϕ by α and essentially rotates the state about the z-axis through angle α

2) R_{ZZ} GATE

The Rzz gate is obtained by taking the tensor product of two R_Z gates. It allows for higher-order encoding as it considers the interaction between two qubits. R_{ZZ} gate effectively extends the R_Z gate by applying it to both qubits, and producing a two-qubit $Z \otimes Z$ with the matrix representation,

$$R_{ZZ}(\alpha) = \begin{pmatrix} e^{-\frac{i\alpha}{2}} & 0 & 0 & 0 \\ 0 & e^{\frac{i\alpha}{2}} & 0 & 0 \\ 0 & 0 & e^{\frac{i\alpha}{2}} & 0 \\ 0 & 0 & 0 & e^{-\frac{i\alpha}{2}} \end{pmatrix} \tag{9}$$

3) CONTROLLED-NOT GATE

Another important two-qubit gate is the controlled-NOT (CNOT) gate. It works by changing the operation that is applied on one qubit depending on the value of the other qubit. This operation is analogous to classical "if-then-else" statements, and in quantum, these statements are called controlled gates. Specifically, CNOT gate transforms the quantum states by inverting the second qubit if and only if the first qubit is in state $|1 >$. Hence, the state of the second qubit, called the "target" qubit, is controlled by the first qubit, called the "control" qubit. Figure 1 displays basic CNOT operations.

The CNOT gate may also be considered analogous to the classical XOR gate. The control (c) and target (t) qubits are XORed, and stored in the target qubit. Its mathematical representations are given by the following equations.

$$\mathrm{CNOT}\,|c\ t > = |c, c \oplus t > \tag{10}$$

$$\mathrm{CNOT}_{ct} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{11}$$

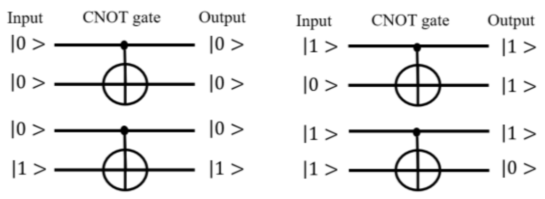


FIGURE 1. **Operations of CNOT gate.**

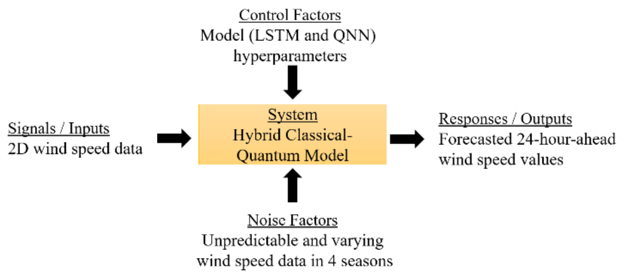

Control Factors
Model (LSTM and QNN) hyperparameters
Signals / Inputs
2D wind speed data
System
Hybrid Classical-Quantum Model
Responses / Outputs
Forecasted 24-hour-ahead wind speed values
Noise Factors
Unpredictable and varying wind speed data in 4 seasons

FIGURE 2. **P-diagram of system [30].**

4) HADAMARD GATE

The Hadamard gate maps the computational basis states into superposition states and vice-versa. It is very useful for any quantum circuit because it allows a quantum register to be loaded efficiently with an equally weighted superposition of all of the values that it contains. The Hadamard gate and its operations are defined [29] below.

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{12}$$

Its basic operation is as follows.

$$H\,|0> = 1/\sqrt{2} * (|0> + |1>) \tag{13}$$

$$H\,|1> = 1/\sqrt{2} * (|0> - |1>) \tag{14}$$

The Hadamard gate rotates the qubits from $|0>$ and $|1>$ on the z-axis to $|+>$ and $|->$ on the y-axis, respectively. If applied to a computational basis state, $|\psi>$, the Hadamard produces the following result.

$$H\,|\psi >= 1/\sqrt{2} * \left(|0> + (-1)^{\psi}\,|1>\right) \tag{15}$$

D. TAGUCHI METHOD OF ROBUST DESIGN

1) OVERVIEW OF THE PROCESS

Robust engineering is an optimization strategy that chooses the best nominal values of design factors while ensuring reliability amidst variability and noise. The Taguchi method is a way of optimizing design processes [30]. The relationship of the system with the factors of interest is captured by a parameter diagram or p-diagram, as in Fig. 2.

These factors are discussed below:

a. System: The hybrid quantum-classical model is the central mechanism of this process. It comprises the LSTM and QNN layers.

b. Signals or Inputs: The signals are input to the system to begin the process and generate an output. The inputs here are 2D wind speed data from seven wind farms in Taiwan, China, South Korea, and the Philippines.

c. Control Factors: Control factors, also called Design Factors, are design parameters that can be controlled. These factors affect the outputs of the system. For this wind speed forecasting problem, these refer to the model hyperparameters. For the classical part, LSTM cells, the number of LSTM layers, and the dropout rate are considered. On the other hand, for quantum part, the design factors are the kind of embedding and the depth of the QNN.

d. Noise Factors: Noise factors are uncontrollable or signals that cause variability in the outputs. Here, this variability corresponds to the unpredictability of wind speed data in the four seasons.

e. Responses or Outputs: These are the outputs of the system that are obtained after the relationship among the inputs, control factors, and noise factors is considered. The outputs herein are the forecasted day-ahead wind speeds.

2) ORTHOGONAL ARRAY

Various design factors are considered and the corresponding design space is systematically explored using an orthogonal array. This method enables the fewest possible experiments to be carried out, considering all possible combinations of levels of the design factors. Orthogonal arrays have the property that in every pair of columns (or design factors), each of the possible ordered pairs of elements (or levels) appears the same number of times. In combinatorics and experimental design, this is widely known as Latin square design. Common orthogonal arrays are L_{12}, L_{18}, or L_{36}, where the symbol L denotes "Latin square" – a mathematical theory developed by Leonhard Euler who used Latin characters in his works. The subscript means the number of experiments, but modifications or hybrids can also be produced.

In this paper, $L_{18}(2^1 3^7)$, with one two-level design factor and seven three-level design factors, is modified to fit the studied problem with only five design factors. Accordingly, design factors A, G, and H, which refer to the first, seventh, and eighth columns are disregarded, as shown in Table 1. The considered columns are enclosed by a red box in the table.

3) EVALUATION OF RESULTS

The Taguchi method helps to optimize design processes by using the signal-to-noise ratio (SNR) as a performance index for experiments and for analyzing the results to identify significant factors whose adjustment would lead to an absolute improvement. The formula for SNR varies with the problem at hand. For this wind forecasting task, the chosen coefficient of determination is R^2, so "the-bigger-the-better" SNR is used, as follows.

$$SNR = \eta_i = -10\log(s/n) \tag{16}$$

TABLE 1. $L_{18}(2^1 3^7)$ **orthogonal array.**

Exp. No.	A	B	C	D	E	F	G	H
1	1	1	1	1	1	1	1	1
2	1	1	2	2	2	2	2	2
3	1	1	3	3	3	3	3	3
4	1	2	1	1	2	2	3	3
5	1	2	2	2	3	3	1	1
6	1	2	3	3	1	1	2	2
7	1	3	1	2	1	3	2	3
8	1	3	2	3	2	1	3	1
9	1	3	3	1	3	2	1	2
10	2	1	1	3	3	2	2	1
11	2	1	2	1	1	3	3	2
12	2	1	3	2	2	1	1	3
13	2	2	1	2	3	1	3	2
14	2	2	2	3	1	2	1	3
15	2	2	3	1	2	3	2	1
16	2	3	1	3	2	3	1	2
17	2	3	2	1	3	1	2	3
18	2	3	3	2	1	2	3	1

where

$$s = \left(\sum_{j=1}^{n} \frac{1}{y_{ij}^2} \right) / \text{n} \tag{17}$$

n = 4 is the number of seasons, and y_{ij} is the observation made in the ith experiment in the jth season.

Thereafter, analysis of means (ANOM) is implemented to identify the level of design factor that most strongly affects the optimal design. To evaluate the ANOM, Table 1 is taken as an example, and the average effects of factor levels (AEFL) on η_i are calculated. The equations below show yields design factors A and B.

Since design factor A has only two levels, which appear nine times each, the relevant calculation is as follows.

$$\bar{A}_1 = \sum_{i=1}^{9} \eta_i / 9 \tag{18}$$

$$\bar{A}_2 = \sum_{i=10}^{18} \eta_i / 9 \tag{19}$$

Design factor B requires more calculations because it has three levels, which appear six times each. Therefore,

$$\bar{B}_1 = \left(\sum_{i=1}^{3} \eta_i + \sum_{i=10}^{12} \eta_i \right) / 6 \tag{20}$$

$$\bar{B}_2 = \left(\sum_{i=4}^{6} \eta_i + \sum_{i=13}^{15} \eta_i \right) / 6 \tag{21}$$

$$\bar{B}_3 = \left(\sum_{i=7}^{9} \eta_i + \sum_{i=16}^{18} \eta_i \right) / 6 \tag{22}$$

Given $\bar{B}_1$, $\bar{B}_2$, and $\bar{B}_3$, the level that yields the maximum AEFL is considered to be optimal. The same calculation and analysis are conducted for all other design factors.

III. PROPOSED MODEL

A. LONG SHORT-TERM MEMORY (LSTM)

LSTM is a popular improved model to the RNNs as it solves the vanishing gradient problem by adding a memory cell to the input, output, and forget gates. Thus, the LSTM can store and access information over long periods.

The forget gate controls how much information must be kept for the previous cell state, according to the following equation.

$$f_t = \sigma \left(W_f \cdot [h_{t-1}, x_t] + b_f \right) \tag{23}$$

where f_t is the forget gate unit for time step t; x_t and h_t are column vectors; $\mathrm{x_t}$ is the current input; h_t is the hidden state in which current outputs are stored; b_f is the bias vector, and W_f is the weight matrix.

The next step involves deciding which new information is to be stored in the cell state. This step has two parts. First, the input gate layer, i_t, decides which values are to be updated by using a sigmoid function. Second, the tanh layer creates a vector of new values, $\tilde{C}_t$.

$$i_t = \sigma \left(W_i \cdot [h_{t-1}, x_t] + b_i \right) \tag{24}$$

$$\tilde{C}_t = \tanh \left(W_C \cdot [h_{t-1}, x_t] + b_C \right) \tag{25}$$

The old cell state, C_{t-1},is updated into the new cell state according to,

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{26}$$

Finally, this cell state is filtered to produce an output. A sigmoid layer decides which parts of the cell state will be outputted. Then the tanh function shifts the values from -1 to 1, and this output is again multiplied to a sigmoid

$$o_t = \sigma \left(W_o [h_{t-1}, x_t] + b_o \right) \tag{27}$$

$$h_t = o_t * \tanh \left(C_t \right) \tag{28}$$

where o_t is the output gate, and h_t is the hidden state. The forget gate, input gate, and output gate all satisfy the equation for a sigmoid, which yields a value between zero and one, enabling the gate to be controlled.

B. QUANTUM NEURAL NETWORK (QNN)

A QNN comprises an embedding layer, a variational layer or parameterized quantum circuit (PQC), and a measurement layer. In this work, two PennyLane templates are used - an embedding template and layers template. Embedding is like feature mapping that encodes classical features into a quantum state, which will then be manipulated and processed by the layers template, which consists of series of trainable quantum gates that perform entanglements and rotations and so effectively act like the layers of a neural network. These ansatzes are developed for the purpose of developing a useful data embedding technique that is believed to be classically difficult to simulate as depth and width increase. Moreover, different variational forms tend to produce variously expressive circuits for quantum algorithms [31].

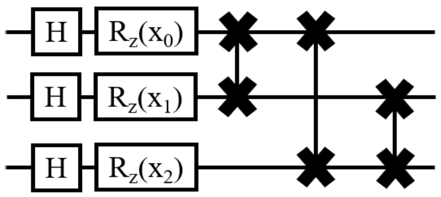


FIGURE 3. One block of an IQP embedding circuit.

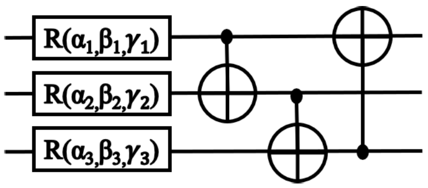


FIGURE 4. One block of the 'StronglyEntanglingLayers' template.

TABLE 2. Artificial neural network vs. quantum neural network.

Classical Artificial Neural Network	Quantum Neural Network
It comprises a collection of neurons connected with other neurons through links that have trainable weights.	It consists of quantum circuits composed of quantum gates with trainable parameters for performing operations or rotations in each qubit.
A bias term is added to the weighted sum of the inputs to find the output of the neuron.	Commonly used quantum gates are Hadamard, CNOT, and rotations, which leverage quantum superposition, interference, and entanglement.
The weighted sum is passed through an activation function.	A measurement gate is applied at the end of the quantum circuit in order to obtain a classical output.
It can be trained using classical computers.	It can be trained using quantum computers or quantum simulators that run on classical computers.

Three types of embedding are used to identify the one that will give the best results. First, amplitude embedding encodes classical features in the amplitude vector of qubits. It requires data to be normalized, as is typical in quantum mechanics. A qubit's amplitude corresponds to the square root of the probability that it will be found in a specific state. This embedding guarantees that each input will have a unique quantum encoding without the need to use several quantum gates. Second, angle embedding encodes features into the rotation angles of the qubits. The field of rotation can be user-defined as R_X, R_Y, or R_Z, but the default is R_X. This embedding requires a single qubit gate per qubit. Third, instantaneous quantum polynomial (IQP) embedding encodes the features into qubits using diagonal gates of an IQP circuit, as proposed by Havlíček et al. [32]. An IQP circuit comprises Hadamard gates, R_Z gates that are diagonal,as in (3), and lastly the two-qubit ZZ entanglers that encode higher-order data, providing interactions between two qubits that encode the product of two features. Figure 3 presents the circuit.

After transforming the classical data into quantum states, they are processed by the variational layer or PQC with trainable parameters. The 'StronglyEntanglingLayers' template used in this study is inspired by the circuit-centric classifier design [33] whose purpose is to obtain a circuit with a parameter count that only grows poly-logarithmically with respect to the number of input dimensions by leveraging the quantum principle of entanglement. The circuit consists of single qubit rotation gates whose classical parameters are learnable, and CNOT operators that provide entanglement, which allows qubits to interact freely and provide information to each other. For the 'StronglyEntanglingLayers', α, β, and γ are the parameters to be learned. Figure 4 presents the circuit.

These quantum layers may be repeated and the repeated layers may be cascaded together to form a more expressive network, which generally performs better [34]. However, these effects on performance still depend on the data and the problem at hand.

Finally, to obtain an output from the series of parameterized quantum circuits, a measurement gate is added. In this work, the expectation value of the qubits is measured in the computational basis at the end of the quantum circuit. This process is like averaging the expected results, weighted by their corresponding probabilities of appearing. This combination of embedding layer and entangling layer, followed by a measurement gate, forms a QNN.

A summary of the main differences of an ANN and QNN is listed in Table 2.

C. HYBRID MODEL

This study develops a combination of classical and quantum layers for forecasting wind speeds. The results that are obtained using the LSTM are then input to the QNN for further learning. The hybrid model is formed by inserting QNNs between classical layers. The classical LSTM is implemented using Keras and Tensorflow, and the quantum layer is implemented using the PennyLane package in Python.

Currently, the number of available qubits is limited to tens of qubits, which are few enough for our purposes. Importantly, before a quantum layer can be inserted, a classical layer that contains the same number of output cells as the number of qubits in the quantum layer must be stacked to connect them seamlessly. Figure 5 presents an example. A LSTM layer with 32 cells must be followed by a classical layer, which is another dense layer in this example, containing ten neurons, before it can be connected to the QNN with ten qubits. Then, the final predictions may be obtained by adding a fully connected layer after the QNN.

D. ROBUST DESIGN – TAGUCHI METHOD

To develop a hybrid model that is robust for the four seasons, the structured, engineering-based Taguchi method is used. Table 3 presents the parameters and hyperparameters that are chosen as design factors. The first three factors (A, B, C) are related to the classical part of the model, and the last two (D, E) are related to the quantum part.

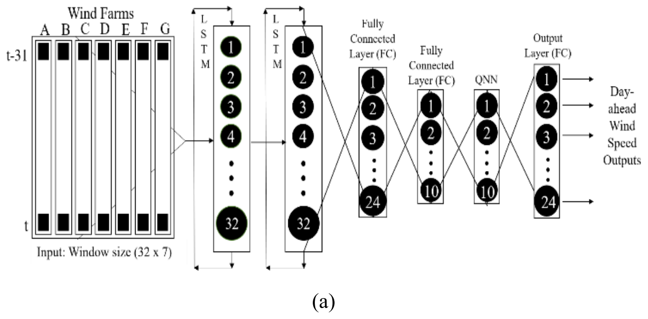
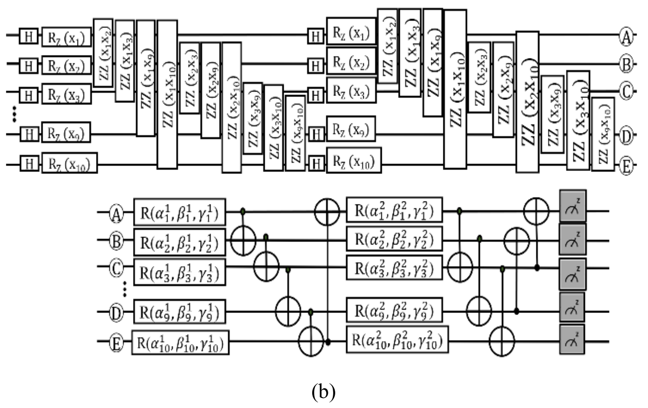



FIGURE 5. **(a) Example architecture with classical layers and a QNN with ten qubits and two circuit repetitions (depth = 2), (b) details of the QNN.**

TABLE 3. **Design factors and levels.**

Design Factors	Level 1	Level 2	Level 3
(A)Number of hidden LSTM layers	1	2	3
(B) Hidden LSTM cells	24	32	50
(C) Dropout rate	0.1	0.2	0.4
(D) Kind of Embedding	Angle	Amplitude	IQP
(E) Quantum Circuit Depth	1	2	3

As discussed in relation to Table 1, five design factors require a modification of the L_{18} orthogonal array. This array requires only 18 experiments to be performed, as opposed to hundreds.

The level of each design factor is set to the smallest value and then gradually increased. Table 3 shows the values of hyperparameters, which are commonly used at each level. Then, the SNR for each experiment and AEFL are calculated. The highest AEFL values obtained determine the optimal robust design. The specified levels of all design factors are applicable if the SNRs in the original 18 experiments are smaller than that in the robust design. Figure 6 presents a flowchart illustrating the process of obtaining the robust design using the Taguchi method. The procedure begins by identifying the design factors and their corresponding levels. Subsequently, performance metrics are recorded for each experiment conducted. ANOM is then carried out to determine the robust design. Finally, the experiment displaying the highest signal-to-noise ratio (SNR) is validated. If not achieved, the process is repeated by changing the levels of design factors to ensure the achievement of the most robust design.

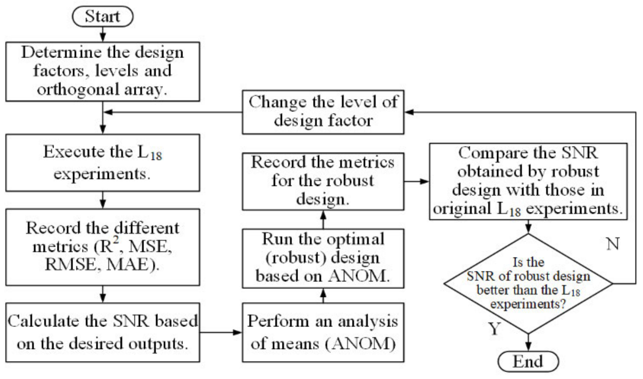

Start
Determine the design factors, levels and orthogonal array.
Change the level of design factor
Execute the L18 experiments.
Record the metrics for the robust design.
Compare the SNR obtained by robust design with those in original L18 experiments.
Record the different metrics (R², MSE, RMSE, MAE).
Run the optimal (robust) design based on ANOM.
N
Is the SNR of robust design better than the L18 experiments?
Y
Calculate the SNR based on the desired outputs.
Perform an analysis of means (ANOM)
End

FIGURE 6. **Flowchart of Taguchi's robust design.**

E. ABOUT THE DATA

Data plays a pivotal role in deep learning. Beyond choosing the most suitable model for the specific problem, ensuring well-prepared and efficiently processed data is essential to enhance model performance and achieve improved prediction accuracy. Each model has its own unique requirements. The subsequent paragraphs discuss the necessary steps for preparing a hybrid LSTM-QNN model.

The wind speed dataset that is used in this study is a record of historical data from seven wind farms in various locations in Taiwan, China, South Korea, and the Philippines from 1st May 2017 until 30th April 2018. This yields a total of 8,760 data points per location. The target wind farm whose data are to be predicted is located in Fuhai, Taiwan. The six other wind farms provide ancillary information, yielding a multivariate forecasting task. The data were purchased from Solargis company (see website solargis.com), which sells various meteorological data. Specifically, most weather projects face challenges due to inconsistent meteorological resource data. Low-quality data significantly impacts the accuracy of performance estimates during the development of prediction methods. Solargis analyst facilitates the use of automatic error detection tools and offers manual flagging options to identify potential issues in meteorological data. Consequently, the purchased data is thoroughly cleaned, ensuring the absence of bad data or missing values.

The wind speeds were estimated at a height of 10 meters above sea level; therefore, they need to be pre-processed and adjusted to the wind speed at hub height. This can be done by (29).

$$v = v_{ref} \frac{\ln(\frac{z}{z_0})}{\ln(\frac{z_{ref}}{z_0})} \tag{29}$$

where v is the wind speed at hub height; v_{ref} denotes the reference wind speed measured at the reference height z_{ref} (10 meters in height); z is the hub height (height above ground level for the desired velocity). The symbol z_0 signifies the

TABLE 4. **Characteristics of Fuhai wind farm.**

Season	Mean (m/s)	Standard Deviation (m/s)	Maximum (m/s)	Minimum (m/s)
Winter	10.52	3.70	21.9	0.2
Spring	7.18	4.43	20.6	0.2
Summer	4.49	2.56	16.4	0.1
Autumn	5.90	4.20	20.6	0.1

TABLE 5. **Interpretation table of spearman correlation coefficients [35].**

Spearman ρ	Correlation
≥ 0.70	Very strong relationship
$0.40 - 0.69$	Strong relationship
$0.30 - 0.39$	Moderate relationship
$0.20 - 0.29$	Weak relationship
$0.01 - 0.19$	No relationship or negligible relationship

roughness length (0.0002 meters) in the target off-shore wind farm (Fuhai).

Secondly, data must be normalized in order to minimize the effects of randomness and the wide range of values in the input data. This is mathematically shown in (30). Let v be the current wind speed, v_{min} be the minimum, v_{max} be the maximum, and v_{norm} be the normalized value. Then,

$$v_{norm} = \frac{v - v_{min}}{v_{max} - v_{min}} \tag{30}$$

The third step is to produce the input-output pairs. For this multi-step, multivariate problem on wind speed forecasting, the authors employed the intact prediction. It considers all values from one-hour ahead until 24-hour ahead to calculate the performance metrics such as R^2 coefficient of determination, MSE, MAE, and RMSE.

Each season has its own characteristics. Some have sudden peaks and extreme lows. For example, the average value of wind speed in summer is lower than those in other seasons because the wind characteristics are not as strong in summer. This fact may affect the model performance. Table 4 summarizes some of the mathematical characteristics of the data. It shows that the Fuhai wind farm has considerable potential for generating power.

Since the wind farms of interest herein are geographically dispersed, Spearman correlations were considered to determine the window size that is most suitable for solving the problem. Table 5 shows the correlation coefficients and their corresponding interpretation. The criteria for evaluation were to determine the number of lags that would yield a non-negligible relationship between the wind speeds at the maximum time lag for at least two locations, or a correlation of 0.2 and higher. Different results were obtained for each location, but the maximum number of hours was selected.

Upon consideration of the four seasons, a window size of 32 hourly data points is used to determine inputs to the LSTM. Figure 7 displays the results for the target site in Taiwan and the wind farm in the Philippines. It shows that at time lag 32, the correlations were higher than 0.2 or exhibit a non-negligible relationship.

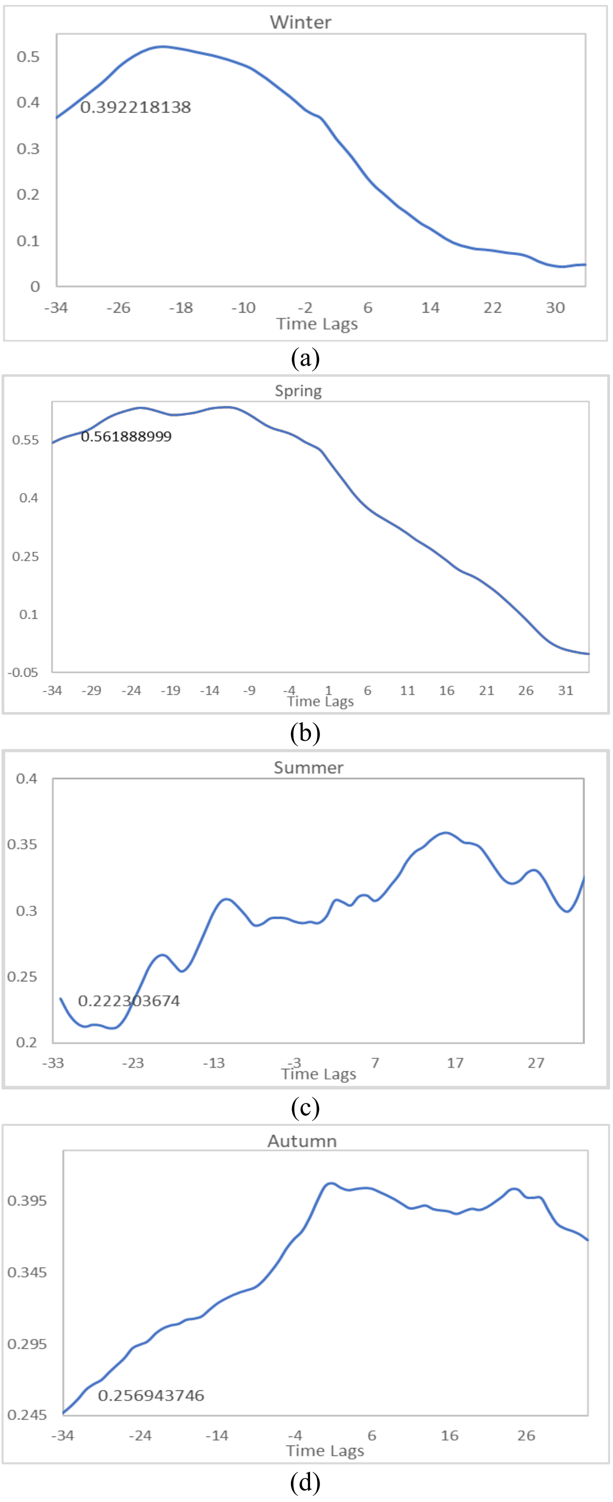


FIGURE 7. **Spearman correlation for wind speeds of wind farms in Taiwan and Philippines for (a) winter, (b) spring, (c) summer, (d) autumn.**

IV. RESULTS AND DISCUSSION

A. PROPOSED HYBRID MODEL

To evaluate the Taguchi experiments, the proposed method was implemented in Python using the Keras library from Tensorflow, and the default.qubit simulator in PennyLane.

TABLE 6. Performance metrics for autumn.

Exp.	R^2	RMSE	MAE
1	0.9506	0.0456	0.0309
2	0.8559	0.0799	0.0585
3	0.9605	0.0408	0.0267
4	0.8468	0.0769	0.0566
5	0.9576	0.0423	0.0274
6	0.9203	0.0579	0.0405
7	0.9633	0.0393	0.0256
8	0.8339	0.0836	0.0602
9	0.9699	0.0356	0.0224
10	0.9382	0.0510	0.0353
11	0.9659	0.0379	0.0244
12	0.8459	0.0806	0.0591
13	0.9556	0.0433	0.0286
14	0.9567	0.0427	0.0291
15	0.8244	0.0860	0.0629
16	0.8639	0.0757	0.0570
17	0.9581	0.0420	0.0273
18	0.9684	0.0365	0.0234

TABLE 7. R^2 scores and SNR values for the Taguchi experiments.

Exp	Winter	Spring	Summer	Autumn	SNR
1	0.9054	0.9206	0.8813	0.9506	39.2137
2	0.7966	0.7084	0.6578	0.8559	37.4204
3	0.9040	0.9183	0.8941	0.9605	39.2588
4	0.6322	0.7434	0.6557	0.8468	36.9724
5	0.9209	0.9259	0.8850	0.9576	39.2877
6	0.8888	0.8961	0.8841	0.9203	39.0559
7	0.9098	0.9247	0.8844	0.9633	39.2685
8	0.8316	0.7353	0.4805	0.8339	36.4470
9	0.9091	0.9409	0.9075	0.9699	39.3772
10	0.8850	0.8999	0.8558	0.9382	39.0196
11	0.9218	0.9288	0.8962	0.9659	39.3433
12	0.7650	0.7060	0.6360	0.8459	37.2212
13	0.9191	0.9134	0.8930	0.9556	39.2706
14	0.9077	0.9224	0.8992	0.9567	39.2827
15	0.5463	0.7171	0.6188	0.8244	36.2994
16	0.6462	0.7626	0.5998	0.8639	36.8646
17	0.9123	0.9291	0.9128	0.9581	39.3465
18	0.9103	0.9193	0.9172	0.9684	39.3507

The robust design process seeks an optimal design that is insensitive to different characteristics in the wind speeds in the four seasons. The Taguchi method provides a systematic way of conducting experiments without the need for full factorial experiments.

Three performance metrics are used to evaluate performance of the proposed model with the data characteristics that vary seasonally. These metrics are the coefficient of determination (R^2), root mean squared error (RMSE), and mean absolute error (MAE). Eighteen experiments, considering different design factors and corresponding levels, are performed, according to Table 1. Table 6 presents the results obtained using the autumn dataset. Based on this season alone, the ninth, 11th, and 18th experiments may be the most effective for producing accurate wind speed forecasts.

Based on the results in Table 6, the RMSE and MAE have wide ranges of values; therefore, they cannot be easily used as performance metrics to evaluate the experiments. Accordingly, R^2 is utilized as a performance index for the proposed robust design. Generally, a good performance is associated with an R^2 value of close to unity (1).

TABLE 8. Summary of analysis of means (Average effects of factor levels (AEFL) on η_i).

	A	B	C	D	E
Level 1	38.579	38.435	38.425	39.252	38.426
Level 2	38.361	38.521	38.637	36.871	38.570
Level 3	38.442	38.427	38.321	39.260	38.387
Delta	0.218	0.086	0.315	2.389	0.183
Rank	3	5	2	1	4

TABLE 9. Summary of results on the test set obtained using the robust design.

	R^2	MSE	RMSE	MAE
Winter	0.9247	0.0023	0.0476	0.0314
Spring	0.9347	0.0029	0.0537	0.0335
Summer	0.9123	0.0020	0.0452	0.0286
Autumn	0.9653	0.0015	0.0383	0.0249

Table 7 presents the R^2 scores of the 18 experiments for each of the four seasons. The last column presents the R^2-based SNR. See (16) and (17).

The best values vary with the season. In some seasons/experiments, an enormous decrease in R^2 score is seen. Therefore, robust optimization must be used to find the optimal levels of design factors, despite the variations in the characteristics of the dataset, based on the analysis of means, given by (16) to (22). Table 8 summarizes the AEFL and presents the optimal level and value of each design factor. It also helps to rank the design factors from the most important to the one with least effect on the outputs.

The results presented in Table 8 show that the most important design factor is the kind of embedding (D), followed by the dropout rate (C), the number of LSTM layers (A), the circuit depth (E), and the LSTM cells (B). These factors are ranked by delta value, which is the difference between the highest and lowest values of AEFL for a design factor.

The robust and optimal design with the highest AEFL values is the combination A1B2C2D3E2, as shown in Fig. 5. This combination states that for this wind speed forecasting problem, the IQP embedding layer (level 3 for design factor D; see Table 3) is the best encoding layer to be combined with the 'StronglyEntanglingLayers'. Moreover, a deep classical network with two hidden LSTMs learns the data better. Accordingly, a quantum circuit with a depth of two, or two repetitions of the quantum layers, is the most expressive quantum circuit in this study. This optimal design is not included in the original 18 experiments. Therefore, an additional 19th experiment with this design is performed, and the results on the test set are shown in Table 9. The R^2-based SNR (39.4036) for the optimal design is observed to be higher, and therefore better than that for any of the original 18 experiments in Table 7.

The model is trained with mean average error as the loss function using Adam optimizer. A workstation with an Intel Xeon W-2245 CPU@3.9 GHz×16, 64 GB of RAM and 3 NVIDIA GeForce RTX 2080Ti cards is used. Training the model with a batch size of 32 for 100 epochs takes 7 hours,

TABLE 10. Comparison between classical LSTM model and proposed hybrid model.

Season	Classical LSTM Model			Hybrid Model		
	R^2 Score	MAE	RMSE	R^2 Score	MAE	RMSE
Winter	0.9015	0.0367	0.0528	0.9247	0.0307	0.0472
Spring	0.9202	0.0399	0.0594	0.9347	0.0352	0.0561
Summer	0.8958	0.0355	0.0515	0.9123	0.0280	0.0451
Autumn	0.9490	0.0305	0.0442	0.9653	0.0261	0.0392

TABLE 11. Comparison among R^2 scores and SNR values of various models.

Model	Winter	Spring	Summer	Autumn	SNR
RF	0.6989	0.7006	0.6139	0.8643	36.9483
SVR	0.6908	0.5459	0.4570	0.7289	35.1850
XGBoost	0.6809	0.7169	0.6210	0.8458	36.9392
NAR	0.8496	0.8989	0.5646	0.8281	37.4363
LSTM	0.8310	0.8355	0.7646	0.9320	38.4297
LSTM Autoen-coder	0.9246	0.9381	0.8635	0.9499	39.2487
Proposed Model	0.9247	0.9347	0.9123	0.9653	39.4036

22 minutes, and 29 seconds. On the other hand, testing the model for one set of datapoints takes only 167 milliseconds.

The time complexity of LSTM for one gradient step is estimated as $O(Td_h^2 + Td_hd_i)$ where T, d_h and d_i denote sequence length, the number of hidden states and input dimension, respectively [36]. Additionally, in the work of Havlíček et al. [32], where the quantum "IQP Embedding" was derived, and its complexity is expressed as $O(\epsilon^{-2}$ - $|\Gamma^4|)$, where Γ represents the size of the training set and ϵ denotes the maximum deviation.

B. COMPARATIVE STUDY

In this section, two parts are presented. First, the quantum layer is removed and the performance of the classical model is compared with that of the hybrid model. Second, the proposed model is compared with five other models that are collected from the literature and that also perform day-ahead forecasting for wind speed or wind power. These include the random forest (RF) [4], the support vector regressor (SVR) [6], extreme gradient boosting (XGBoost) [7], the nonlinear autoregressive (NAR) model [8], long short-term memory (LSTM) [13] and LSTM-autoencoder.

1) PROPOSED NEURAL MODEL WITHOUT THE QUANTUM LAYER

The quantum layer is removed from the robust design obtained through the Taguchi method, and similar training is conducted on the classical LSTM model. Subsequently, the performance on the test set is recorded. Table 10 presents a comparison of the performance between the proposed hybrid model and the classical LSTM model. Notably, the performance of the classical model experiences a more pronounced decrease in the R^2 score during the summer. Several reasons might underlie this phenomenon, and one potential explanation could be the distinctive characteristics of summer data that pose challenges for the classical model's learning process.

2) COMPARISON WITH OTHER EXISTING MODELS

This section presents a summary of literature related to solving the same problem as this paper. The first model, Random Forest (RF), is popular due to its ability to deliver favorable results with minimal hyperparameter tuning. It makes predictions by combining outcomes from multiple decision trees. For this study, RF is implemented with the default number of trees (100), and the number of features considered for the best split is the square root of the total number of features [4]. The second model, SVR, is based on SVM (support vector machine), which maps samples to categories until they are clearly separated by a decision boundary that is ideally as wide as possible. It can be used for both classification and regression. For this regression problem, the kernel parameter settings are as follows: kernel type is ANOVA/polynomial, C is 1000, and the degree, G, and epsilon values are all set to unity [6]. The third model is XGBoost, which develops an ensemble of trees sequentially, unlike RF. In each iteration, a new tree is developed, depending on previously obtained information, which is foundational for minimizing the loss function. The hyperparameters are set as follows: the maximum depth is six; for early stopping, the "patience" argument is set to five; number of estimators is 10000, and eta is 0.2 [7]. The fourth method, NAR, involves an open loop network that can be transformed into a closed loop to predict as many future values as needed. The architecture has one hidden layer with five neurons and a feedback delay vector that contains 48 records. The hidden layer uses a hyperbolic tangent sigmoid transfer function while the output uses a linear transfer function. Its training procedure involves the Levenberg-Marquardt algorithm, which is one of the most popular algorithms and known to be fast and reliable [8]. The fifth model is the classical LSTM-RNN model with an input layer of 50 neurons and one hidden layer of another 50 neurons [13]. Additionally, an LSTM autoencoder is investigated as a deep neural architecture hypothesized to handle uncertainties in the data. Given the limited availability of references for day-ahead wind speed forecasting involving autoencoders, the architecture is modeled following the proposed hybrid LSTM-QNN approach. Specifically, it employs four stacked LSTM layers with hidden cell counts of 64, 32, 32, and 64, respectively. To ensure comparability with the proposed model, the number of trainable parameters is maintained at a similar level.

These model architectures are applied to the same wind speed datasets to obtain a comparison. Figure 8 presents the test R^2 scores.

Figure 8 illustrates that the proposed hybrid LSTM-QNN model consistently outperforms the other models by yielding consistently high R^2 scores, even in the face of volatile wind speeds across all four seasons. Both RF and XGBoost demonstrate similar performance, achieving scores above 0.8 for the

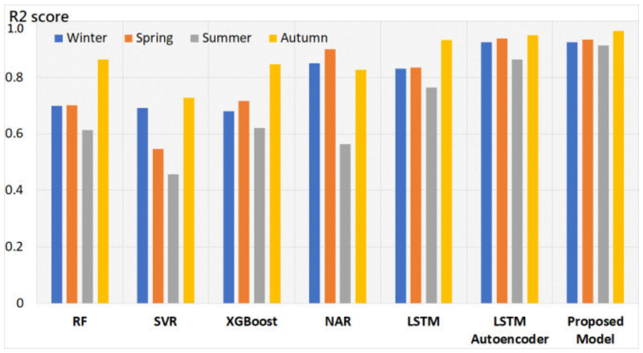


FIGURE 8. **Comparison of R^2 scores of various models on test set.**

autumn dataset but diverging in the other three seasons. Conversely, SVR consistently obtains an R^2 score below 0.8 for all four seasons. NAR performs well in winter, spring, and autumn but exhibits weaker results for the summer season. These variations may stem from the inherent variability in seasonal datasets. It is also worth noting that ML models generally perform better when provided with more data points and features; however, their ability to handle sudden variations is limited. Moreover, the classical LSTM performs well across all four seasons, and its results match those of the proposed model for the autumn dataset. Nevertheless, the proposed hybrid LSTM-QNN model still consistently outperforms the classical LSTM across all seasons. Furthermore, the LSTM autoencoder also performs admirably, comparable to the proposed model in spring, autumn, and winter. However, a notable decrease in the R^2 score is observed for the summer season. As theorized, the proposed hybrid LSTM-QNN model demonstrates robustness and superiority over its counterparts, offering enhanced accuracy despite the varying characteristics of the four seasons.

Table 11 presents the R^2 scores and SNR of seven different models used for wind speed forecasting, allowing for a comprehensive comparison of their performances. The R^2 score indicates how well the predicted values fit the actual data, with higher values representing a better fit. A higher SNR indicates a more robust model. By examining these statistical measures, insights can be gained into the performance and robustness of each forecasting model, helping to identify the most effective approach for predicting wind speeds. The proposed model consistently outperforms all other models, as evidenced by its higher SNR.

C. DISCUSSION

It's important to note that quantum computing is still in its early stages. Many of the proposed benefits are theoretical and will require advancements in quantum hardware, error correction, and quantum algorithms to become practical advantages in real-world applications. Nonetheless, the potential advantages of QNNs make them an exciting area of research and development in the intersection of quantum computing and machine learning.

In contrast to previous models, QNNs have the potential to offer quantum advantages over LSTM and can effectively capture complex nonlinear relationships, akin to SVR. While LSTM and SVR might be more practical in certain cases due to their established frameworks and widespread availability, this doesn't hold true for specific 24-hour-ahead wind speed forecasting. Despite sharing modeling capabilities with XGBoost, the proposed LSTM-QNN consistently demonstrates superior accuracy in forecasts. Similarly, RF stands out for its interpretability and ease of implementation, making it more popular than QNNs. Both NAR and QNN are capable of capturing complex nonlinear relationships in time series data, yet NAR's practicality outweighs the current limitations of QNNs. Amid the ease of implementing other models, the proposed hybrid classical quantum model showcases advantages in accuracy and robustness, rendering it a promising and powerful choice that stands to be further enhanced by ongoing advancements in quantum computing. These advancements may eventually pave the way for wider adoption of QNNs in the future.

V. CONCLUSION

This study proposes a robust hybrid classical-quantum model for 24-hour ahead wind speed forecasting. The goal is to exploit the advantages of two promising methods, classical deep LSTM and quantum machine learning. The innovations/findings are summarized as follows.

(1) LSTM is combined with a QNN, which leverages quantum mechanical principles, to provide an unparalleled quantum advantage. A QNN includes an embedding layer that maps classical data into a quantum state, which is then processed by a variational layer with trainable parameters, and then a measurement is made to obtain a classical output.

(2) The robust design of this proposed hybrid model is achieved by performing Taguchi orthogonal experiments, considering characteristics of various seasonal datasets. The hyperparameters (such as the number of hidden LSTM layers, number of hidden LSTM cells, dropout rate, kind of embedding, and quantum circuit depth) of the proposed hybrid model are determined systematically without trial-and-error.

(3) The templates that are used to build the proposed QNN can be made more expressive by repeating each one and cascading them, effectively creating a deep QNN. This results from the quantum principles - entanglement, which states that two or more entangled qubits share information, and interference, which denotes a single quantum state interfering or combining with other quantum states of itself. In this investigation, two repetitions (or depth $= 2$) is found to yield the best results as a result of constructive interference of the probability of obtaining the correct state, and the destructive interference of that of the incorrect state.

(4) Experiments reveal that the proposed hybrid model consistently produces more optimal outcomes than its counterpart classical model and other models, such as RF, XGBoost, SVR, NAR, LSTM, and LSTM autoencoder. Specifically, while the performance of classical models may

be comparable for some seasons, a notable decrease is observed during summer – the dataset with different statistical characteristics. The results obtained from this robust design, which integrates a classical LSTM with an expressive quantum circuit, demonstrate its great promise for near-term quantum devices.

Conducting the L_{18} experiments, evaluating the SNR, and performing an analysis of means, revealed that IQP embedding works best with 'StronglyEntanglingLayers' variational circuit. Moreover, the QNN has a depth of two circuit repetitions, making it more expressive through interference. This design is robust and the least sensitive to variations in the four seasonal data sets.

The promising results and consistently accurate predictions amidst different seasonal datasets demonstrated by this study lead to a more optimistic outlook on the potential expansion of quantum computing applications in areas such as solar irradiance forecasting and renewable energy hosting capacity optimization involving quantum computing. Specifically, QC and QNN may contribute in the areas of simulation, scheduling and dispatch, and even reliability analyses in the renewable energy industry.

This work did not take into account the effects of noise on the performance of the quantum circuit, as a perfect quantum computer was assumed in the numerical simulations. Future research may explore the impacts of qubit decoherence and loss, incorporating error correction and noise models on prediction accuracy. Additionally, various hybrid architectures and more sophisticated quantum ansatzes may also be employed.

REFERENCES

[1] P. Veers, K. Dykes, E. Lantz, S. Barth, C. L. Bottasso, O. Carlson, A. Clifton, J. Green, P. Green, H. Holttinen, D. Laird, V. Lehtomäki, J. K. Lundquist, J. Manwell, M. Marquis, C. Meneveau, P. Moriarty, X. Munduate, M. Muskulus, J. Naughton, L. Pao, J. Paquette, J. Peinke, A. Robertson, J. Sanz Rodrigo, A. M. Sempreviva, J. C. Smith, A. Tuohy, and R. Wiser, "Grand challenges in the science of wind energy," *Science*, vol. 366, no. 6464, Oct. 2019, Art. no. eaau2027, doi: 10.1126/science.aau2027.

[2] N. Chen, Z. Qian, I. T. Nabney, and X. Meng, "Wind power forecasts using Gaussian processes and numerical weather prediction," *IEEE Trans. Power Syst.*, vol. 29, no. 2, pp. 656–665, Mar. 2014, doi: 10.1109/TPWRS.2013.2282366.

[3] K. Yunus, T. Thiringer, and P. Chen, "ARIMA-based frequency-decomposed modeling of wind speed time series," *IEEE Trans. Power Syst.*, vol. 31, no. 4, pp. 2546–2556, Jul. 2016, doi: 10.1109/TPWRS.2015.2468586.

[4] Z.-H. Zhou. *Machine Learning*. Berlin, Germany: Springer, 2021, doi: 10.1007/978-981-15-1967-3.

[5] H. Demolli, A. S. Dokuz, A. Ecemis, and M. Gokcek, "Wind power forecasting based on daily wind speed data using machine learning algorithms," *Energy Convers. Manag.*, vol. 198, Oct. 2019, Art. no. 111823, doi: 10.1016/J.ENCONMAN.2019.111823.

[6] S. Nurunnahar, D. B. Talukdar, R. I. Rasel, and N. Sultana, "A short term wind speed forcasting using SVR and BP-ANN: A comparative analysis," in *Proc. 20th Int. Conf. Comput. Inf. Technol. (ICCIT)*, Dec. 2017, pp. 1–6, doi: 10.1109/ICCITECHN.2017.8281802.

[7] B. Bochenek, J. Jurasz, A. Jaczewski, G. Stachura, P. Sekuła, T. Strzyzewski, M. Wdowikowski, and M. Figurski, "Day-ahead wind power forecasting in Poland based on numerical weather prediction," *Energies*, vol. 14, no. 8, p. 2164, Apr. 2021.

[8] M. Morina, F. Grimaccia, S. Leva, and M. Mussetta, "Hybrid weather-based ANN for forecasting the production of a real wind power plant," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 4999–5005, doi: 10.1109/IJCNN.2016.7727858.

[9] J. Yan, H. Zhang, Y. Liu, S. Han, L. Li, and Z. Lu, "Forecasting the high penetration of wind power on multiple scales using multi-to-multi mapping," *IEEE Trans. Power Syst.*, vol. 33, no. 3, pp. 3276–3284, May 2018, doi: 10.1109/TPWRS.2017.2787667.

[10] M. Khodayar, J. Wang, and M. Manthouri, "Interval deep generative neural network for wind speed forecasting," *IEEE Trans. Smart Grid*, vol. 10, no. 4, pp. 3974–3989, Jul. 2019, doi: 10.1109/TSG.2018.2847223.

[11] C.-Y. Zhang, C. L. P. Chen, M. Gan, and L. Chen, "Predictive deep Boltzmann machine for multiperiod wind speed forecasting," *IEEE Trans. Sustain. Energy*, vol. 6, no. 4, pp. 1416–1425, Oct. 2015, doi: 10.1109/TSTE.2015.2434387.

[12] J.-F. Toubeau, J. Bottieau, F. Vallée, and Z. De Grève, "Deep learning-based multivariate probabilistic forecasting for short-term scheduling in power markets," *IEEE Trans. Power Syst.*, vol. 34, no. 2, pp. 1203–1215, Mar. 2019, doi: 10.1109/TPWRS.2018.2870041.

[13] U. Cali and V. Sharma, "Short-term wind power forecasting using long-short term memory based recurrent neural network model and variable selection," *Int. J. Smart Grid Clean Energy*, vol. 8, no. 2, pp. 103–110, 2019. [Online]. Available: http://www.ijsgce.com/index.php?m=content&amp;c=index&amp;a=show&amp;catid=78&amp;id=418

[14] Y.-Y. Hong and T. R. A. Satriani, "Day-ahead spatiotemporal wind speed forecasting using robust design-based deep learning neural network," *Energy*, vol. 209, Oct. 2020, Art. no. 118441.

[15] Y. Ju, G. Sun, Q. Chen, M. Zhang, H. Zhu, and M. U. Rehman, "A model combining convolutional neural network and LightGBM algorithm for ultra-short-term wind power forecasting," *IEEE Access*, vol. 7, pp. 28309–28318, 2019, doi: 10.1109/ACCESS.2019.2901920.

[16] X. Liu, L. Yang, and Z. Zhang, "Short-term multi-step ahead wind power predictions based on a novel deep convolutional recurrent network method," *IEEE Trans. Sustain. Energy*, vol. 12, no. 3, pp. 1820–1833, Jul. 2021, doi: 10.1109/TSTE.2021.3067436.

[17] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," *Nature*, vol. 549, pp. 195–202, Sep. 2017.

[18] T. Xin, L. Che, C. Xi, A. Singh, X. Nie, J. Li, Y. Dong, and D. Lu, "Experimental quantum principal component analysis via parametrized quantum circuits," *Phys. Rev. Lett.*, vol. 126, no. 11, Mar. 2021, Art. no. 110502.

[19] J.-E. Park, B. Quanz, S. Wood, H. Higgins, and R. Harishankar, "Practical application improvement to quantum SVM: Theory to practice," 2020, *arXiv:2012.07725*.

[20] N. Killoran, T. R. Bromley, J. M. Arrazola, M. Schuld, N. Quesada, and S. Lloyd, "Continuous-variable quantum neural networks," *Phys. Rev. Res.*, vol. 1, no. 3, Oct. 2019, Art. no. 033063, doi: 10.1103/PhysRevResearch.1.033063.

[21] M. Pistoia, S. F. Ahmad, A. Ajagekar, A. Buts, S. Chakrabarti, D. Herman, S. Hu, A. Jena, P. Minssen, P. Niroula, A. Rattew, Y. Sun, and R. Yalovetzky, "Quantum machine learning for finance," 2021, *arXiv:2109.04298*.

[22] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, Aug. 2018.

[23] S. Endo, Z. Cai, S. C. Benjamin, and X. Yuan, "Hybrid quantum-classical algorithms and quantum error mitigation," *J. Phys. Soc. Jpn.*, vol. 90, no. 3, Mar. 2021, Art. no. 032001.

[24] *IBM Quantum*. Accessed: Feb. 2, 2023. [Online]. Available: https://quantum-computing.ibm.com/

[25] *Simulation: CIRQ: Google Quantum AI*. Google Quantum AI. Accessed: Feb. 2, 2023. [Online]. Available: https://quantumai.google/cirq/simulate/simulation

[26] *Quantum Computing: D-Wave*. Accessed: Feb. 2, 2023. [Online]. Available: https://www.dwavesys.com/learn/quantum-computing/

[27] *Fujitsu Digital Annealer—Solving Large-Scale Combinatorial Optimization Problems Instantly*. What is Digital Annealer: Fujitsu Globa. Accessed: Feb. 2, 2023. [Online]. Available: https://www.fujitsu.com/global/services/business-services/digital-annealer/what-is-digital-annealer/

[28] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, M. S. Alam, S. Ahmed, J. M. Arrazola, C. Blank, A. Delgado, S. Jahangiri, K. McKiernan, J. J. Meyer, Z. Niu, A. Szava, and N. Killoran, "PennyLane: Automatic differentiation of hybrid quantum-classical computations," 2018, *arXiv:1811.04968*.

[29] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge, U.K.: Cambridge Univ. Press, 2003, pp. 15–19.

[30] G. Taguchi, S. Chowdhury, and S. Taguchi, *Robust Engineering*. London, U.K.: McGraw-Hill, 2000.

[31] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner, "The power of quantum neural networks," *Nature Comput. Sci.*, vol. 1, no. 6, pp. 403–409, 2021.

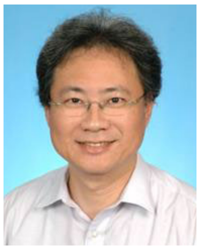
[32] V. Havlícek, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, "Supervised learning with quantum-enhanced feature spaces," *Nature*, vol. 567, no. 7747, pp. 209–212, Mar. 2019.

[33] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, "Circuit-centric quantum classifiers," *Phys. Rev. A, Gen. Phys.*, vol. 101, no. 3, Mar. 2020, Art. no. 032308, doi: 10.1103/PhysRevA.101.032308.

[34] T. Nguyen, I. Paik, Y. Watanobe, and T. C. Thang, "An evaluation of hardware-efficient quantum neural networks for image data classification," *Electronics*, vol. 11, no. 3, p. 437, Feb. 2022, doi: 10.3390/electronics11030437.

[35] Z. Yan, S. Wang, D. Ma, B. Liu, H. Lin, and S. Li, "Meteorological factors affecting pan evaporation in the Haihe River Basin, China," *Water*, vol. 11, no. 2, p. 317, Feb. 2019, doi: 10.3390/w11020317.

[36] M. Rotman and L. Wolf, "Shuffling recurrent neural networks," in *Proc. AAAI Conf. Artif. Intell.*, May 2021, vol. 35, no. 11, pp. 9428–9435.



YING-YI HONG (Senior Member, IEEE) received the B.S.E.E. degree from Chung Yuan Christian University (CYCU), Taiwan, in 1984, the M.S.E.E. degree from National Cheng Kung University (NCKU), Taiwan, in 1986, and the Ph.D. degree from the Department of Electrical Engineering, National Tsing Hua University (NTHU), Taiwan, in December 1990. Sponsored by the Ministry of Education, Taiwan, he conducted research with the Department of Electrical Engineering, University of Washington, Seattle, from August 1989 to August 1990. He has been with CYCU, since 1991, where he was the Dean of the College of Electrical Engineering and Computer Science, from 2006 to 2012. He was promoted to be a distinguished professor due to his exceptional research, leadership, teamwork, and international collaboration, in 2012. From 2012 to 2018, he was a General Secretary with CYCU, where he is currently the Vice President. His research interests include power system analysis and artificial intelligence applications. He received the Outstanding Professor of the Electrical Engineering Award from the Chinese Institute of Electrical Engineering (CIEE), Taiwan, in 2006. He was the Chair of the IEEE PES Taipei Chapter, in 2001.

CHRISTINE JOY E. ARCE received the B.S. degree in electronics engineering from the Technological Institute of the Philippines (T. I. P.), Manila, in October 2019. She is currently pursuing the master's degree in electrical engineering with Chung Yuan Christian University, Taiwan. Prior to this, she was an Instructor with the Electronics Engineering Department, T. I. P., from 2019 to 2021. Her research interests include machine learning and quantum computing.

TSUNG-WEI HUANG received the M.S. and Ph.D. degrees from the Department of Physics, National Taiwan University (NTU), in 2009 and 2017, respectively. From 2017 to 2020, he was a Research Fellow working on quantum phenomena with the Nitrogen-Vacancy (NV) Center and implementing quantum algorithms on the quantum computer with the NTU-IBM Quantum Computer Center. He is currently an Assistant Professor with the Department of Information and Computer Engineering and a member of the Quantum Information Center, Chung Yuan Christian University, Taiwan. His research interests include optimal gate operation in open quantum systems, quantum transport in graphene systems with defects, and quantum algorithms.