www.nature.com/scientificreports

scientific reports

OPEN

Electrical load forecasting in power systems based on quantum computing using time series-based quantum artificial intelligence

Mohammad Reza Habibi[1]✉, Saeed Golestan[1], Yanpeng Wu[2], Josep M. Guerrero[1,3] & Juan C. Vasquez[1]

A proper, reliable, and economic operation of a power system relies on a precise energy management strategy. For a reliable energy management strategy, information about the power system including power production and power consumption is required. However, consumer behaviour can be unpredictable, which can result to a high level of uncertainties for the load profile. So, this type of issue (existence of the uncertainty in power system) makes the energy management a complex task. The knowledge about the future state of the power system (e.g., the values of loads) can reduce the difficulty of this task, and it can lead to a more efficient energy management. This paper implements quantum computing-based artificial neural network to predict the future values of loads. For this purpose, this paper uses hybrid quantum/classical artificial neural network for a short-term forecasting of loads. The implemented quantum computing-based strategy is deployed using time series-based technique without using extra information (e.g., the weather condition, and behaviour of the consumer), and it only uses the current and historical values of the load to predict the future value of that. To examine the effectiveness of the hybrid quantum/classical artificial neural network, two different types of loads are selected from an experimental lab and the quantum-based approach is tested on those loads. The obtained results can proof the potential of quantum artificial intelligence to be used for forecasting-based challenges in smart grids.

Keywords Quantum computing, Load forecasting, Artificial intelligence, Artificial neural networks, Smart grid, Residential load

Artificial intelligence (AI) and especially artificial neural networks (ANNs) have been used to solve various issues and in different applications in smart grids, e.g., control-based applications[1,2], protection-based applications[3–5], detection and mitigation of cyber-attacks[6–13], sensorless voltage estimation-based strategy for calculation of the total harmonic distortion[14], wind and solar power prediction[15], cybersecurity monitoring of cyber-phyisical power electronic converters[16], etc. One of issues in smart grids which can be solved by ANNs is the prediction of the loads. There can be uncertainty related to electrical loads, and load forecasting can be considered as one of important challenges and duties for industry, where it plays an important role in power systems and it can affect the operation of power systems[17–19]. To have a reliable operation of a power system, short-term prediction of loads is a necessary task[20]. Previously, some studies have been done to support the application of load forecasting in power systems. For example, phase space reconstruction and stacking ensemble learning have been implemented for anticipation of load in[21]. Also, a strategy using transfer learning and deep residual neural networks for residential load prediction in[22] can be mentioned as another instance. An approach for the short-term prediction of electrical loads using temporal feature selection and based on long short-term memory (LSTM) has been introduced by[23]. As another example, an interpretable memristive LSTM-based strategy has been implemented by[24] for probabilistic-based prediction of residential loads. In addition, a federated learning-based methodology has been deployed in[25] for short-term anticipation of residential loads and based on LSTM. Furthermore, an approach using improved temporal convolutional network and densely connected convolutional network has been used for short-term prediction of loads in[26].

[1]AAU Energy, Aalborg University, Aalborg, Denmark. [2]The R&D department, Brodersen Systems, Rødovre, Denmark. [3] University of Valladolid, Valladolid, Spain. ✉email: mre@energy.aau.dk

Previous works have shown effective results. However, they are mainly structured based classical approaches, e.g., classical AI. But, the advantages of quantum computing can lead to replace the classical computing with quantum-based strategy to gain more benefits in different ways using quantum mechanics, e.g., speedup the calculations using the concepts of the superposition and entanglement, and communication using the concept of the entanglement[27–30]. Therefore, it is necessary to open a window for the deployment of quantum computing to solve the challenges of smart grids, e.g., load forecasting. Before, some attempts have been initialized to use the concept of the quantum computing in power and energy applications. For example,[31–34] have done studies related to quantum computing, communication, and cybersecurity of microgrids. Also, a quantum computing-based methodology have been developed by[35] in order to unit commitment. As another example,[36] has deployed quantum computing to propose quantum power flow. Also,[37,38] have tried to use quantum computing for the electromagnetic transients program. Besides, quantum computing has been used to provide a methodology for the stability assessment of power systems in[39]. For more examples,[40] and[41] have suggested hybrid strategies for load forecasting based on support vector regression using quantum tabu search and chaotic quantum genetic algorithm, respectively. For more investigation regarding power applications and quantum computing,[28,42–45] can be studied.

Although the above-mentioned studies have implemented quantum computing for solving different issues in power systems (e.g., cybersecurity, unit commitment, and power flow), but still there is a gap to address the forecasting-based challenges (e.g., load prediction) in power systems. In addition, the few mentioned works related to the prediction of loads have implemented the concept of quantum computing for the optimization part related to finding the optimized values of the parameters of the model, that is used for the forecasting. In other words, the mentioned works did not implement quantum computing for modeling the quantum layer as a part of the model to be used for the prediction. In addition, the mentioned previously studied did not deploy quantum computing for ANN-based applications, which are a powerful tool to solve prediction-based challenges.

Therefore, this paper tries to fill this gap by the deployment of quantum artificial intelligence for a short-term prediction of loads. This paper uses a hybrid quantum/classical strategy, i.e., hybrid quantum/classical ANN (that can be titled Q/C-ANN). Still, the number of the available quantum computers is very limited. In addition, the number of the accessible qubits is very limited and it is not possible to implement a fully quantum computing-based strategy for a large scale system. Furthermore, by increasing the number of the quantum gates and qubits, the sensitivity of the quantum-based circuit to the environmental noises (e.g., thermal and magnetic noises) can be increased. Therefore, at the moment, the deployment of a fully quantum circuit can not guarantee the reliability of the system. So, due to the current limitation about the fully quantum computing-based applications, this paper implements a hybrid quantum/classical approach to initialize a quantum computing-based strategy to address a prediction-based challenge (i.e., load forecasting) in a power system.

In the rest of this paper, Section II will talk about the basics of quantum computing. Also, in Section III, Q/C-ANNs will be discussed. In addition, Section IV will introduce the proposed strategy to use Q/C-ANNs for load forecasting. In section V, more math-based discussions (i.e., updating the states of the qubits, quantum measurement and calculation the output of the quantum layer, the encoding technique to map classical data into quantum states, and data pre-processing) will be talked. Besides, the proposed strategy will be examined on two different residential loads by Section VI, and this section will show the results. In Section VII, other classical AI-based methods will be examined on the implemented dataset. Further, Section VIII will discuss this study. Furthermore, Section IX and Section X will conclude the paper, and talk about the suggested future works.

Introduction to quantum computing

Currently, there are various types of computation-based strategies, e.g., swarm intelligence, AI, and quantum computing. However, the physical concept behind them can be considered different. For example, classical intelligence-based approaches implement classical bits as the basics of the computation. But, among the mentioned computing strategies, quantum computing can be considered more distinguish, due to the implementation of qubits as the basis of the computation and also for carrying out the information. For more clarification, a bit can be 0 or 1. But, the state of a qubit can be as follows[46]:

$$|\Gamma\rangle = a_1|0\rangle + a_2|1\rangle, \tag{1}$$

where, a_1 and a_2 are complex numbers, which should satisfy the following equality constraint[46]:

$$|a_1|^2 + |a_2|^2 = 1. \tag{2}$$

When the state of a qubit is measured, it can be either in state $|0\rangle$ or $|1\rangle$, with the probability of $|a_1|^2$ and $|a_2|^2$, respectively[46]. Also, (1) can be written as follows[46]:

$$|\Gamma\rangle = cos\left(\frac{\sigma}{2}\right)|0\rangle + e^{i\lambda} sin\left(\frac{\sigma}{2}\right)|1\rangle. \tag{3}$$

Where, σ and λ are real numbers ($\sigma, \lambda \in \mathbb{R}$). For a visual representation of the state of a qubit, the Bloch sphere can be implemented[46]. In Fig. 1, the state of a qubit using the Bloch sphere is shown.

Typically, based on Fig. 2, a quantum circuit can contain three main parts. The first part includes initialized qubit/qubits, which can be implemented as the basis of the calculations. In addition, the second part can have quantum operators including gates to structure the main part related to the algorithm. Furthermore, the last part is related to measurement/measurements, for measuring the final state of the qubit/qubits.

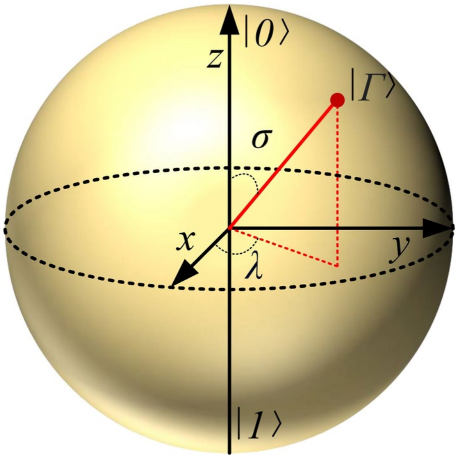


Fig. 1. The representation of the Bloch sphere for (3), where, for more details,[46] can be studied.

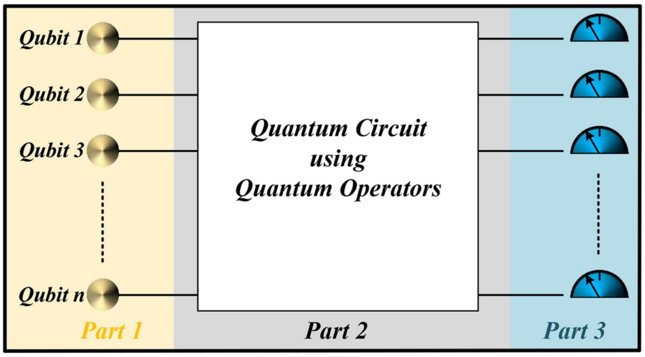


Fig. 2. The general architecture of a quantum circuit to be used for quantum computing.

The state of a qubit can be represented using a vector. Therefore, $|\Gamma\rangle$ can be represented as follows[46]:

$$|\Gamma\rangle = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}. \tag{4}$$

Also, there is a matrix representation for a quantum operator and using the linear algebra, the final state of a qubit system or a multi qubit system can be obtained. For more clarification, the final state of a single qubit circuit (based on Fig. 3(a)) can be obtained as follows:

$$|\Gamma_{out}\rangle = U|\Gamma in\rangle. \tag{5}$$

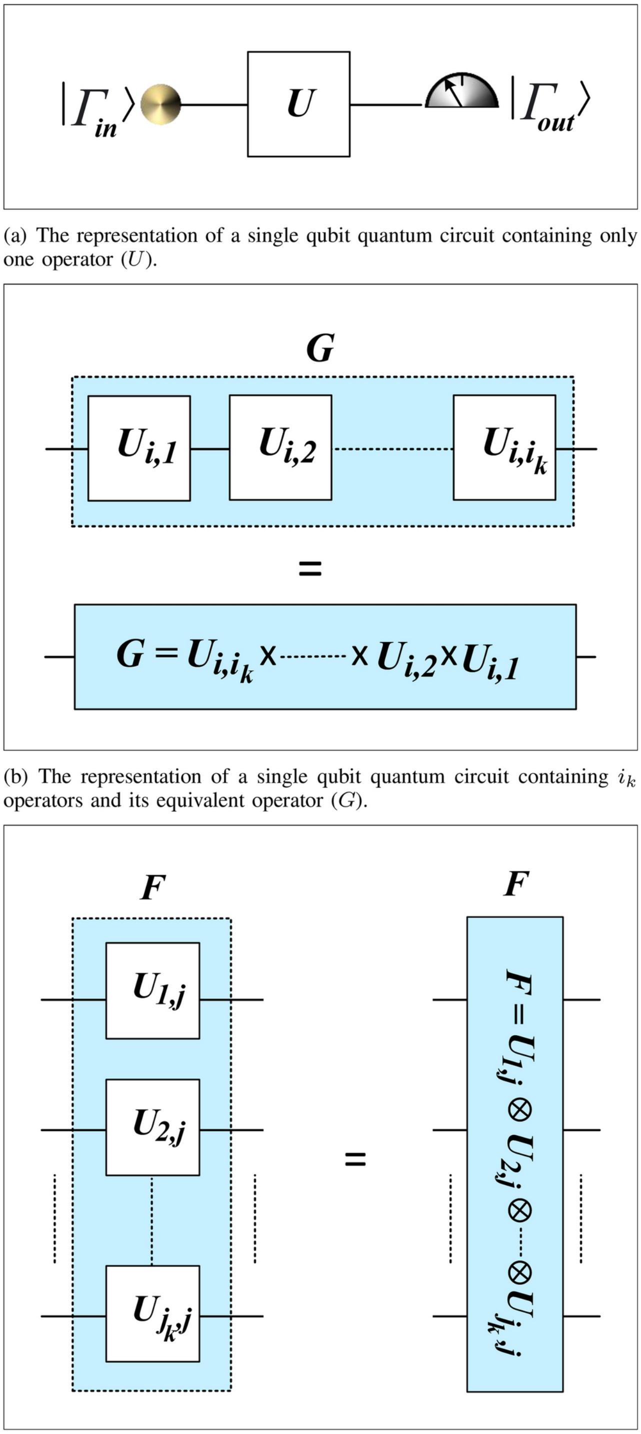
In addition, for a single qubit system, in the case of a series operators ($U_{i,1}$, $U_{i,2}$,...,U_{i,i_k}), the equivalent operator G (based on Fig. 3(b)) of the system can be calculated as follows:

$$G = U_{i,i_k} \times U_{i,i_{k-1}} \times ... \times U_{i,2} \times U_{i,1}. \tag{6}$$

Further, for the case of a multi qubit circuit with j_k qubits and one operator for each qubit, the equivalent operator (F) of the system can be calculated (based on Fig. 3(c)) as follows:

$$F = U_{1,j} \otimes U_{2,j} \otimes ... \otimes U_{j_k-1,j} \otimes U_{j_k,j}. \tag{7}$$

For the calculation of the equivalent operator of a multi qubit system, firstly, the equivalent operator of each qubit can be calculated based on (6). So, the system can be converted to a multi qubit circuit with one equivalent operator for each qubit. Then, the equivalent operator of all the system can be obtained using (7). It is imporant to note that, for the case of quantum circuits including entangled qubits, the calculations can be more complex. In the next part, it will be shown that how a quantum circuit can be used in an AI-based strategy. Therefore, the structure of a Q/C-ANN including quantum operators and also classical parts will be discussed.


$|\Gamma_{in}\rangle$
U
$|\Gamma_{out}\rangle$
G
$U_{i,1}$
$U_{i,2}$
U_{i,i_k}
=
$G = U_{i,i_k} \times \cdots \times U_{i,2} \times U_{i,1}$
F
$U_{1,j}$
$U_{2,j}$
$U_{j_k,j}$
=
F
$F = U_{1,j} \otimes U_{2,j} \otimes \cdots \otimes U_{j_k,j}$

(a) The representation of a single qubit quantum circuit containing only one operator (U).

(b) The representation of a single qubit quantum circuit containing i_k operators and its equivalent operator (G).

(c) The representation of a multi qubit quantum circuit containing j_k qubits and using only one operator for each qubit and the equivalent operator (F) of the system.

Fig. 3. The representation of: (**a**) a single qubit circuit with one operator, (**b**) a single qubit circuit with more than one operator, and (**c**) a multi qubit circuit with one operator for each qubit.

Basics of Q/C-ANNs

In this part, the architecture of a quantum computing-based AI is talked. For this purpose, application of hybrid/classical ANNs is discussed, which have classical parts and also quantum computing-based operators. Generally, a Q/C-ANN includes three main parts, which can be named as Part 1, Part 2, and Part 3. For a more clarification, Fig. 4 shows the architecture of a Q/C-ANN.

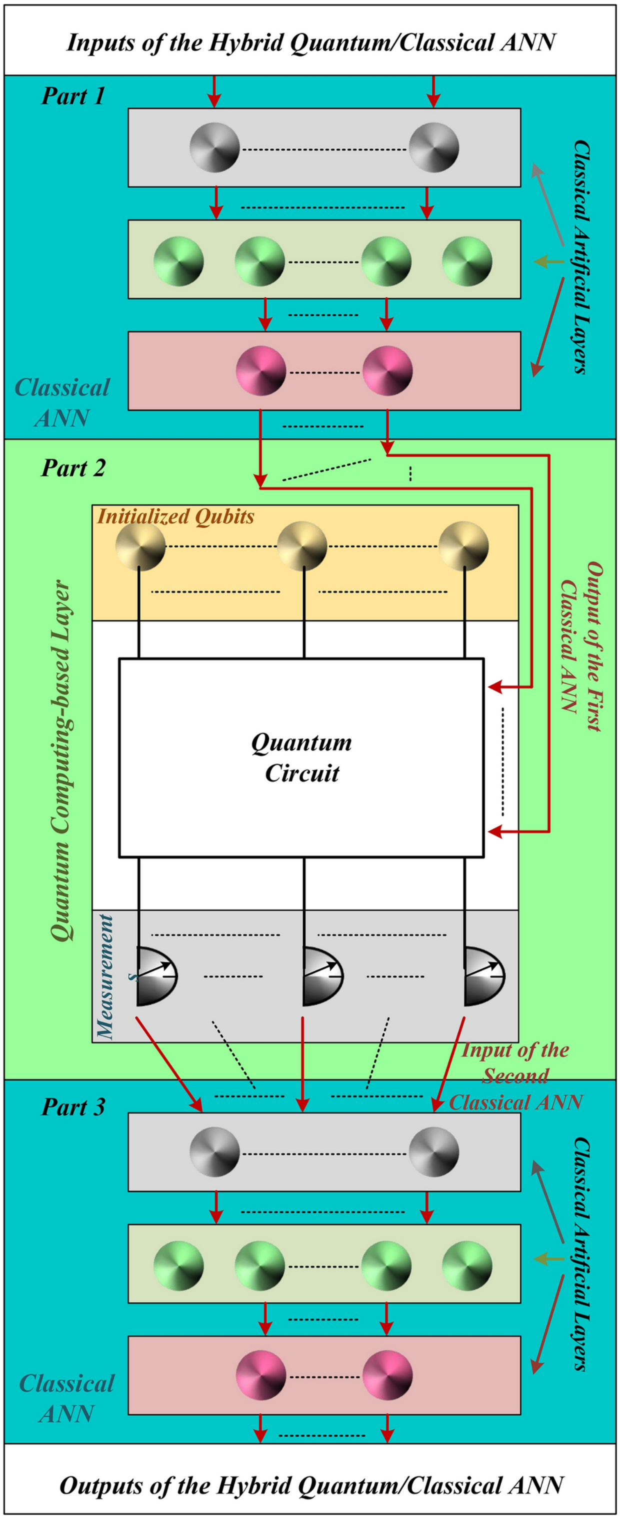

Inputs of the Hybrid Quantum/Classical ANN
Part 1
Classical Artificial Layers
Classical ANN
Part 2
Initialized Qubits
Quantum Computing-based Layer
Quantum Circuit
Output of the First Classical ANN
Measurement
Input of the Second Classical ANN
Part 3
Classical Artificial Layers
Classical ANN
Outputs of the Hybrid Quantum/Classical ANN

Fig. 4. The structure of a Q/C-ANN.

The main task of the first part is to receive the input signals of the hybrid network. In addition, Part 1 can contain classical layers to receive the inputs and produce the output signals of Part 1. Each classical layer can be made using artificial neurons, which receive input signals from the previous layer and produce the output signals considering weighting coefficients, the bias factor, and the activation function. Therefore, in Part 1, artificial neurons play the most important role. For more clarification, the output of a neuron can be updated as follows:

$$\mu_{out} = f\left(\chi\right). \tag{8}$$

Where, *f* is an activation function, and χ is as follows:

$$\chi = (\omega_1 \times \kappa_1 + \omega_2 \times \kappa_2 + \omega_3 \times \kappa_3 + \cdots + \omega_z \times \kappa_z) + \varrho. \tag{9}$$

In (9), z is the number neurons of the previous layer, which are connected to the desired neuron. In addition, ϱ is the bias factor, and ω_j is the weighting coefficient of the j^{th} input signal of the desired neuron. Also, κ_j is the j^{th} input signal of the desired neuron, which can be produced by the j^{th} neuron of the previous layer.

Also, Part 2 of Fig. 4 is a quantum computing-based layer, which has three sub-parts, i.e., initialized qubits, quantum circuit, and measurements. For the initialized qubits, different strategies can be done. For example, all the implemented qubits can be in state $|0\rangle$. Therefore, if Part 2 is considered a *n* qubit system with state $|0\rangle$ for all qubits, the initial state of the system ($S_{1,0}$) can be obtained as follows:

$$|S_{1,0}\rangle = \overbrace{|0\rangle \otimes |0\rangle \otimes \cdots \otimes |0\rangle}^{n}. \tag{10}$$

In other words:

$$|S_{1,0}\rangle = \overbrace{|00\cdots 0\rangle}^{n}. \tag{11}$$

And as a result:

$$|S_{1,0}\rangle = 2^n \left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \right. . \tag{12}$$

Also, if all the initialized qubits of the system are in state $|1\rangle$, the initial state of the system ($S_{1,1}$) is as follows:

$$|S_{1,1}\rangle = \overbrace{|1\rangle \otimes |1\rangle \otimes \cdots \otimes |1\rangle}^{n}. \tag{13}$$

Therefore,

$$|S_{1,1}\rangle = \overbrace{|11\cdots 1\rangle}^{n}. \tag{14}$$

Then,

$$|S_{1,1}\rangle = 2^n \left\{ \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \right. . \tag{15}$$

Furthermore, if each qubit is in an equal superposition, the overall initial state of the system ($|S_{1,e}\rangle$) can be as follows:

$$|S_{1,e}\rangle = \overbrace{\frac{(|0\rangle + |1\rangle)}{\sqrt{2}} \otimes \cdots \otimes \frac{(|0\rangle + |1\rangle)}{\sqrt{2}}}^{n}. \tag{16}$$

Also, (16) can be written as follows:

$$|S_{1,e}\rangle = 2^n \left\{ \begin{bmatrix} \frac{1}{\sqrt{2^n}} \\ \frac{1}{\sqrt{2^n}} \\ \frac{1}{\sqrt{2^n}} \\ \vdots \\ \frac{1}{\sqrt{2^n}} \end{bmatrix} \right. . \tag{17}$$

In addition, in Part 2, the quantum circuit can have quantum gates or operators, which receive the initialized qubits. There are different quantum gates, which can be represented by matrices. For example, a Pauli-X gate

($\mathbb{U}_X$), a Hadamard gate ($\mathbb{U}_H$), and a rotation gate about $\hat{y}$ axis ($\mathbb{U}_{Rot_y}(\varpi)$) can be represented by matrices as follows[46]:

$$\mathbb{U}_X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \tag{18}$$

$$\mathbb{U}_H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \tag{19}$$

and

$$\mathbb{U}_{Rot_y}(\varpi) = \begin{bmatrix} cos(\frac{\varpi}{2}) & -sin(\frac{\varpi}{2}) \\ sin(\frac{\varpi}{2}) & cos(\frac{\varpi}{2}) \end{bmatrix}. \tag{20}$$

It is important to note that, there are different quantum operators. For the rest of this part, the matrix representation of some important quantum operators will be talked about. Before, the matrix representation of Pauli-X gate, Hadamard gate, and rotation gate about $\hat{y}$ axis have been talked about. However, there are other quantum gates, e.g., Pauli-Y gate, Pauli-Z gate, swap gate, controlled-swap, controlled-Not gate, Toffoli gate, as well as rotation gates about $\hat{x}$ and $\hat{z}$ axes. Some gates are operated on single qubit such as Pauli-Y gate ($\mathbb{U}_Y$) and Pauli-Z gate ($\mathbb{U}_Z$), where the matrix representation of them is as follows[46]:

$$\mathbb{U}_Y = \begin{bmatrix} 0 & -\sqrt{-1} \\ \sqrt{-1} & 0 \end{bmatrix}, \tag{21}$$

and

$$\mathbb{U}_Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{22}$$

In addition, rotation gates about $\hat{x}$ and $\hat{z}$ axes ($\mathbb{U}_{Rot_x}(\varpi)$ and $\mathbb{U}_{Rot_z}(\varpi)$, respectively) can be considered as other examples of quantum operators for single qubits, where they can be represented mathematically as follows[46]:

$$\mathbb{U}_{Rot_x}(\varpi) = \begin{bmatrix} cos(\frac{\varpi}{2}) & -\sqrt{-1}sin(\frac{\varpi}{2}) \\ -\sqrt{-1}sin(\frac{\varpi}{2}) & cos(\frac{\varpi}{2}) \end{bmatrix}, \tag{23}$$

and

$$\mathbb{U}_{Rot_z}(\varpi) = \begin{bmatrix} e^{\frac{-\sqrt{-1}\varpi}{2}} & 0 \\ 0 & e^{\frac{\sqrt{-1}\varpi}{2}} \end{bmatrix}. \tag{24}$$

Furthermore, swap ($\mathbb{U}_{swap}$) and controlled-not ($\mathbb{U}_{C-N}$) gates operate on two qubits, and their matrix representations are as follows[46]:

$$\mathbb{U}_{swap} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{25}$$

and

$$\mathbb{U}_{C-N} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{26}$$

Besides, controlled-swap gate ($\mathbb{U}_{C-swap}$) and Toffoli gate ($\mathbb{U}_{Toffoli}$) are used for three qubis, where their matrix representations are as follows[46]:

$$\mathbb{U}_{C-swap} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{27}$$

and

$$\mathbb{U}_{Toffoli} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \tag{28}$$

In this part, the matrix representation of different quantum gates has been discussed. Some of the mentioned quantum gates such as Pauli-Y, Pauli-Z, and rotation gate gates can be operated on a single qubit. Also, the swap gate and controlled-not gate are operated on two qubits. In addition, controlled-swap gate and Toffoli gate are operated on three qubits. For more information about the mentioned gates and other quantum gates,[46] can be studied.

It is worth to mention that, for quantum gates, those can be considered which have parameters to tune them. Therefore, using and modifying the parameters of the operator gates, they can be used to support the hybrid network for the goal of the network (e.g., regression application). Finally, Part 3 of the hybrid network is a classical ANN. This part has classical layers, and it receives the measurements of the quantum computing-based layer. Then, the output of this part is the output of the hybrid network.

Time-series-based load forecasting using Q/C-ANNs

In this study, application of Q/C-ANNs is used for ultra short-term forecasting (one-step-ahead prediction) of loads and in a form of time-series. Therefore, if $P_L(t)$ is the value of the load at time = t, the set that includes the inputs of the hybrid network is as follows:

$$X = \{P_L(t - \Delta t), \quad P_L(t - 2\Delta t), \quad \cdots \quad , P_L(t - \beta\Delta t)\}. \tag{29}$$

Where, in (29), Δt and β are the sampling time and inputs delay, respectively. Also, for Part 1 and Part 3, classical layers including artificial neurons are implemented in a form of a feedforward structure-based neural network. Besides, for Part 2, initialized qubits are considered based on (10). In addition, for the quantum circuit, for each qubit, a rotation operator about $\hat{y}$ axis can be considered. In Fig. 5, the structure of a n qubit system including rotation gates is depicted. Based on (20), each rotation operator of Fig. 5 can be represented in a form of a matrix as follows:

$$R_y(\xi) = mathbbU_{Rot_y}(\xi) = \begin{bmatrix} cos(\frac{\xi}{2}) & -sin(\frac{\xi}{2}) \\ sin(\frac{\xi}{2}) & cos(\frac{\xi}{2}) \end{bmatrix}. \tag{30}$$

So, in a case of a n qubit system, the equivalent operator (R) of the quantum circuit can be as follows:

$$R = \overbrace{R_y(\xi_1)}^{\mathbb{U}_{Rot_y}(\xi_1)} \otimes \overbrace{R_y(\xi_2)}^{\mathbb{U}_{Rot_y}(\xi_2)} \otimes \cdots \otimes \overbrace{R_y(\xi_n)}^{\mathbb{U}_{Rot_y}(\xi_n)}. \tag{31}$$

Where, the rotation parameter ξ_i is related to the i^{th} qubit of the system.

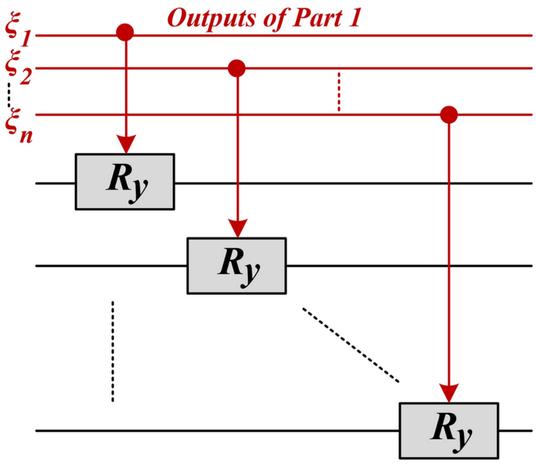


Fig. 5. The quantum circuit of a n qubit system, including a rotation gate ($R_y = \mathbb{U}_{Rot_y}$) for each qubit.

Therefore, using (30) and considering that each qubit is in initial state $|0\rangle$, the final state of each single qubit for $1 \leq i \leq n$ is as follows:

$$|\phi_i\rangle = \begin{bmatrix} cos(\frac{\xi_i}{2}) & -sin(\frac{\xi_i}{2}) \\ sin(\frac{\xi_i}{2}) & cos(\frac{\xi_i}{2}) \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \tag{32}$$

and in other words,

$$|\phi_i\rangle = \begin{bmatrix} cos(\frac{\xi_i}{2}) \\ sin(\frac{\xi_i}{2}) \end{bmatrix}. \tag{33}$$

Therefore, the final state of qubit *i* can be written as follows:

$$|\phi_i\rangle = \left(cos(\tfrac{\xi_i}{2})|0\rangle + sin(\tfrac{\xi_i}{2})|1\rangle\right). \tag{34}$$

Also, the vector corresponding to the final state of the system $|S_2\rangle$ can be calculated as follows:

$$|S_2\rangle = |\phi_1\rangle \otimes |\phi_2\rangle \otimes \cdots \otimes |\phi_{n-1}\rangle \otimes |\phi_n\rangle. \tag{35}$$

And considering (33) and (35), $|S_2\rangle$ can be obtained as follows:

$$|S_2\rangle = \begin{bmatrix} cos(\frac{\xi_1}{2}) \\ sin(\frac{\xi_1}{2}) \end{bmatrix} \otimes \begin{bmatrix} cos(\frac{\xi_2}{2}) \\ sin(\frac{\xi_2}{2}) \end{bmatrix} \otimes \cdots \otimes \begin{bmatrix} cos(\frac{\xi_{n-1}}{2}) \\ sin(\frac{\xi_{n-1}}{2}) \end{bmatrix} \otimes \begin{bmatrix} cos(\frac{\xi_n}{2}) \\ sin(\frac{\xi_n}{2}) \end{bmatrix}. \tag{36}$$

The output of the quantum computing-based layer depends on $\xi_1, \xi_2, \cdots, \xi_n$. The measured outputs of each qubit can be used as the inputs of the Part 3 in order to connect the quantum computing-based layer into the Part 3 of the Q/C-ANN.

For more clarification, in continuation of this part, it will be shown that how a Q/C-ANN can predict the future value of a load. So, consider a Q/C-ANN, which has N_{in} inputs ($\beta = N_{in}$). Also, for this hybrid network (Q/C-ANN), each classical part (Part 1 and Part 3) has one classical layer. In addition, the numbers of classical neurons for Part 1 and Part 3 are $N_{n,1}$ and $N_{n,3}$. Then, based on (29), the vector related to the input dataset of this hybrid network is modified as follows:

$$X = [P_L(t) \quad P_L(t-\Delta t) \quad \cdots \quad P_L(t-(N_{in}-1)\Delta t)]. \tag{37}$$

Also, the output of this hybrid network is as follows:

$$Y = [P_L(t+\Delta t)]. \tag{38}$$

Considering the mentioned parameters and the definition of the inputs and the output of Q/C-ANN, the relation between the inputs and the output of the system can be explained. Therefore, for the rest of this part, the calculations related to Part 1, Part 2, and Part 3 will be discussed to show how the hybrid network can be used to predict the desired data. To achieve this goal, Subsection 4.1, Subsection 4.2, and Subsection 4.3 will talk about Part 1, Part2, and Part 3 of Q/C-ANN, respectively. The next part will show how the output signals of Part 1 can be calculated.

Calculation the outputs of part 1

Part 1 is a classical layer, which includes neurons. In addition, the input vector of Part 1 (I_1) is fed by the inputs of the hybrid network. In other words:

$$I_1 = X^T. \tag{39}$$

Also, each neuron of Part 1 is connected to the inputs of the hybrid network. In addition, each neuron has weighting coefficients and a bias factor. The matrix representation of weighting coefficients of Part 1 is as follows:

$$W_1 = \begin{bmatrix} w_{1,1,1} & w_{1,1,2} & w_{1,1,3} & \cdots & w_{1,1,N_{in}} \\ w_{1,2,1} & w_{1,2,2} & w_{1,2,3} & \cdots & w_{1,2,N_{in}} \\ w_{1,3,1} & w_{1,3,2} & w_{1,3,3} & \cdots & w_{1,3,N_{in}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ w_{1,N_{n,1},1} & w_{1,N_{n,1},2} & w_{1,N_{n,1},3} & \cdots & w_{1,N_{n,1},N_{in}} \end{bmatrix}. \tag{40}$$

Where, $w_{1,j,k}$ is the weighting coefficient between the j^{th} neuron of the classical layer of Part 1 and the k^{th} input of the hybrid network ($P_L(t-(k-1)\Delta t)$). Also, the vector corresponding to the bias factors of Part 1 can be represented as follows:

$$B_1 = \begin{bmatrix} b_{1,1} \\ b_{1,2} \\ \vdots \\ b_{1,N_{n,1}-1} \\ b_{1,N_{n,1}} \end{bmatrix}. \tag{41}$$

In (41), $b_{1,j}$ is the bias factor of the j^{th} neuron of the classical layer of Part 1. Therefore, by the implementation of (39), (40), and (41), the output of the classical layer without the implementation of the activation function of Part 1 is as follows:

$$P_1 = W_1 \times I_1 + B_1 \tag{42}$$

Also, P_1 can be represented in a form of a vector as follows:

$$P_1 = \begin{bmatrix} p_{1,1} \\ p_{1,2} \\ \vdots \\ p_{1,N_{n,1}-1} \\ p_{1,N_{n,1}} \end{bmatrix}. \tag{43}$$

Using an activation function f_1, the i^{th} output of Part 1 is $f_1(p_{1,i})$. Then, the vector corresponding to the outputs of Part 1 is as follows:

$$O_1 = \begin{bmatrix} f_1(p_{1,1}) \\ f_1(p_{1,2}) \\ \vdots \\ f_1(p_{1,N_{n,1}-1}) \\ f_1(p_{1,N_{n,1}}) \end{bmatrix}. \tag{44}$$

The output signals of Part 1 will be used as the input signals of Part 2. In other words, the input signals of Part 2 will be used to update the state of the qubits using rotation gates, which are deployed to structure the quantum circuit in Part 2. The next part will discuss how the output signals of Part 1 can be used in Part 2 to update the state of the qubits and how it can produce the output signals of Part 2.

Calculation the outputs of part 2

As discussed above, the outputs of Part 1 modify the rotation parameters of Part 2. Therefore, the input vector of Part 2 (I_2) is as follows:

$$I_2 = O_1. \tag{45}$$

In other words, the i^{th} rotation parameter is as follows:

$$\xi_i = f_1(p_{1,i}), \tag{46}$$

and the vector representation related to the rotation parameters of Part 2 is as follows:

$$\Xi = \begin{bmatrix} \xi_1 \\ \xi_2 \\ \vdots \\ \xi_{N_{n,1}-1} \\ \xi_{N_{n,1}} \end{bmatrix}. \tag{47}$$

Where, using (46) and (47), Ξ can be obtained. Also, if the initial state of the i^{th} qubit is $|0\rangle$, based on (34) and (46), the final state of the i^{th} qubit is as follows:

$$|\phi_i\rangle = \left(cos(\frac{f_1(p_{1,i})}{2})|0\rangle + sin(\frac{f_1(p_{1,i})}{2})|1\rangle\right). \tag{48}$$

In other words and based on (33),

$$|\phi_i\rangle = \begin{bmatrix} cos(\frac{f_1(p_{1,i})}{2}) \\ sin(\frac{f_1(p_{1,i})}{2}) \end{bmatrix}. \tag{49}$$

Therefore, (49) can be encoded to be used as the input for Part 3. So, the following dataset (matrix O_2) can be considered to create the outputs of Part 2.

$$O_2 = \begin{bmatrix} cos(\frac{f_1(p_{1,1})}{2}) & sin(\frac{f_1(p_{1,1})}{2}) \\ cos(\frac{f_1(p_{1,2})}{2}) & sin(\frac{f_1(p_{1,2})}{2}) \\ \vdots & \vdots \\ cos(\frac{f_1(p_{1,N_{n,1}-1})}{2}) & sin(\frac{f_1(p_{1,N_{n,1}-1})}{2}) \\ cos(\frac{f_1(p_{1,N_{n,1}})}{2}) & sin(\frac{f_1(p_{1,N_{n,1}})}{2}) \end{bmatrix}. \tag{50}$$

The next part will discuss that how the outputs of Part 2 can be encoded to provide the inputs of Part 3. Then, it will talk about how the encoded data can be used to obtain the output of Part 3 and as a result the output of Q/C-ANN.

Calculation the output of part 3

The output of Part 2 is based on (49) and (50). As mentioned above, the input of Part 3 is the encoded version of (49) and (50). In this work, the elements related to state $|0\rangle$ ($cos(\frac{f_1(p_{1,i})}{2})$) is used as the input of Part 3 and as a result, the first column of (50) is used as the input of Part 3. So, the vector representation of the inputs for Part 3 is as follows:

$$I_3 = \begin{bmatrix} cos(\frac{f_1(p_{1,1})}{2}) \\ cos(\frac{f_1(p_{1,2})}{2}) \\ \vdots \\ cos(\frac{f_1(p_{1,N_{n,1}-1})}{2}) \\ cos(\frac{f_1(p_{1,N_{n,1}})}{2}) \end{bmatrix}. \tag{51}$$

In addition, the matrix representation of the weighting coefficients (W_3) and the vector representation of the bias factors of the classical layer of Part 3 are as follows:

$$W_3 = \begin{bmatrix} w_{3,1,1} & w_{3,1,2} & w_{3,1,3} & \cdots & w_{3,1,N_{n,1}} \\ w_{3,2,1} & w_{3,2,2} & w_{3,2,3} & \cdots & w_{3,2,N_{n,1}} \\ w_{3,3,1} & w_{3,3,2} & w_{3,3,3} & \cdots & w_{3,3,N_{n,1}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ w_{3,N_{n,3},1} & w_{3,N_{n,3},2} & w_{3,N_{n,3},3} & \cdots & w_{3,N_{n,3},N_{n,1}} \end{bmatrix}, \tag{52}$$

$$B_3 = \begin{bmatrix} b_{3,1} \\ b_{3,2} \\ \vdots \\ b_{3,N_{n,3}-1} \\ b_{3,N_{n,3}} \end{bmatrix}. \tag{53}$$

Where, in (52) and (53), $w_{3,j,k}$ and $b_{3,j}$ are the weighting coefficient to connect the k^{th} input of Part 3 to the j^{th} neuron of the classical layer of Part 3, and the bias factor of the j^{th} mentioned neuron. So, based on (51), (52), and (53), the output of the classical layer of Part 3 without the deployment of activation functions can be calculated as follows:

$$P_3 = W_3 \times I_3 + B_3. \tag{54}$$

Where, the vector representation of P_3 can be written as follows:

$$P_3 = \begin{bmatrix} p_{3,1} \\ p_{3,2} \\ \vdots \\ p_{3,N_{n,3}-1} \\ p_{3,N_{n,3}} \end{bmatrix}, \tag{55}$$

and using an activation function f_3 for this layer, the output vector of this layer (O_3) is as follows:

$$O_3 = \begin{bmatrix} f_3(p_{3,1}) \\ f_3(p_{3,2}) \\ \vdots \\ f_3(p_{3,N_{n,3}-1}) \\ f_3(p_{3,N_{n,3}}) \end{bmatrix}. \tag{56}$$

Also, the main task of Part 3 is to produce the output of the hybrid network and as a result the prediction of the value of the load. So, in this part, in addition to the previous classical layer, a classical output layer is existed. The output of this layer is the prediction of P_L at time $t + \Delta t$. Therefore, the output layer of Part 3 has only one neuron and as a result one bias factor (b_o) for that neuron. In addition, the weighting coefficients representation of this layer can be shown by a vector as follows:

$$W_o = [w_{o,1} \quad w_{o,2} \quad \cdots \quad w_{o,N_{n,3}-1} \quad w_{o,N_{n,3}}]. \tag{57}$$

Finally, using a linear activation function, the output of the hybrid network can be calculated as follows:

$$P_o = W_o \times O_3 + b_o. \tag{58}$$

This part discussed that how the output of Q/C-ANN can be calculated. In the next part, a brief discussion about load forecasting using Q/C-ANN is provided. The next part shows how the three parts of Q/C-ANN can be used in a unified algorithm (i.e., Algorithm 1) to produce the output of the hybrid network in a nutshell.

The Implemented Strategy in a Nutshell

Briefly, the output of the hybrid network is P_o. The hybrid network receives X in (37). Then, using (39), (42), (43), and (44), the outputs of Part 1 is calculated. The outputs of Part 1 is used as the inputs of Part 2 to tune rotation parameters of the quantum circuit. Therefore, based on (46), (47), (48), (49), and (50), the outputs of Part 2 is determined. After that, the outputs of Part 2 are encoded to the inputs of Part 3 using (51). Then, using (54), (55), and (56), the output signals of the neurons of the classical layer of Part 3 can be obtained. Finally, using (58), the output of the hybrid network (predicted value of the load at time = $t + \Delta t$ ($\overline{P}_L(t + \Delta t)$)) is calculated.

```
Step 1. Gathering Data: Gathering the value of the
  load ($P_L$) at different times.
Step 2. Data Pre-Processing: E.g., data cleaning,
  filling missing data, and data normalization.
Step 3. Time-Series Dataset: Creating an input and a
  output dataset in a form of a time-series considering
  the inputs delay ($\beta$).
Step 4. Tuning of the Parameters: Optimizing the
  parameters ($W_1$, $W_3$, $W_o$, $B_1$, $B_3$, and $b_o$) of the
  hybrid network to find the well-tuned values of them.
Step 5. Predicting the value of the load:
5.1 Calculation of $O_1$ using $I_1$, (42), (43), and (44)
5.2 $I_2 = O_1$
5.3 Calculation of $O_2$ using $I_2$, (46), (47), (48), (49),
  and (50)
5.4 $I_3 =$ the encoded version of $O_2$ using (51)
5.5 Calculation of $O_3$ using $I_3$, (54), (55), and (56)
5.6 Calculation of $P_o$ using (58)
5.7 $\overline{P}_L(t + \Delta t) = P_o$
```

Algorithm 1. Load Forecasting based on Q/C-ANN.

In addition to the inputs of the hybrid network, there are other parameters, which should be known before the calculation of the output of the hybrid network. This parameters are the parameters related to the weighting coefficients and the bias factors, and they can be shown by (40), (41), (52), (53), (57), and b_o. These parameters can be tuned using training process. For more clarification about the implementation of Q/C-ANN, Algorithm 1 shows the steps to estimate the future value of the load.

More math-based discussions

In this section, the implementation of the classical data to update the states of the qubits of the quantum layer, quantum measurement and the relation between that and the output of Part 2 of Q/C-ANN, different encoding techniques to map classical data into quantum states, and data pre-processing will be discussed in a more detail and mathematically.

Updating the states of the qubits

In this part, the implementation of the classical data (the produced classical data by Part 1 of Q/C-ANN) to update the initialized qubits (with general states) of the quantum circuit will be clarified mathematically. In a general form, for the quantum circuit of Q/C-ANN with n qubits, the initial state of the i^{th} qubit (for $1 \leq i \leq n$) can be considered as follows:

$$|\gamma_{in,i}\rangle = \alpha_{1,i}|0\rangle + \alpha_{2,i}|1\rangle. \tag{59}$$

Where, based on (2),

$$|\alpha_{1,i}|^2 + |\alpha_{2,i}|^2 = 1. \tag{60}$$

Also, as mentioned before, the quantum circuit is made using rotation operators. Then, the final state of each single qubit can be calculated as follows:

$$|\gamma_{out,i}\rangle = R_y(f_1(p_1,i)) \times |\gamma_{in,i}\rangle. \tag{61}$$

In other words,

$$|\gamma_{out,i}\rangle = \begin{bmatrix} cos(\frac{f_1(p_1,i)}{2}) & -sin(\frac{f_1(p_1,i)}{2}) \\ sin(\frac{f_1(p_1,i)}{2}) & cos(\frac{f_1(p_1,i)}{2}) \end{bmatrix} \times \begin{bmatrix} \alpha_{1,i} \\ \alpha_{2,i} \end{bmatrix}. \tag{62}$$

So, if

$$|\gamma_{out,i}\rangle = \begin{bmatrix} \beta_{1,i} \\ \beta_{2,i} \end{bmatrix}, \tag{63}$$

then,

$$\beta_{1,i} = \alpha_{1,i} \times cos(\frac{f_1(p_1,i)}{2}) - \alpha_{2,i} \times sin(\frac{f_1(p_1,i)}{2}), \tag{64}$$

and

$$\beta_{2,i} = \alpha_{1,i} \times sin(\frac{f_1(p_1,i)}{2}) + \alpha_{2,i} \times cos(\frac{f_1(p_1,i)}{2}). \tag{65}$$

Therefore, the final state of each single qubit (for example qubit *i*) can be obtained using (64) and (65). In addition, using (5) and (31), if $U = R$, the final state of the system in the form of a *n* qubits system can be calculated as follows:

$$|\Gamma_{out}\rangle = R \times |\Gamma_{in}\rangle. \tag{66}$$

Also, the initial and the final states (i.e., $|\Gamma_{in}\rangle$ and $|\Gamma_{out}\rangle$, respectively) of the quantum circuit of Q/C-ANN can be written in the form of *n* qubits system in a general form as follows:

$$|\Gamma_{in}\rangle = \overbrace{|\gamma_{in,1}\rangle \otimes |\gamma_{in,2}\rangle \otimes \cdots \otimes |\gamma_{in,n}\rangle}^{n}, \tag{67}$$

and

$$|\Gamma_{out}\rangle = \overbrace{|\gamma_{out,1}\rangle \otimes |\gamma_{out,2}\rangle \otimes \cdots \otimes |\gamma_{out,n}\rangle}^{n}. \tag{68}$$

Then,

$$|\Gamma_{in}\rangle = \overbrace{\begin{bmatrix} \alpha_{1,1} \\ \alpha_{2,1} \end{bmatrix} \otimes \begin{bmatrix} \alpha_{1,2} \\ \alpha_{2,2} \end{bmatrix} \otimes \cdots \otimes \begin{bmatrix} \alpha_{1,n} \\ \alpha_{2,n} \end{bmatrix}}^{n}, \tag{69}$$

and

$$|\Gamma_{out}\rangle = \overbrace{\begin{bmatrix} \beta_{1,1} \\ \beta_{2,1} \end{bmatrix} \otimes \begin{bmatrix} \beta_{1,2} \\ \beta_{2,2} \end{bmatrix} \otimes \cdots \otimes \begin{bmatrix} \beta_{1,n} \\ \beta_{2,n} \end{bmatrix}}^{n}. \tag{70}$$

In this part, the mathematical-based representation to deploy the output signals (which are classical data) of Part 1 of Q/C-ANN to be implemented for the rotation gates to update the states of the initialized qubits of the system have been explained. To achieve a more efficient clarification, the states of the initialized qubits have been considered in a general form.

Quantum measurement and the outputs of the quantum layer

Measurements play a key role in a quantum circuit by considering the desired basis and using the result for different purposes, e.g, coding/decoding, interpretation, or transferring them to other units. A general description of a measurement can be demonstrated as follows[46,47]:

$$|\Gamma_M\rangle = \frac{M_O|\Gamma\rangle}{\sqrt{\langle\Gamma|M_O^\dagger M_O|\Gamma\rangle}}, \tag{71}$$

and

$$Prob(O) = \langle\Gamma|M_O^\dagger M_O|\Gamma\rangle. \tag{72}$$

Where, M_O is the operator of the measurement with the possible result *O*, $|\Gamma_M\rangle$ is the new state if the measurement result is *O*, and *Prob*(*O*) is the probability corresponded that the measurement reach result *O*. Considering the case of projective measurements, a projection operator corresponding to measurement result $|j\rangle$ is Prj_j where $Prj_j^\dagger = Prj_j$, $Prj_j^2 = Prj_j$, and as a result[46,47]:

$$|\Gamma_j\rangle = \frac{Prj_j|\Gamma\rangle}{\sqrt{\langle\Gamma|Prj_j|\Gamma\rangle}}, \tag{73}$$

and

$$Prob(j) = \langle\Gamma|Prj_j|\Gamma\rangle. \tag{74}$$

A projective measurement can be characterized by a complete set of orthogonal projectors, and if the measurement is done with respect to z axis, the complete set of the orthogonal projectors is $\{Prj_0, Prj_1\}$[46,47]:

$$Prj_0 = |0\rangle\langle 0|, \tag{75}$$

and

$$Prj_1 = |1\rangle\langle 1|. \tag{76}$$

Considering the measurement corresponded to z axis of the i^{th} qubit, and if *Prob*(*j*) is called r_{ij} (for $j \in \{0,1\}$), (74) can be written as follows:

$$Prob(0) = r_{i0} = \langle\gamma_{out,i}|Prj_0|\gamma_{out,i}\rangle, \tag{77}$$

and

$$Prob(1) = r_{i1} = \langle\gamma_{out,i}|Prj_1|\gamma_{out,i}\rangle. \tag{78}$$

Then considering that $|\gamma_{out,i}\rangle = \beta_{1,i}|0\rangle + \beta_{2,i}|1\rangle$, and using (75) and (76), (77) and (78) can be updated as follows:

$$r_{i0} = (\overline{\beta_{1,i}}\langle 0| + \overline{\beta_{2,i}}\langle 1|)(|0\rangle\langle 0|)(\beta_{1,i}|0\rangle + \beta_{2,i}|1\rangle), \tag{79}$$

and

$$r_{i1} = (\overline{\beta_{1,i}}\langle 0| + \overline{\beta_{2,i}}\langle 1|)(|1\rangle\langle 1|)(\beta_{1,i}|0\rangle + \beta_{2,i}|1\rangle). \tag{80}$$

In other words:

$$r_{i0} = (\overline{\beta_{1,i}}\langle 0|0\rangle\langle 0| + \overline{\beta_{2,i}}\langle 1|0\rangle\langle 0|)(\beta_{1,i}|0\rangle + \beta_{2,i}|1\rangle), \tag{81}$$

and

$$r_{i1} = (\overline{\beta_{1,i}}\langle 0|1\rangle\langle 1| + \overline{\beta_{2,i}}\langle 1|1\rangle\langle 1|)(\beta_{1,i}|0\rangle + \beta_{2,i}|1\rangle). \tag{82}$$

Where, $\overline{\beta_{1,i}}$, $\overline{\beta_{2,i}}$, and more generally $\overline{\iota}$ denote the complex conjugate of $\beta_{1,i}$, $\beta_{2,i}$, and ι, respectively. In addition, $|0\rangle$ and $|1\rangle$ are orthonormal basis and as a result, $\langle 0|0\rangle = \langle 1|1\rangle = 1$ and $\langle 0|1\rangle = \langle 1|0\rangle = 0$[46,47]. Therefore, using (64) and (65), (81) and (82) are modified as follows:

$$\begin{aligned} r_{i0} = (&\overline{(\alpha_{1,i} \times cos(\frac{f_1(p_1,i)}{2}) - \alpha_{2,i} \times sin(\frac{f_1(p_1,i)}{2}))}\langle 0|) \times \ldots \\ &((\alpha_{1,i} \times cos(\frac{f_1(p_1,i)}{2}) - \alpha_{2,i} \times sin(\frac{f_1(p_1,i)}{2}))|0\rangle + \ldots \\ &(\alpha_{1,i} \times sin(\frac{f_1(p_1,i)}{2}) + \alpha_{2,i} \times cos(\frac{f_1(p_1,i)}{2}))|1\rangle), \end{aligned} \tag{83}$$

and

$$\begin{aligned} r_{i1} = (&\overline{(\alpha_{1,i} \times sin(\frac{f_1(p_1,i)}{2}) + \alpha_{2,i} \times cos(\frac{f_1(p_1,i)}{2}))}\langle 1|) \times \ldots \\ &((\alpha_{1,i} \times cos(\frac{f_1(p_1,i)}{2}) - \alpha_{2,i} \times sin(\frac{f_1(p_1,i)}{2}))|0\rangle + \ldots \\ &(\alpha_{1,i} \times sin(\frac{f_1(p_1,i)}{2}) + \alpha_{2,i} \times cos(\frac{f_1(p_1,i)}{2}))|1\rangle). \end{aligned} \tag{84}$$

Again by considering that $|1\rangle$ and $|0\rangle$ are orthonormal basis, the updated version of (83) and (84) are as follows:

$$\begin{aligned} r_{i0} = &\overline{(\alpha_{1,i} \times cos(\frac{f_1(p_1,i)}{2}) - \alpha_{2,i} \times sin(\frac{f_1(p_1,i)}{2}))} \times \ldots \\ &(\alpha_{1,i} \times cos(\frac{f_1(p_1,i)}{2}) - \alpha_{2,i} \times sin(\frac{f_1(p_1,i)}{2})) \end{aligned} \tag{85}$$

and

$$\begin{aligned} r_{i1} = &\overline{(\alpha_{1,i} \times sin(\frac{f_1(p_1,i)}{2}) + \alpha_{2,i} \times cos(\frac{f_1(p_1,i)}{2}))} \times \ldots \\ &(\alpha_{1,i} \times cos(\frac{f_1(p_1,i)}{2}) - \alpha_{2,i} \times sin(\frac{f_1(p_1,i)}{2})) \end{aligned} \tag{86}$$

Additionally, considering that $\overline{\Lambda}\Lambda = |\Lambda|^2$[48], by defining Λ_0 as

$$\Lambda_0 = \alpha_{1,i} \times cos(\frac{f_1(p_1,i)}{2}) - \alpha_{2,i} \times sin(\frac{f_1(p_1,i)}{2}), \tag{87}$$

and defining Λ_1 as

$$\Lambda_1 = \alpha_{1,i} \times sin(\frac{f_1(p_1,i)}{2}) + \alpha_{2,i} \times cos(\frac{f_1(p_1,i)}{2}), \tag{88}$$

(85) and (86) can be modified as follows:

$$r_{i0} = \overline{\Lambda_0}\Lambda_0 = |\Lambda_0|^2, \tag{89}$$

and

$$r_{i1} = \overline{\Lambda_1}\Lambda_1 = |\Lambda_1|^2. \tag{90}$$

In other words:

$$r_{i0} = \left|\alpha_{1,i} \times cos(\frac{f_1(p_1,i)}{2}) - \alpha_{2,i} \times sin(\frac{f_1(p_1,i)}{2})\right|^2, \tag{91}$$

and

$$r_{i1} = \left|\alpha_{1,i} \times sin(\frac{f_1(p_1,i)}{2}) + \alpha_{2,i} \times cos(\frac{f_1(p_1,i)}{2})\right|^2. \tag{92}$$

Besides, if the initial state of each qubit is in a pure state $|0\rangle$ (i.e., $\alpha_{1,i} = 1$ and $\alpha_{2,i} = 0$):

$$r_{i0} = \left|cos(\frac{f_1(p_1,i)}{2})\right|^2, \tag{93}$$

and

$$r_{i1} = \left| sin(\frac{f_1(p_1, i)}{2}) \right|^2 . \tag{94}$$

Therefore,

$$cos(\frac{f_1(p_1, i)}{2}) = \sqrt{r_{i0}}, \tag{95}$$

and

$$sin(\frac{f_1(p_1, i)}{2}) = \sqrt{r_{i1}}. \tag{96}$$

Finally, $\sqrt{r_{i0}}$ for $1 \leq i \leq N_{n,1}$, can be deployed to construct the input vector (i.e., I_3 in (51)) of the next layer (i.e., Part 3). More mathematically, (51) can be updated as follows:

$$I_3 = \begin{bmatrix} cos(\frac{f_1(p_{1,1})}{2}) \\ cos(\frac{f_1(p_{1,2})}{2}) \\ \vdots \\ cos(\frac{f_1(p_{1,N_{n,1}})}{2}) \end{bmatrix} = \begin{bmatrix} \sqrt{r_{10}} \\ \sqrt{r_{20}} \\ \vdots \\ \sqrt{r_{N_{n,1}0}} \end{bmatrix} \tag{97}$$

Briefly, in this part, the measurement approach has been explained, and it has been discussed with detail that how the measurement characteristics can be mapped into the classical data to be used for the next layer. The next part will discuss that how a classical data is mapped into a quantum state. Besides, other different encoding techniques to map classical data into quantum states with details will be explained. In addition, to have a more clarification, some examples will be solved, which can help to have a better understanding about encoding classical data into the quantum states.

Classical data mapping into quantum states

In this study, an angle encoding technique is deployed to map the classical output data of Part 1 of QC-ANN into the quantum states in Part 2 of QC-ANN. Generally, there are different encoding approaches that can be implemented to encode classical data into quantum states, and as a simple example of an encoding technique, basis encoding or mapping can be mentioned to map classical data into quantum data. In this approach, a binary string can be mapped into a quantum state with orthonormal basis $|0\rangle$ and $|1\rangle$ like mapping 0 to $|0\rangle$ and 1 to $|1\rangle$[49,50]. For instance, 123, 1237, and 111111 in the binary number systems are equivalent to 1111011_2 , 10011010101_2, and 11011001000000111_2, respectively. herefore, they can be encoded to the respective quantum states as follows:

$$\overbrace{123}^{\text{Classical Data}} \rightarrow \overbrace{|1111011\rangle}^{\text{Quantum State}} , \tag{98}$$

$$\overbrace{1237}^{\text{Classical Data}} \rightarrow \overbrace{|10011010101\rangle}^{\text{Quantum State}}, \tag{99}$$

$$\overbrace{111111}^{\text{Classical Data}} \rightarrow \overbrace{|11011001000000111\rangle}^{\text{Quantum State}} . \tag{100}$$

As another encoding technique, quantum associative memory encoding method can be mentioned. In this approach, a vector of classical data can be mapped into an equal superposition of different states corresponded to each classical data[51,52]. In other words, each classical data can be converted to the equivalent number in basis 2 (i.e., binary number). Then, an equal superposition of different states related each binary string represents the encoded quantum state. As an example, each element of a vector of classical data $V_{data} = [4004 \quad 4080 \quad 4081 \quad 4090]$ can be written in the binary system, i.e., $4004 = 111110100100_2$, $4080 = 111111110000_2$, $4081 = 111111110001_2$, and $4090 = 111111111010_2$. Then, the quantum encoded version of the classical dataset V_{data} is as follows:

$$\begin{aligned} |QV_{data}\rangle = & \frac{|111110100100\rangle}{2} + \frac{|111111110000\rangle}{2} \\ & ... + \frac{|111111110001\rangle}{2} + \frac{|111111111010\rangle}{2}. \end{aligned} \tag{101}$$

Therefore, the vector of the classical data V_{data} is mapped into a mixed quantum state in an equal superposition as follows:

$$\overbrace{V_{data}}^{\text{Classical Dataset}} \rightarrow \overbrace{|QV_{data}\rangle}^{\text{Quantum State}} \quad . \tag{102}$$

In this study, the quantum circuit is made based on rotation gates and as a result, an angle encoding approach is automatically implemented to map classical data into the state of the qubits by updating the state of each initialized qubit using a rotation operator with an angle that is the classical output data from Part 1. In this method, a classical data such as κ can be implemented to provide the angle of a rotation gate[53]. Then, the rotation gate can be used to map a classical data into a quantum state. In other words, if the initial state of a qubit is a known state $|Q_{In}\rangle$, then by applying a rotation gate about $\hat{y}$ axis with angle κ, the new state is as follows:

$$|Q_{In}\rangle \xrightarrow{R_y(\kappa)} |Q_S\rangle. \tag{103}$$

In other words:

$$\overbrace{\begin{bmatrix} q_{S0} \\ q_{S1} \end{bmatrix}}^{|Q_S\rangle} = \overbrace{\begin{bmatrix} cos(\frac{\kappa}{2}) & -sin(\frac{\kappa}{2}) \\ sin(\frac{\kappa}{2}) & cos(\frac{\kappa}{2}) \end{bmatrix}}^{R_y(\kappa)} \times \overbrace{\begin{bmatrix} q_{In0} \\ q_{In1} \end{bmatrix}}^{|Q_{In}\rangle}, \tag{104}$$

So, the classical data κ can be encoded into a quantum state as follows:

$$\overbrace{\kappa}^{\text{Classical Data}} \xrightarrow{\overbrace{R_y(\kappa), |Q_{In}\rangle}^{\text{Rotation Gate, and Initialized Qubit}}} \overbrace{|Q_S\rangle}^{\text{Quantum State}} \quad . \tag{105}$$

For a more clarification regarding the angle encoding, two examples are provided with different initial states as follows:

Example 1: In this example, the initial state of the qubit is $|0\rangle$. Therefore, $q_{In0} = 1$ and $q_{In1} = 0$. So, to map $\kappa_1 = \frac{8\pi}{17}$, $\kappa_2 = \frac{3\pi}{7}$, $\kappa_3 = \frac{\pi}{12}$, and $\kappa_4 = \frac{3\pi}{10}$ to the corresponding quantum states, $R_y(\frac{8\pi}{17})$, $R_y(\frac{3\pi}{7})$, $R_y(\frac{\pi}{12})$, and $R_y(\frac{3\pi}{10})$ can be applied as follows:

$$|Q_{S1}\rangle = \begin{bmatrix} q_{S01} \\ q_{S11} \end{bmatrix} = \begin{bmatrix} cos(\frac{8\pi}{34}) & -sin(\frac{8\pi}{34}) \\ sin(\frac{8\pi}{34}) & cos(\frac{8\pi}{34}) \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \tag{106}$$

$$|Q_{S2}\rangle = \begin{bmatrix} q_{S02} \\ q_{S12} \end{bmatrix} = \begin{bmatrix} cos(\frac{3\pi}{14}) & -sin(\frac{3\pi}{14}) \\ sin(\frac{3\pi}{14}) & cos(\frac{3\pi}{14}) \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \tag{107}$$

$$|Q_{S3}\rangle = \begin{bmatrix} q_{S03} \\ q_{S13} \end{bmatrix} = \begin{bmatrix} cos(\frac{\pi}{24}) & -sin(\frac{\pi}{24}) \\ sin(\frac{\pi}{24}) & cos(\frac{\pi}{24}) \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \tag{108}$$

and

$$|Q_{S4}\rangle = \begin{bmatrix} q_{S04} \\ q_{S14} \end{bmatrix} = \begin{bmatrix} cos(\frac{3\pi}{20}) & -sin(\frac{3\pi}{20}) \\ sin(\frac{3\pi}{20}) & cos(\frac{3\pi}{20}) \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \tag{109}$$

So,

$$|Q_{S1}\rangle = cos(\frac{8\pi}{34})|0\rangle + sin(\frac{8\pi}{34})|1\rangle \approx \begin{bmatrix} 0.7390 \\ 0.6737 \end{bmatrix} \tag{110}$$

$$|Q_{S2}\rangle = cos(\frac{3\pi}{14})|0\rangle + sin(\frac{3\pi}{14})|1\rangle \approx \begin{bmatrix} 0.7818 \\ 0.6235 \end{bmatrix} \tag{111}$$

$$|Q_{S3}\rangle = cos(\frac{\pi}{24})|0\rangle + sin(\frac{\pi}{24})|1\rangle \approx \begin{bmatrix} 0.9914 \\ 0.1305 \end{bmatrix} \tag{112}$$

and

$$|Q_{S4}\rangle = cos(\frac{3\pi}{20})|0\rangle + sin(\frac{3\pi}{20})|1\rangle \approx \begin{bmatrix} 0.8910 \\ 0.4540 \end{bmatrix} \tag{113}$$

Therefore, $K = \begin{bmatrix} \kappa_1 \\ \kappa_2 \\ \kappa_3 \\ \kappa_4 \end{bmatrix}$ is encoded to a quantum state as follows:

$$K = \overbrace{\begin{bmatrix}\kappa_1\\ \kappa_2\\ \kappa_3\\ \kappa_4\end{bmatrix}}^{\text{Classical Dataset}} \rightarrow \overbrace{|Q_{S1}\rangle \otimes |Q_{S2}\rangle \otimes |Q_{S3}\rangle \otimes |Q_{S4}\rangle}^{\text{Quantum State}}. \tag{114}$$

In other words:

$$\overbrace{\begin{bmatrix}\kappa_1 = \frac{8\pi}{17}\\ \kappa_2 = \frac{3\pi}{7}\\ \kappa_3 = \frac{\pi}{12}\\ \kappa_4 = \frac{3\pi}{10}\end{bmatrix}}^{\text{Classical Dataset}} \rightarrow \overbrace{\overbrace{\begin{bmatrix}0.7390\\0.6737\end{bmatrix}}^{|Q_{S1}\rangle} \otimes \overbrace{\begin{bmatrix}0.7818\\0.6235\end{bmatrix}}^{|Q_{S2}\rangle} \otimes \overbrace{\begin{bmatrix}0.9914\\0.1305\end{bmatrix}}^{|Q_{S3}\rangle} \otimes \overbrace{\begin{bmatrix}0.8910\\0.4540\end{bmatrix}}^{|Q_{S4}\rangle}}^{\text{Quantum State}} \tag{115}$$

Example 2: In this example, the initial state of the qubit is $\frac{15}{113}|0\rangle + \frac{112}{113}|1\rangle$. Therefore, $q_{In0} = \frac{15}{113}$ and $q_{In1} = \frac{112}{113}$. Additionally, the state of the initialized quabit satisfies $q^2_{In0} + q^2_{In1} = 1$. As same as previous example, here, $\kappa_1 = \frac{8\pi}{17}$, $\kappa_2 = \frac{3\pi}{7}$, $\kappa_3 = \frac{\pi}{12}$, and $\kappa_4 = \frac{3\pi}{10}$. Then, by applying the corresponded rotation gate for each initialized qubit, the new state of that qubit containing the encoded classical data can be obtained as follows:

$$|Q_{S1}\rangle = \begin{bmatrix}q_{S01}\\q_{S11}\end{bmatrix} = \begin{bmatrix}cos(\frac{8\pi}{34}) & -sin(\frac{8\pi}{34})\\ sin(\frac{8\pi}{34}) & cos(\frac{8\pi}{34})\end{bmatrix} \times \begin{bmatrix}\frac{15}{113}\\ \frac{112}{113}\end{bmatrix}, \tag{116}$$

$$|Q_{S2}\rangle = \begin{bmatrix}q_{S02}\\q_{S12}\end{bmatrix} = \begin{bmatrix}cos(\frac{3\pi}{14}) & -sin(\frac{3\pi}{14})\\ sin(\frac{3\pi}{14}) & cos(\frac{3\pi}{14})\end{bmatrix} \times \begin{bmatrix}\frac{15}{113}\\ \frac{112}{113}\end{bmatrix}, \tag{117}$$

$$|Q_{S3}\rangle = \begin{bmatrix}q_{S03}\\q_{S13}\end{bmatrix} = \begin{bmatrix}cos(\frac{\pi}{24}) & -sin(\frac{\pi}{24})\\ sin(\frac{\pi}{24}) & cos(\frac{\pi}{24})\end{bmatrix} \times \begin{bmatrix}\frac{15}{113}\\ \frac{112}{113}\end{bmatrix}, \tag{118}$$

and

$$|Q_{S4}\rangle = \begin{bmatrix}q_{S04}\\q_{S14}\end{bmatrix} = \begin{bmatrix}cos(\frac{3\pi}{20}) & -sin(\frac{3\pi}{20})\\ sin(\frac{3\pi}{20}) & cos(\frac{3\pi}{20})\end{bmatrix} \times \begin{bmatrix}\frac{15}{113}\\ \frac{112}{113}\end{bmatrix}, \tag{119}$$

Therefore,

$$\begin{aligned}|Q_{S1}\rangle &= (\frac{15}{113} \times cos(\frac{8\pi}{34}) - \frac{112}{113} \times sin(\frac{8\pi}{34}))|0\rangle\\ &... + (\frac{15}{113} \times sin(\frac{8\pi}{34}) + \frac{112}{113} \times cos(\frac{8\pi}{34}))|1\rangle\\ &\approx \begin{bmatrix}-0.5696\\0.8219\end{bmatrix},\end{aligned} \tag{120}$$

$$\begin{aligned}|Q_{S2}\rangle &= (\frac{15}{113} \times cos(\frac{3\pi}{14}) - \frac{112}{113} \times sin(\frac{3\pi}{14}))|0\rangle\\ &... + (\frac{15}{113} \times sin(\frac{3\pi}{14}) + \frac{112}{113} \times cos(\frac{3\pi}{14}))|1\rangle\\ &\approx \begin{bmatrix}-0.5142\\0.8577\end{bmatrix},\end{aligned} \tag{121}$$

$$\begin{aligned}|Q_{S3}\rangle &= (\frac{15}{113} \times cos(\frac{\pi}{24}) - \frac{112}{113} \times sin(\frac{\pi}{24}))|0\rangle\\ &... + (\frac{15}{113} \times sin(\frac{\pi}{24}) + \frac{112}{113} \times cos(\frac{\pi}{24}))|1\rangle\\ &\approx \begin{bmatrix}0.0022\\0.9999\end{bmatrix},\end{aligned} \tag{122}$$

and,

$$\begin{aligned}|Q_{S4}\rangle &= (\frac{15}{113} \times cos(\frac{3\pi}{20}) - \frac{112}{113} \times sin(\frac{3\pi}{20}))|0\rangle\\ &... + (\frac{15}{113} \times sin(\frac{3\pi}{20}) + \frac{112}{113} \times cos(\frac{3\pi}{20}))|1\rangle\\ &\approx \begin{bmatrix}-0.3317\\0.9434\end{bmatrix},\end{aligned} \tag{123}$$

So, in this example, $K = \begin{bmatrix}\kappa_1\\ \kappa_2\\ \kappa_3\\ \kappa_4\end{bmatrix}$ is mapped into a quantum state as follows:

$$\overbrace{\begin{bmatrix} \kappa_1 = \frac{8\pi}{17} \\ \kappa_2 = \frac{3\pi}{7} \\ \kappa_3 = \frac{\pi}{12} \\ \kappa_4 = \frac{3\pi}{10} \end{bmatrix}}^{\text{Classical Dataset}} \rightarrow \overbrace{|Q_{S1}\rangle \otimes |Q_{S2}\rangle \otimes |Q_{S3}\rangle \otimes |Q_{S4}\rangle}^{\text{Quantum State}} = \ldots$$
$$= \overbrace{\overbrace{\begin{bmatrix} -0.5696 \\ 0.8219 \end{bmatrix}}^{|Q_{S1}\rangle} \otimes \overbrace{\begin{bmatrix} -0.5142 \\ 0.8577 \end{bmatrix}}^{|Q_{S2}\rangle} \otimes \overbrace{\begin{bmatrix} 0.0022 \\ 0.9999 \end{bmatrix}}^{|Q_{S3}\rangle} \otimes \overbrace{\begin{bmatrix} -0.3317 \\ 0.9434 \end{bmatrix}}^{|Q_{S4}\rangle}}^{\text{Quantum State}} \tag{124}$$

Data pre-proccessing

In this part, details regarding two data pre-processing techniques will be talked mathematically. To achieve this goal, normalization as well as filling missed data and outliers will be discussed.

For the mapping data into the range of $[\varrho_{min}, \varrho_{max}]$, the following transformation as min-max normalization can be used[54]:

$$\lambda_{Nr} = \frac{(\lambda - MIN_\lambda) \times (\varrho_{max} - \varrho_{min})}{MAX_\lambda - MIN_\lambda} + \varrho_{min}. \tag{125}$$

Where, λ and λ_{Nr} are the desired data and the normalized vale of that, respectively. In addition, MAX_λ and MAX_λ are the maximum and the minimum value of the λ, respectively. So, based on (125), for normalizing P_L into $[0, 1]$ and $[-1, 1]$, (126) and (127) can be implemented,

$$P_{Nr,(0,1)}(j) = \frac{P_L(j) - MIN_P}{MAX_P - MIN_P}, \tag{126}$$

and,

$$P_{Nr,(-1,1)}(j) = \frac{2P_L(j) - 2MIN_P}{MAX_P - MIN_P} - 1. \tag{127}$$

Where, $P_{Nr,(0,1)}(j)$ and $P_{Nr,(-1,1)}(j)$ are the normalized value of P_L at $t = j$ into range of $[0, 1]$ and $[-1, 1]$, respectively.

Besides, to fill missed data, different strategies can be implemented. One of the common methods to fill the missed data, is based on a linear interpolation strategy as follows[55]:

$$\frac{\Xi(t) - \Xi(t_1)}{\Xi(t_2) - \Xi(t_1)} = \frac{t - t_1}{t_2 - t_1}. \tag{128}$$

Where, by substituting Ξ, t, t_1, and t_2 with P_L, m, j, and $j + k$, (128) can be updated as follows:

$$\frac{P_L(m) - P_L(j)}{P_L(j + k) - P_L(j)} = \frac{m - j}{(j + k) - j}, \tag{129}$$

In other words:

$$P_L(m) = \frac{(P_L(j + k) - P_L(j)) \times (m - j)}{k} + P_L(j), \tag{130}$$

Results

In this part, a dataset related to a real experimental setup is implemented to test the effectiveness of the proposed strategy. The dataset includes data of a refrigerator and a work station. Therefore, in this part, two scenarios are shown for the case of the refrigerator and the work station. In addition, before to implement the dataset, a data pre-proccessing strategy is used including actions such as data cleaning, filling missed data, removing and filling outliers, and normalization. For the case of normalization, min-max normalization technique is implemented, and for the case of filling missed data and replacing outliers, a linear interpolation approach is used. It is important to note that, in this study, Matlab programming language is implemented to simulate the quantum-based strategy.

For this part, N_{in} is 5. So, Q/C-ANN includes 5 inputs. Also, the sampling time Δt is $2s$. In addition, Part 1 of the hybrid network has a classical layer with 10 neurons. Also, Part 2 is made using 10 qubits and 10 rotation gates. Besides, Part 3 includes two classical layer. The first classical layer of Part 3 contains 3 neurons and the last classical layer of Part 3 includes one neuron, which produces the output of the hybrid network. Also, for both case studies, the size of the dataset is about 86400, and 70% of the samples are implemented to train the hybrid network.

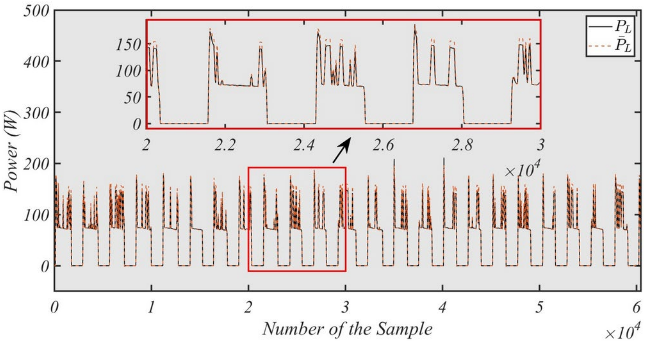


Fig. 6. Case Study 1 (Training Phase): The value of the power consumption of the refrigerator (P_L) and the predicted value of that ($\bar{P}_L$) for the training phase.

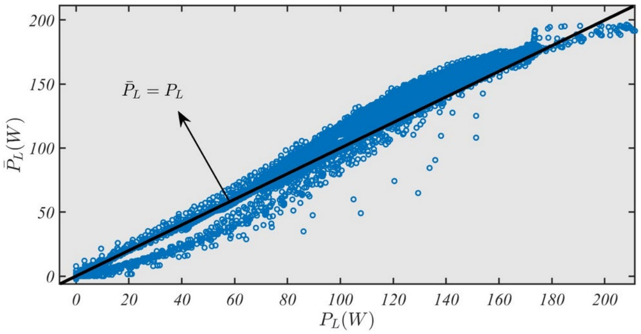


Fig. 7. Case Study 1 (Training Phase): The regression plot for the power consumption of the refrigerator (P_L) and the predicted value of that ($\bar{P}_L$) rlated to the training phase.

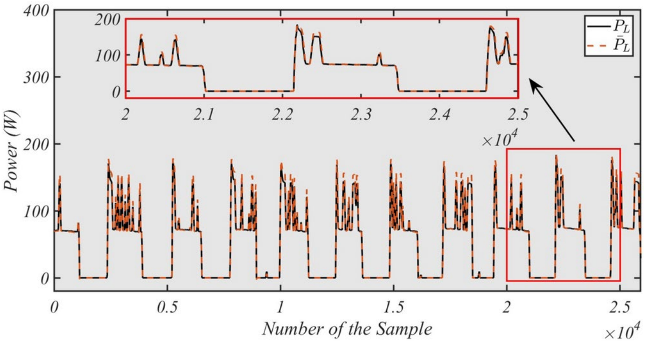


Fig. 8. Case Study 1 (Testing Phase): The value of the power consumption of the refrigerator (P_L) and the predicted value of that ($\bar{P}_L$) for the testing phase.

Case study 1: a refrigerator

In this scenario, a refrigerator is considered as the tested load. The goal of this scenario is to test the proposed strategy to predict the power consumption of a refrigerator. In Fig. 6, the value of the power consumption of the refrigerator (P_L) and the predicted value of that ($\bar{P}_L$) for the training phase is depicted. Based on Fig. 6, the hybrid network is trained properly. Further, the regression plot related to the training is shown by Fig. 7. In addition, Fig. 8 is related to the examination of the trained hybrid network and Fig. 9 is related to the regression plot for examining the trained hybrid network. Based on Fig. 8 and Fig. 9, the trained hybrid network works effectively.

Case study 2: a work station

Here, the goal of this case study is to predict the power consumption of a work station. The value of the power consumption and the anticipated value of that related to the training phase is depicted by Fig. 10. Based on the obtained result from Fig. 10, the hybrid network is trained successfully. Besides, Fig. 11 depicts the regression plot

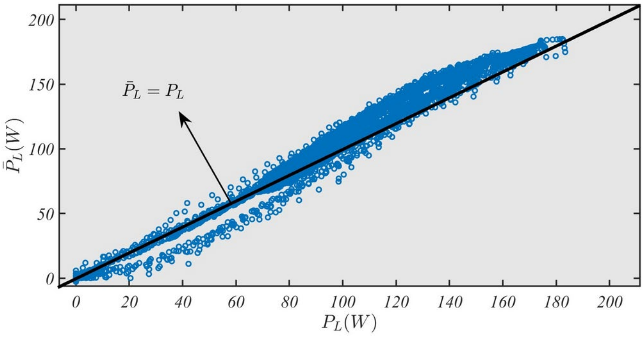


Fig. 9. Case Study 1 (Testing Phase): The regression plot for the power consumption of the refrigerator (P_L) and the predicted value of that ($\bar{P}_L$) related to the testing phase.

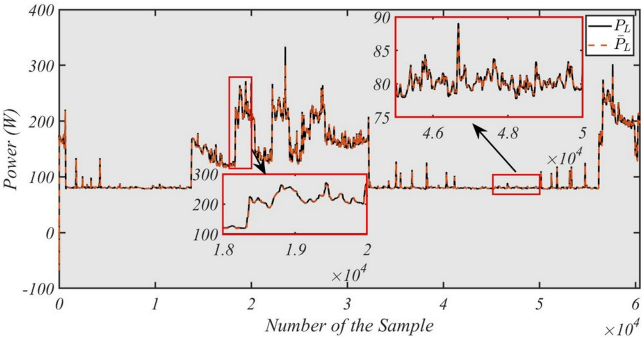


Fig. 10. Case Study 2 (Training Phase): The value of the power consumption of the work station (P_L) and the predicted value of that ($\bar{P}_L$) for the training phase.

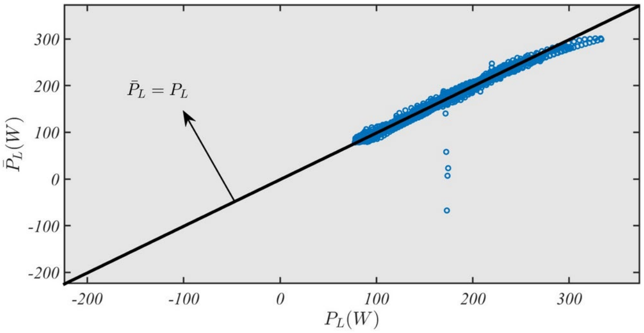


Fig. 11. Case Study 2 (Training Phase): The regression plot for the power consumption of the work station (P_L) and the predicted value of that ($\bar{P}_L$) related to the training phase.

of the training phase. Furthermore, Fig. 12 shows the value of the power consumption and the predicted value of that and for the test dataset. Besides, in Fig. 13, the regression plot regarding to the values of the parameters of Fig. 12 is plotted. Based on Fig. 12 and Fig. 13, the trained hybrid network can properly predict the power consumption of the work station.

Other AI-based approaches: applications of machine learning

In this part, to have a more in-depth view regarding the ability of AI to forecast the desired values and parameters, three classical models for the prediction of loads will be discussed. The models are linear regression, support vector machine (SVM) regression, and decision tree regression. The mentioned models are three well-known models of machine learning, which have been used widely to predict loads. It is important to note that, for all the case studies of this part, the same dataset related to Section VI is implemented for the refrigerator and the work station. In addition, the inputs and the output of the machine learning-based models are as same as the inputs and the output, which have been discussed in Section IV.

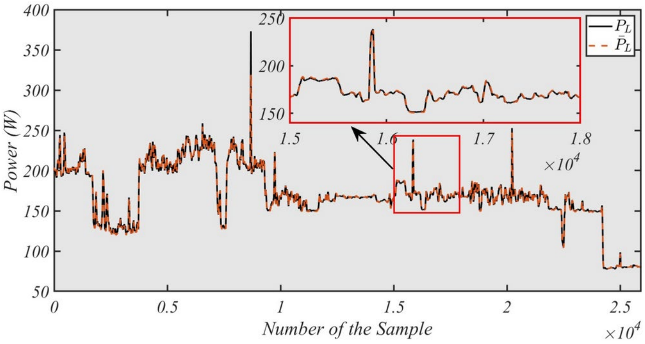


Fig. 12. Case Study 2 (Testing Phase): The value of the power consumption of the work station (P_L) and the predicted value of that ($\bar{P}_L$) for the testing phases.

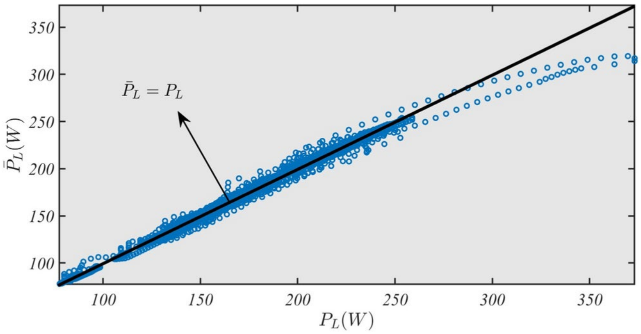
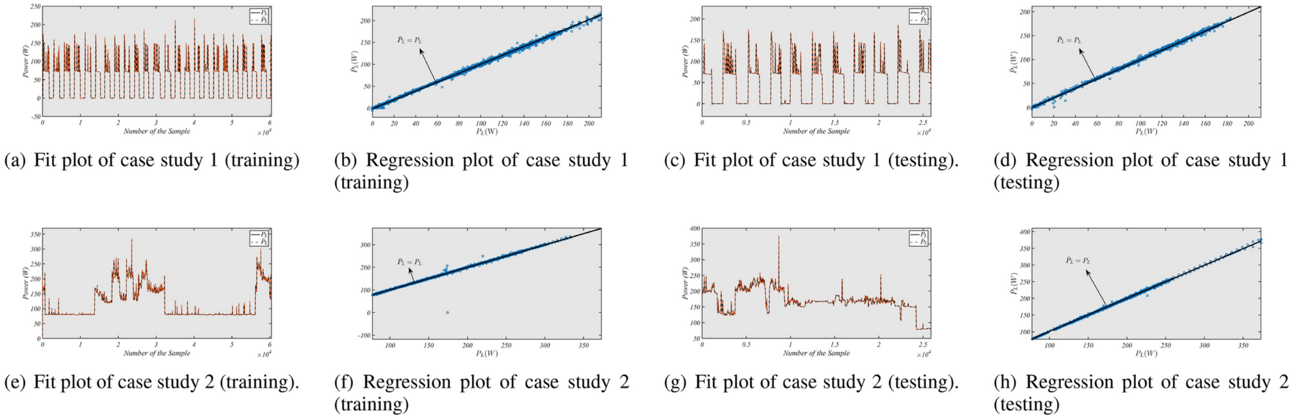


Fig. 13. Case Study 2 (Testing Phase): The regression plot for the power consumption of the work station (P_L) and the predicted value of that ($\bar{P}_L$) related to the testing phase.



(a) Fit plot of case study 1 (training)

(b) Regression plot of case study 1 (training)

(c) Fit plot of case study 1 (testing).

(d) Regression plot of case study 1 (testing)

(e) Fit plot of case study 2 (training).

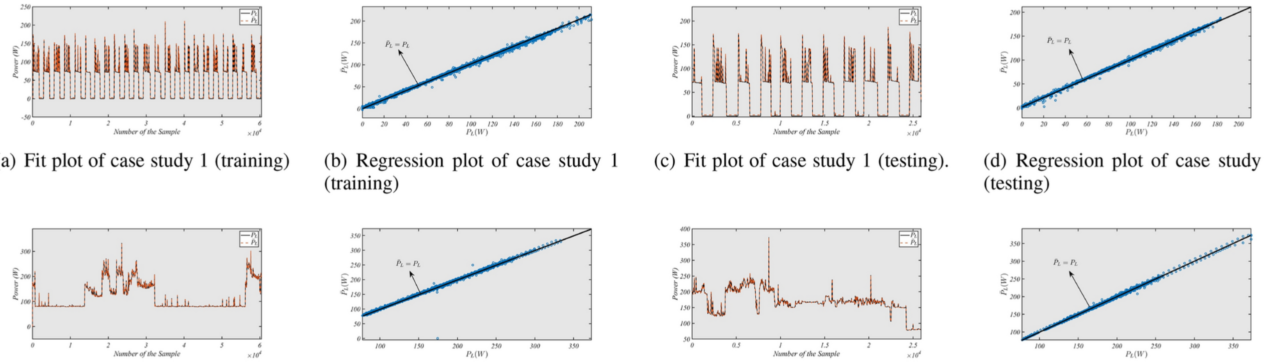
(f) Regression plot of case study 2 (training)

(g) Fit plot of case study 2 (testing).

(h) Regression plot of case study 2 (testing)

Fig. 14. The results of the linear regression related to forecasting the loads.

Linear regression-based strategies have been used before for load forecasting, e.g.,[56] and[57]. Figure. 14 shows the results related to the linear regression-based load forecasting. In Fig. 14(a), 14(b), 14(c), and 14(d), the training fit plot, training regression plot, testing fit plot, and testing regression plot of case study 1 are depicted, respectively. Furthermore, Fig. 14(e), 14(f), 14(g), and 14(h) show the training fit plot, training regression plot, testing fit plot, and testing regression plot of case study 2, respectively. Furthermore, previously, some studies have been done to predict the loads using support vector machine regression-based approaches such as[58–63]. The results related to support vector regression-based method is depicted in Fig. 15. For more clarification, Fig. 15(a), 15(b), 15(c), and 15(d) illustrate the results of case study 1, i.e., the fit and the regression plots. Further, Fig. 15(e), 15(f), 15(g), and 15(h) are related to case study 2. In addition, different decision tree-based strategies have been deployed to forecast the loads like[64] and[65]. Here, Fig. 16 shows the results of the decision tree regression for case study 1 and case study 2.



(a) Fit plot of case study 1 (training)

(b) Regression plot of case study 1 (training)

(c) Fit plot of case study 1 (testing).

(d) Regression plot of case study 1 (testing)

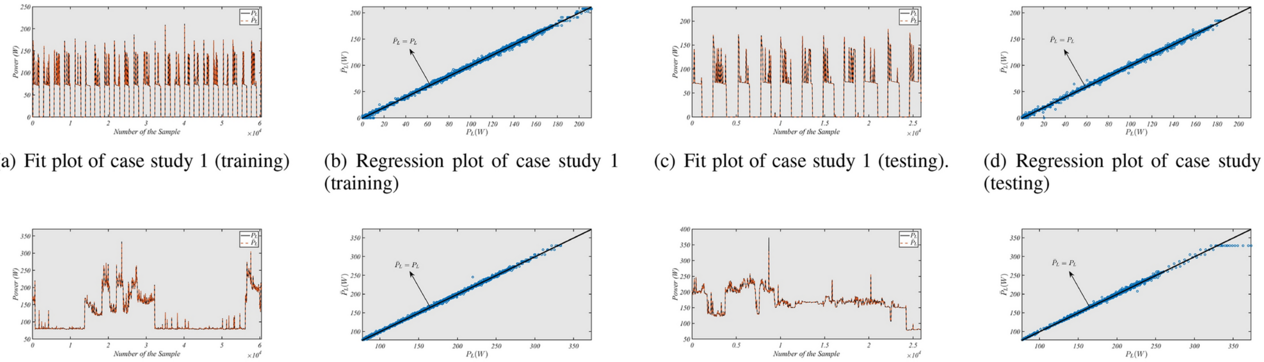
(e) Fit plot of case study 2 (training).

(f) Regression plot of case study 2 (training)

(g) Fit plot of case study 2 (testing).

(h) Regression plot of case study 2 (testing)

Fig. 15. The results of the support vector regression related to forecasting the loads.



(a) Fit plot of case study 1 (training)

(b) Regression plot of case study 1 (training)

(c) Fit plot of case study 1 (testing).

(d) Regression plot of case study 1 (testing)

(e) Fit plot of case study 2 (training).

(f) Regression plot of case study 2 (training)

(g) Fit plot of case study 2 (testing).

(h) Regression plot of case study 2 (testing)

Fig. 16. The results of the decision tree regression related to forecasting the loads.

Discussion

Briefly, mathematical-based modeling of a system considering all details of the system can be a challenging task. In other words, a system can involve some hidden or undiscovered details and as a result, it is not easy and practical to extract all information from the system using the existing mathematical-based model of the system. On the other hand, an alternative solution to achieve the information of a system as much as possible without the deployment of mathematical-based modeling of the studied system is access to data employing measurements if possible. A Q/C-ANN can work only with data that do not require the mathematical-based model of the system. Therefore, a Q/C-ANN can be considered as a proper data-driven-based computational tool to extract the information of the system (e.g., the value of the electrical loads).

In addition, a Q/C-ANN includes quantum gates and as a result, a quantum circuit using those gates, that can provide the opportunity to exploit the advantages of quantum computing, e.g., making an entanglement between the deployed qubits to have a secure data transmission strategy to transfer the information of the system between different units in a power system. Furthermore, typically to perform a quantum algorithm three main parts can be required, i.e., Part1 for coding classical data to create the state of the input qubits, Part 2 to use the coded qubits in a quantum circuit that is made using quantum gates, and Part 3 to encode the measurements of the states of the output qubits to classical data. A Q/C-ANN has the main required three parts naturally. For more clarification, Part 1 and Part 3 of a Q/C-ANN are made by classical hidden layers and they are cooperating for coding classical data to modify the state of the initialized qubits and decoding the measured state of the output qubits to classical data, respectively. In addition, Part 2 of a Q/C-ANN is made using quantum gates to structure a quantum circuit.

Besides, challenges regarding the deployment of a fully quantum-based strategy are the limitation of access to a large number of qubits, and the sensitivity of a fully quantum-based methodology to the noises. Based on the mentioned challenges, the implementation of quantum-based approaches for a large-scale system (e.g., a power system) can be a difficult task. A solution to overcome these kinds of difficulties is the implementation of Q/C-ANNs, which can require less number of quantum gates due to the implementation of classical parts.

Therefore, based on the above-mentioned advantages, a Q/C-ANN can offer an effective solution for forecasting-based applications in power systems such as the prediction of the values of the loads.

Conclusion

Q/C-ANNs have quantum and classical parts. The quantum part includes a quantum circuit containing quantum operators. Also, classical parts are made by classical ANNs and using classical neurons. A Q/C-ANN can be implemented to solve different challenges. In this work, application of Q/C-ANNs has been implemented for load forecasting in smart grids. This study implemented the hybrid network in a form of time-series, which requires only the historical as well as the current values of the load to predict the future value of that. In addition, in this study, two different types of loads (i.e., refrigerator, and work station) have been predicted in a short-term mode. The implemented strategy was able to predict the loads effectively.

Future works

For the future direction, the hybrid network can be used to solve other problems, e.g., load classification, fault detection, etc. In addition, quantum annealing has been studied before by some studies such as[66–68], and the implementation of that as well as the comparison of that with various methods can be considered for further investigation to predict the values of the loads.

Data availibility

Data are available from the authors (M.R.H. Can be contacted) upon reasonable request (in the case of permission of IOT MICROGRID LIVING LABORATORY (IOT-MGLAB) at AAU Energy).

Received: 21 August 2023; Accepted: 10 February 2025
Published online: 03 March 2025

References

1. Akpolat, A. N. et al. Dynamic stabilization of dc microgrids using ann-based model predictive control. *IEEE Transactions on Energy Conversion* **37**(2), 999–1010 (2022).
2. Akpolat, A. N. et al. Sensorless control of dc microgrid based on artificial intelligence. *IEEE Transactions on Energy Conversion* **36**(3), 2319–2329 (2021).
3. Z. Li, Z. Jiao, and A. He, "Dynamic differential current-based transformer protection using convolutional neural network," *CSEE Journal of Power and Energy Systems*, pp. 1–15, 2022.
4. Kim, W.-H., Kim, J.-Y., Chae, W.-K., Kim, G. & Lee, C.-K. Lstm-based fault direction estimation and protection coordination for networked distribution system. *IEEE Access* **10**, 40348–40357 (2022).
5. Murugan, S. K., Simon, S. P. & Eapen, R. R. A novel signal localized convolution neural network for power transformer differential protection. *IEEE Transactions on Power Delivery* **37**(2), 1242–1251 (2022).
6. Habibi, M. R., Baghaee, H. R., Dragičević, T. & Blaabjerg, F. False data injection cyber-attacks mitigation in parallel dc/dc converters based on artificial neural networks. *IEEE Transactions on Circuits and Systems II: Express Briefs* **68**(2), 717–721 (2021).
7. M. R. Habibi, T. Dragicevic, and F. Blaabjerg, "Secure control of dc microgrids under cyber-attacks based on recurrent neural networks," in *2020 IEEE 11th International Symposium on Power Electronics for Distributed Generation Systems (PEDG)*, pp. 517–521, 2020.
8. Diaba, S. Y., Shafie-Khah, M. & Elmusrati, M. Cyber security in power systems using meta-heuristic and deep learning algorithms. *IEEE Access* **11**, 18660–18672 (2023).
9. Habibi, M. R., Baghaee, H. R., Dragičević, T. & Blaabjerg, F. Detection of false data injection cyber-attacks in dc microgrids based on recurrent neural networks. *IEEE Journal of Emerging and Selected Topics in Power Electronics* **9**(5), 5294–5310 (2021).
10. Wang, Y., Zhang, Z., Ma, J. & Jin, Q. Kfrnn: An effective false data injection attack detection in smart grid based on kalman filter and recurrent neural network. *IEEE Internet of Things Journal* **9**(9), 6893–6904 (2022).
11. Habibi, M. R., Sahoo, S., Rivera, S., Dragičević, T. & Blaabjerg, F. Decentralized coordinated cyberattack detection and mitigation strategy in dc microgrids based on artificial neural networks. *IEEE Journal of Emerging and Selected Topics in Power Electronics* **9**(4), 4629–4638 (2021).
12. Habibi, M. R., Baghaee, H. R., Blaabjerg, F. & Dragičević, T. Secure control of dc microgrids for instant detection and mitigation of cyber-attacks based on artificial intelligence. *IEEE Systems Journal* **16**(2), 2580–2591 (2022).
13. Habibi, M. R., Baghaee, H. R., Blaabjerg, F. & Dragičević, T. Secure mpc/ann-based false data injection cyber-attack detection and mitigation in dc microgrids. *IEEE Systems Journal* **16**(1), 1487–1498 (2022).
14. Adineh, B., Habibi, M. R., Akpolat, A. N. & Blaabjerg, F. Sensorless voltage estimation for total harmonic distortion calculation using artificial neural networks in microgrids. *IEEE Transactions on Circuits and Systems II: Express Briefs* **68**(7), 2583–2587 (2021).
15. Wu, Z. & Wang, B. An ensemble neural network based on variational mode decomposition and an improved sparrow search algorithm for wind and solar power forecasting. *IEEE Access* **9**, 166709–166719 (2021).
16. Habibi, M. R., Guerrero, J. M. & Vasquez, J. C. Artificial intelligence for cybersecurity monitoring of cyber-physical power electronic converters: a dc/dc power converter case study. *Scientific Reports* **14**(1), 22072 (2024).
17. Habibi, M. R., Golestan, S., Guerrero, J. M. & Vasquez, J. C. Deep learning for forecasting-based applications in cyber-physical microgrids: Recent advances and future directions. *Electronics* **12**(7), 1685 (2023).
18. Zhou, Y., Ding, Z., Wen, Q. & Wang, Y. Robust load forecasting towards adversarial attacks via bayesian learning. *IEEE Transactions on Power Systems* **38**(2), 1445–1459 (2023).
19. Parizad, A. & Hatziadoniu, C. Deep learning algorithms and parallel distributed computing techniques for high-resolution load forecasting applying hyperparameter optimization. *IEEE Systems Journal* **16**(3), 3758–3769 (2022).
20. Kim, N., Park, H., Lee, J. & Choi, J. K. Short-term electrical load forecasting with multidimensional feature extraction. *IEEE Transactions on Smart Grid* **13**(4), 2999–3013 (2022).
21. Hou, H. et al. Load forecasting combining phase space reconstruction and stacking ensemble learning. *IEEE Transactions on Industry Applications* **59**(2), 2296–2304 (2023).
22. Zhang, Z., Zhao, P., Wang, P. & Lee, W.-J. Transfer learning featured short-term combining forecasting model for residential loads with small sample sets. *IEEE Transactions on Industry Applications* **58**(4), 4279–4288 (2022).
23. Ijaz, K. et al. A novel temporal feature selection based lstm model for electrical short-term load forecasting. *IEEE Access* **10**, 82596–82613 (2022).
24. Li, C. et al. Interpretable memristive lstm network design for probabilistic residential load forecasting. *IEEE Transactions on Circuits and Systems I: Regular Papers* **69**(6), 2297–2310 (2022).
25. Briggs, C., Fan, Z. & Andras, P. Federated learning for short-term residential load forecasting. *IEEE Open Access Journal of Power and Energy* **9**, 573–583 (2022).

26. Liu, M., Qin, H., Cao, R. & Deng, S. Short-term load forecasting based on improved tcn and densenet. *IEEE Access* **10**, 115945–115957 (2022).
27. S. K. Sood and Pooja, "Quantum computing review: A decade of research," *IEEE Transactions on Engineering Management*, vol. 71, pp. 6662–6676, 2024.
28. M. R. Habibi, S. Golestan, A. Soltanmanesh, J. M. Guerrero, and J. C. Vasquez, "Power and energy applications based on quantum computing: The possible potentials of grover's algorithm," *Electronics*, vol. 11, no. 18, 2022.
29. Koudia, S., Cacciapuoti, A. S. & Caleffi, M. Deterministic generation of multipartite entanglement via causal activation in the quantum internet. *IEEE Access* **11**, 73863–73878 (2023).
30. Li, J., Jia, Q., Xue, K., Wei, D. S. L. & Yu, N. A connection-oriented entanglement distribution design in quantum networks. *IEEE Transactions on Quantum Engineering* **3**, 1–13 (2022).
31. Tang, Z., Zhang, P. & Krawec, W. O. A quantum leap in microgrids security: The prospects of quantum-secure microgrids. *IEEE Electrification Magazine* **9**(1), 66–73 (2021).
32. Tang, Z., Qin, Y., Jiang, Z., Krawec, W. O. & Zhang, P. Quantum-secure microgrid. *IEEE Transactions on Power Systems* **36**(2), 1250–1263 (2021).
33. Tang, Z., Zhang, P. & Krawec, W. O. Enabling resilient quantum-secured microgrids through software-defined networking. *IEEE Transactions on Quantum Engineering* **3**, 1–11 (2022).
34. Yan, R., Wang, Y., Dai, J., Xu, Y. & Liu, A. Q. Quantum-key-distribution-based microgrid control for cybersecurity enhancement. *IEEE Transactions on Industry Applications* **58**(3), 3076–3086 (2022).
35. Nikmehr, N., Zhang, P. & Bragin, M. A. Quantum distributed unit commitment: An application in microgrids. *IEEE Transactions on Power Systems* **37**(5), 3592–3603 (2022).
36. Feng, F., Zhou, Y. & Zhang, P. Quantum power flow. *IEEE Transactions on Power Systems* **36**(4), 3810–3812 (2021).
37. Zhou, Y., Feng, F. & Zhang, P. Quantum electromagnetic transients program. *IEEE Transactions on Power Systems* **36**(4), 3813–3816 (2021).
38. Zhou, Y., Zhang, P. & Feng, F. Noisy-intermediate-scale quantum electromagnetic transients program. *IEEE Transactions on Power Systems* **38**(2), 1558–1571 (2023).
39. Zhou, Y. & Zhang, P. Noise-resilient quantum machine learning for stability assessment of power systems. *IEEE Transactions on Power Systems* **38**(1), 475–487 (2023).
40. C.-W. Lee and B.-Y. Lin, "Application of hybrid quantum tabu search with support vector regression (svr) for load forecasting," *Energies*, vol. 9, no. 11, 2016.
41. C.-W. Lee and B.-Y. Lin, "Applications of the chaotic quantum genetic algorithm with support vector regression in load forecasting," *Energies*, vol. 10, no. 11, 2017.
42. Eskandarpour, R. et al. Quantum computing for enhancing grid security. *IEEE Transactions on Power Systems* **35**(5), 4135–4137 (2020).
43. Morstyn, T. Annealing-based quantum computing for combinatorial optimal power flow. *IEEE Transactions on Smart Grid* **14**(2), 1093–1102 (2023).
44. S. Golestan, M. Habibi, S. Mousazadeh Mousavi, J. Guerrero, and J. Vasquez, "Quantum computation in power systems: An overview of recent advances," *Energy Reports*, vol. 9, pp. 584–596, 2023.
45. A. Safari and A. A. Ghavifekr, "Quantum technology & quantum neural networks in smart grids control: Premier perspectives," in *2022 8th International Conference on Control, Instrumentation and Automation (ICCIA)*, pp. 1–6, 2022.
46. Nielsen, M. A. & Chuang, I. *Quantum computation and quantum information: 10th anniversary edition* (Cambridge Univ. Press, Cambridge, U.K., 2011).
47. D. McMahon, *Quantum computing explained*. John Wiley & Sons, 2008.
48. S. Axler, *Linear algebra done right*. Springer Nature, 2024.
49. Rath, M. & Date, H. Quantum data encoding: a comparative analysis of classical-to-quantum mapping techniques and their impact on machine learning accuracy. *EPJ Quantum Technology* **11**(1), 72 (2024).
50. M. Weigold, J. Barzen, F. Leymann, and M. Salm, "Expanding data encoding patterns for quantum algorithms," in *2021 IEEE 18th International Conference on Software Architecture Companion (ICSA-C)*, pp. 95–101, 2021.
51. Khan, M. A., Aman, M. N. & Sikdar, B. Beyond bits: A review of quantum embedding techniques for efficient information processing. *IEEE Access* **12**, 46118–46137 (2024).
52. G. Agliardi and E. PRATI, "Quantum data encoding as a distinct abstraction layer in the design of quantum circuits," *Quantum Science and Technology*, 2025.
53. N. Munikote, "Comparing quantum encoding techniques," arXiv preprint arXiv:2410.09121, 2024.
54. S. García, J. Luengo, and F. Herrera, *Data preprocessing in data mining*, vol. 72. Springer, 2015.
55. Q. Miao, Y. Zhang, S. Xu, J. Yang, Y. Yang, and G. Wei, "Research on filling method of tidal missing data based on gated recurrent neural network," in *2022 4th International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI)*, pp. 324–329, 2022.
56. A. Y. Saber and A. K. M. R. Alam, "Short term load forecasting using multiple linear regression for big data," in *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–6, 2017.
57. Dudek, G. Pattern-based local linear regression models for short-term load forecasting. *Electric Power Systems Research* **130**, 139–147 (2016).
58. Y. Cai, Q. Xie, C. Wang, and F. Lü, "Short-term load forecasting for city holidays based on genetic support vector machines," in *2011 International Conference on Electrical and Control Engineering*, pp. 3144–3147, 2011.
59. C. Lei, B. Fang, H. Gao, W. Jia, and W. Pan, "Short-term power load forecasting based on least squares support vector machine optimized by bare bones fireworks algorithm," in *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pp. 2231–2235, 2018.
60. Ye, N., Liu, Y. & Wang, Y. Short-term power load forecasting based on svm. *World Automation Congress* **2012**, 47–51 (2012).
61. J. Masood, S. Javaid, S. Ahmed, S. Ullah, and N. Javaid, "An optimized linear-kernel support vector machine for electricity load and price forecasting in smart grids," in *2019 International Conference on Advances in the Emerging Computing Technologies (AECT)*, pp. 1–6, 2020.
62. Li, G., Li, Y. & Roozitalab, F. Midterm load forecasting: A multistep approach based on phase space reconstruction and support vector machine. *IEEE Systems Journal* **14**(4), 4967–4977 (2020).
63. Ceperic, E., Ceperic, V. & Baric, A. A strategy for short-term load forecasting by support vector regression machines. *IEEE Transactions on Power Systems* **28**(4), 4356–4364 (2013).
64. D. Chowdhury, M. Sarkar, M. Z. Haider, and T. Alam, "Zone wise hourly load prediction using regression decision tree model," in *2018 International Conference on Innovation in Engineering and Technology (ICIET)*, pp. 1–6, 2018.
65. X. Yu, Z. Xu, X. Zhou, J. Zheng, Y. Xia, L. Lin, and S.-H. Fang, "Load forecasting based on smart meter data and gradient boosting decision tree," in *2019 Chinese Automation Congress (CAC)*, pp. 4438–4442, 2019.
66. M. Takahashi, H. Nishioka, M. Hirai, and H. Takano, "A study of the optimization problem on the combination of sectionalizing switches in power grid with quantum annealing," 2023.
67. Paterakis, N. G. Hybrid quantum-classical multi-cut benders approach with a power system application. *Computers & Chemical Engineering* **172**, 108161 (2023).

68. Rajak, A., Suzuki, S., Dutta, A. & Chakrabarti, B. K. Quantum annealing: An overview. *Philosophical Transactions of the Royal Society A* **381**(2241), 20210417 (2023).

Acknowledgements

The work was supported by VILLUM FONDEN under the VILLUM Investigator GRANT, no. 25920: Center for Research on Microgrids (CROM), Denmark.

Author contributions

M. R. Habibi wrote the manuscript text, did data cleaning and pre-processing data, tested the quantum artificial intelligence strategy on data, and obtained the results. Also, Y. Wu gathered the initial and non-pre-process data. In addition, all other co-authors have done supervision.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to M.R.H.