Quantum neural networks for power flow analysis

Kaseb, Zeynab; Möller, Matthias; Balducci, Giorgio Tosti; Palensky, Peter; Vergara, Pedro P.

DOI
10.1016/j.epsr.2024.110677

Publication date
2024

Document Version
Final published version

Published in
Electric Power Systems Research

Electric Power Systems Research 235 (2024) 110677

Contents lists available at ScienceDirect

Electric Power Systems Research

journal homepage: www.elsevier.com/locate/epsr

Quantum neural networks for power flow analysis

Zeynab Kaseb [a,*], Matthias Möller [b], Giorgio Tosti Balducci [c], Peter Palensky [a], Pedro P. Vergara [a]

[a] Electrical Sustainable Energy, Delft University of Technology, Delft, The Netherlands
[b] Applied Mathematics, Delft University of Technology, Delft, The Netherlands
[c] Aerospace Structures and Materials, Delft University of Technology, Delft, The Netherlands

ARTICLE INFO

Keywords:
Distribution networks
Load flow calculation
Gate-based quantum computing model
Parameterized quantum circuit
Variational quantum algorithm

ABSTRACT

This paper explores the potential application of quantum and hybrid quantum–classical neural networks in power flow analysis. Experiments are conducted using two datasets based on 4-bus and 33-bus test systems. A systematic performance comparison is also conducted among quantum, hybrid quantum–classical, and classical neural networks. The comparison is based on (i) generalization ability, (ii) robustness, (iii) training dataset size needed, (iv) training error, and (v) training process stability. The results show that the developed hybrid quantum–classical neural network outperforms both quantum and classical neural networks, and hence can improve deep learning-based power flow analysis in the noisy-intermediate-scale quantum (NISQ) and fault-tolerant quantum (FTQ) era.

1. Introduction

Efficient and secure power system operation depends heavily on power flow (PF) analysis. This analysis is traditionally performed using iterative numerical methods, which pose computational challenges for large-scale modern power systems and suffer from inaccuracies in certain scenarios [1]. As a result, the development of scalable, reliable, and computationally tractable PF methods is of great importance to meet the evolving demands of modern power systems characterized by a large number of distributed energy resources, variable loads, and bidirectional power flows, among others [2,3].

Neural networks have been widely used for PF analysis due to their capacity to address complexities present in modern power systems [4]. By learning from historical data, neural networks reveal nonlinear and complex relationships between inputs and outputs even in cases for which traditional iterative numerical methods fail to converge, e.g. for ill-conditioned scenarios. This capability is especially critical in accommodating varying load demands and integrating distributed energy resources in large-scale modern power systems [5,6].

However, while neural networks offer significant advantages for PF analysis, their practical implementation presents certain challenges. Firstly, they rely on large, high-quality datasets, which can be hard to obtain due to various reasons, such as privacy concerns and high proportions of missing data. In addition, the computational complexity of the training process increases with the size and complexity of power systems. For instance, the increased depth of neural networks necessitates complicated hyperparameter tuning, which demands substantial computational resources. Finally, ensuring scalability, generalization ability, and robustness to noisy training data remains a critical concern for classical neural networks (NNs). They are often case-specific and prone to overfitting to training data, which can potentially compromise the reliability of classical NNs in safety-critical applications (e.g. [7,8]).

A radically different machine learning approach that can help overcome the challenges faced by classical NNs is quantum neural networks (QNNs), which means the enrichment of NNs with quantum computing. Quantum computing is increasingly gaining attention as it has the potential to address the complexities of modern power systems, see Table 1. The scope of the studies is grouped into two main quantum computing paradigms: (i) the gate-based quantum computing model (GQC) and (ii) the adiabatic quantum computing model (AQC). A majority of the studies, summarized in Table 1, focus on the GQC approach, which can simulate specific computations using quantum gates and discrete time steps, while a few studies implemented the AQC approach. AQC is polynomially equivalent to GQC, yet its nature is analog, which means that there are no quantum gates and no discrete time steps in this approach [9]. AQC is specifically suitable for (combinatorial) optimization applications.

Table 1 shows that the Harrow–Hassidim–Lloyd (HHL) algorithm has been widely used in different PF applications, including PF analysis and state estimation (SE) (e.g. [7,10]). The HHL algorithm theoretically offers up to exponential speedup in solving systems of linear equations compared to state-of-the-art classical solvers. However, it requires a

* Corresponding author.
E-mail addresses: Z.Kaseb@tudelft.nl (Z. Kaseb), M.Moller@tudelft.nl (M. Möller), G.B.L.TostiBalducci@tudelft.nl (G.T. Balducci), P.Palensky@tudelft.nl (P. Palensky), P.P.VergaraBarrios@tudelft.nl (P.P. Vergara).

https://doi.org/10.1016/j.epsr.2024.110677
Received 1 October 2023; Received in revised form 10 March 2024; Accepted 17 June 2024
Available online 3 July 2024

Table 1
Literature on the use of quantum computing for PF.

Paradigm[a]	Algorithm[b]	Application[c]	Test case size[d]	Hardware
GQC	HHL [17]	DCPF	3-bus	IBM
	VQC [18]	TSA	11-bus/NPCC	IBM
	HHL [7]	FDPF	5-bus	–
	HHL [19]	ACPF	3-bus/5-bus	IBM
	VQLS [12]	ACPF	5-bus	IBM
	HHL [10]	SE	2-bus/4-bus	–
	QAOA [20]	UC	9-DER	IBM
	QAOA [21]	MS	24-bus	–
	HHL [22]	DCOPF	3-bus/14-bus	–
AQC	VQC [23]	FD	30-bus	D-Wave
	QA [24]	UC	2-TU	D-Wave
	QA [13]	ACOPF	55-bas	D-Wave

[a] GQC = gate-based quantum computing model, AQC = adiabatic quantum computing model.
[b] HHL = Harrow–Hassidim–Lloyd algorithm, VQC = variational quantum circuits, VQLS = variational quantum linear solver, QAOA = quantum approximate optimization algorithm, QA = quantum annealing.
[c] DCPF = direct current power flow, TSA = transient stability assessment, FDPF = fast-decoupled power flow, ACPF = alternating current power flow, SE = state estimation, UC = unit commitment, MS = maximum sections of power delivery and data traffic, DCOPF = direct current optimal power flow, FD = fault detection, ACOPF = alternating current optimal power flow.
[d] NPCC = Northeast Power Coordinating Council test system [25], DER = distributed energy resource, TU = thermal unit.

large number of quantum gates, even for small-scale problems, and hence, its performance is adversely affected by the noise inherent in current quantum computers [11]. Moreover, it does not allow to extract the full solution vector $\vec{z}$ efficiently but only a scalar key performance indicator of the form $\vec{z}^T M \vec{z}$, where M is some sparse matrix. In [12], a variational quantum linear solver (VQLS) has been developed, which uses fewer quantum gates and, therefore, enables resilient PF analysis using noisy quantum computers. However, there is no theoretical justification (yet) that VQLS might offer any computational speedup. In addition to addressing the solution of linear systems of equations, several studies have focused on Quadratic Unconstrained Binary Optimization (QUBO) formulation for specific applications, such as Optimal Power Flow (OPF) [13]. Note that QUBO formulation can be solved by GQC and AQC approaches, as presented in Table 1.

This paper specifically looks at GQC and QNNs. QNNs are a topic of active research in the broader field of quantum machine learning (QML), which comprise layers of quantum gates. They are able to explore high-dimensional feature spaces with a limited number of quantum gates, which potentially leads to superior performance in practical applications, including PF analysis [14]. In addition, the inherent randomness of quantum phenomena allows for the capture of complex relationships between inputs and output with reduced reliance on large datasets [15]. These advantages not only enhance the training process but also address data scarcity concerns and make QNNs a potential candidate for learning from small, incomplete, and/or noisy datasets [16].

Several studies have recently explored the successful application of QNNs using quantum simulators in different fields and highlighted their growing potential. In facial recognition and analysis, for example, researchers developed a software system equipped with a camera and a QNN to efficiently differentiate various face patterns [26]. Another example is in financial predictions, where the high accuracy and efficiency of QNNs are proved in predicting financial time series [27]. Likewise, a novel QNN was proposed in precision oncology for drug response prediction. The study showed that the developed QNN outperformed classical methods in predicting drug effectiveness values [28].

Although significant success has been achieved in the use of QNNs in various fields, and different quantum algorithms have been developed for PF applications, QNNs have not yet been systematically explored for PF analysis. This paper serves as a proof of concept and represents the first endeavor to systematically investigate the use of QNNs for deep learning-based PF analysis. The main contributions of this paper are:

(1) A hybrid quantum–classical neural network (QCNN), and a pure quantum neural network (QNN) are developed for PF analysis. The proposed algorithms provide improved generalization ability and reduce the training dataset size needed compared to classical NNs while maintaining similar accuracy compared to iterative numerical methods, i.e. the Newton–Raphson method (NR).
(2) A thorough performance comparison is conducted between classical NNs, QCNNs, and QNNs for PF analysis. The comparison is based on (i) generalization ability, (ii) robustness, (iii) required training dataset size, (iv) training error, and (v) training process stability.

All quantum components are implemented in Qiskit (version 0.46.0) and executed using the Aer statevector simulator on a classical computing system, specifically Ubuntu 22.04, with 16 physical cores and 64 GB of RAM. The simulations are made more realistic by introducing statistical noise and hardware noise. The same computer was used to obtain the ground truth data with the Newton–Raphson method (NR). The focus is on a 4-bus test system [29]. Supplementary experiments are also performed on a 33-bus test system [30] to show the scalability and the potential benefits of the QCNN for PF analysis.

2. Power flow analysis

The aim of PF analysis is to calculate the voltage magnitude and phase angle for all buses within power systems. It can be performed based on the alternating current power flow (ACPF) equations, which are a set of nonlinear equations that relate the complex voltages and power at each bus of a power system:

$$p_i = \sum_{j=1}^{n} v_i v_j (g_{ij} \cos \delta_{ij} + b_{ij} \sin \delta_{ij}), \tag{1}$$

$$q_i = \sum_{j=1}^{n} v_i v_j (g_{ij} \sin \delta_{ij} - b_{ij} \cos \delta_{ij}), \tag{2}$$

where i and j are the indices of the buses, n is the total number of buses, v_i and δ_i are the magnitude and phase angle of the complex voltage at bus i, p_i and q_i are the active and reactive power injection/consumption at bus i, g_{ij} and b_{ij} are the real and imaginary parts of the admittance between buses i and j, and $\delta_{ij} = \delta_i - \delta_j$ is the phase angle difference between the voltages at buses i and j. The equations are traditionally solved using the Newton–Raphson method (NR) [31].

3. Classical neural networks for PF analysis

A feed-forward NN is employed to approximate $\vec{y} \in \{(\vec{v}_i, \vec{\delta}_i) : i = 1, 2, \dots, n\}$ as output labels given $\vec{x} \in \{(\vec{p}_i, \vec{q}_i) : i = 1, 2, \dots, n\}$ as input features, see Fig. 1. That is, the input and output layers consist of $n \times 2$ neurons each, where n is the number of buses. The architecture employs a chain of functions $f(\vec{x}) = l_k \circ \dots \circ l_1(\vec{x})$, sequentially processing $\vec{x}$ through k hidden layers to obtain $\vec{f}(\cdot) \in \{(\vec{\hat{v}}_i, \vec{\hat{\delta}}_i) : i = 1, 2, \dots, n\}$. Here, $l_k(\vec{x}) = \sigma(W_k^T \cdot \vec{x} + b_k)$ is the kth hidden layer. W_k^T and b_k are weight matrix and bias vector for the respective hidden layer. Each hidden layer applies a linear transformation, i.e. $W_k^T \cdot \vec{x} + b_k$, followed by a nonlinear transformation, i.e. $\sigma(\cdot)$, to capture complex relationships between $\vec{x}$ and $\vec{y}$. W_k and b_k are trainable parameters optimized during the training process to minimize the difference between $\vec{f}(\cdot)$, approximated by the NN, and ground-truth output labels $\vec{y}$, obtained from the NR, based on a loss function of choice, e.g. the mean squared error (MSE):

$$\mathscr{L} = \frac{1}{N} \sum_{j=1}^{N} (\vec{y}_j - f(\vec{x}_j))^2, \tag{3}$$

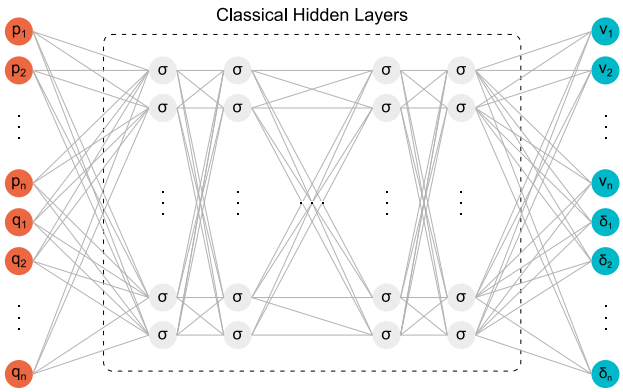

Fig. 1. Schematic depiction of the NN designed for PF analysis. $\vec{x} \in \{(\vec{p}_i, \vec{q}_i) : i = 1, 2, \dots, n\}$ and $\vec{y} \in \{(\vec{v}_i, \vec{\delta}_i) : i = 1, 2, \dots, n\}$ respectively represent the input features and output labels, where n is the number of buses.

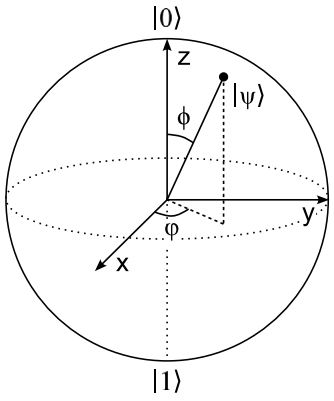

Fig. 2. Schematic representation of the quantum state, which can be in a superposition of states $|0\rangle$ and $|1\rangle$. ϕ and φ determine the probability amplitude of the state being $|0\rangle$ and $|1\rangle$, respectively. The qubit can be manipulated by applying quantum gates to change ϕ and φ.

where j is the index of the training data point, N is the total number of training data points, $\vec{x}_j$ and $\vec{y}_j$ are the vector of input features and output labels obtained from the NR for the jth training data point. $\vec{f}(.)$ is the vector of approximated output labels obtained from the NN.

4. Quantum neural networks for PF analysis

A qubit is defined by complex coefficients and is represented by two angles on the Bloch sphere, as shown in Fig. 2. Unlike a classical bit confined to the values 0 or 1, a qubit can exist in a superposition of both. This means that prior to measurement, a qubit can be in a state that is a linear combination of the states $|0\rangle$ and $|1\rangle$, which introduces a probabilistic nature, and hence, improves the generalization ability of QNNs but also their robustness against noisy datasets [16]. The state of a qubit $|\psi\rangle$ is mathematically expressed as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad |\alpha|^2 + |\beta|^2 = 1, \quad \alpha, \beta \in \mathbb{C}, \tag{4}$$

where $\alpha = \cos\frac{\phi}{2}$ and $\beta = \sin\frac{\phi}{2}$ are the probability amplitudes of the basic state $|0\rangle$ and $|1\rangle$, respectively. This can be easily extended to a quantum register $|\psi\rangle = \sum_{i=0}^{2^n-1} \gamma_i |i\rangle$, with $\gamma_i \in \mathbb{C}$, $\sum_{i=0}^{2^n-1} |\gamma_i|^2 = 1$ and $|i\rangle$ denoting the ith unit state.

In the context of QNN, parameterized quantum circuits (PQC) function as the core units rather than classical hidden layers. Fig. 3 illustrates a PQC that involves a pair of qubits interconnected with three quantum gates. The first qubit undergoes a Hadamard gate H and the second qubit experiences a rotation around the y-axis $R_y(w^r)$. A

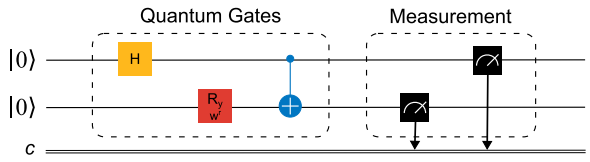

Fig. 3. Schematic representation of a PQC involving a pair of qubits. It includes both parameterized gate $R_y(w^r)$ with adjustable parameter w^r, and non-parameterized gate H. $CNOT$ gate entangles the second qubit to the first, executing a two-qubit operation. Finally, the measurement is performed to determine the expectation values of the involved qubits.

controlled NOT gate $CNOT$ entangles the second qubit to the first. The probabilistic expected value of the measurement, i.e. the expectation value of the probability distribution of the final quantum state, is then determined. The expectation value is represented by $\langle\psi|\psi\rangle$, which is subsequently subjected to post-processing to form the approximations.

Similar to NNs, QNNs can be fine-tuned through classical optimization processes to find the relationship between $\vec{x}$ as input features and $\vec{y}$ as output labels. $\vec{x}$ is initially encoded into a quantum state of multiple qubits, a phase known as feature map. Subsequently, the data is processed using PQCs and form an ansatz. The measurement of the PQCs is fed into a loss function. Optimization of QNNs can happen through gradient-based methods, facilitated by the analytical techniques for measuring the gradient of the expectation of PQCs, as outlined in [32]. The representation of QNNs is given by:

$$|\psi^{\text{in}}\rangle = U(\vec{x})|0 \dots 0\rangle, \quad |\psi^{\text{out}}\rangle = \omega(\vec{w}^r)|\psi^{\text{in}}\rangle, \tag{5}$$

where the input state $|\psi^{\text{in}}\rangle$ is created by transforming $\vec{x} \in \{(\vec{p}_i, \vec{q}_i) : i = 1, 2, \dots, n\}$ into valid quantum states using the feature map $U(.)$ applied to the vacuum state $|0 \dots 0\rangle$. The ansatz $\omega(.)$ is applied to $|\psi^{\text{in}}\rangle$. There are no parameters for the feature map that need to be tuned by the optimizer. In contrast, the vector of adjustable parameters of the ansatz $\vec{w}^r$ is fine-tuned through the training process using a dataset consisting of N training pairs $\{\vec{x}, \vec{y}\}$. The resulting output state $|\psi^{\text{out}}\rangle$ cannot be read out directly but needs to be deduced by the measurement to obtain $\hat{\vec{y}} \in \{(\vec{v}_i, \vec{\delta}_i) : i = 1, 2, \dots, n\}$.

This paper explores two distinct implementations of QNNs: a pure QNN and a hybrid quantum–classical neural network (QCNN), described in the following subsections.

4.1. Quantum neural networks

Fig. 4 shows an illustration of pure QNNs, which includes three fundamental components: a feature map, an ansatz, and the measurement. The feature map $U(\cdot)|0 \dots 0\rangle$ transforms $\vec{x} \in \{(\vec{p}_i, \vec{q}_i) : i = 1, 2, \dots, n\}$ into a vector of quantum states $|\psi^{\text{in}}\rangle$. The dimension of this vector corresponds to the number of qubits in the PQC, i.e. $n \times 2$, where n is the number of buses. The ansatz $\omega_{(.)}|\psi^{\text{in}}\rangle$ takes $|\psi^{\text{in}}\rangle$ as input and applies a combination of quantum gates. Finally, the expectation value is determined, which is then subjected to additional post-processing to derive $\hat{\vec{y}} \in \{(\vec{v}_i, \vec{\delta}_i) : i = 1, 2, \dots, n\}$.

4.2. Hybrid quantum-classical neural networks

For the QCNN, one or more of the hidden layers can be replaced by PQC(s), as shown in Fig. 5. The data flow begins from a hidden layer, propagates through the feature map and ansatz, and then proceeds to the next hidden layer after the measurement. Note that the QCNN architecture can potentially resemble classical auto-encoders, where the first classical component encodes $\vec{x} \in \{(\vec{p}_i, \vec{q}_i) : i = 1, 2, \dots, n\}$ to a lower dimensional latent space at which the QNN operates. Subsequently, the second classical component decodes the measurement to a higher dimensional space to derive $\hat{\vec{y}} \in \{(\vec{v}_i, \vec{\delta}_i) : i = 1, 2, \dots, n\}$. This

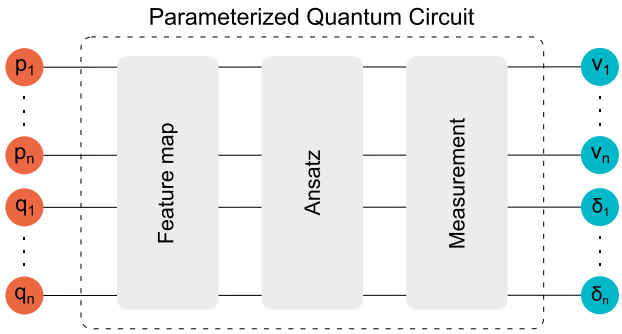

Fig. 4. Schematic depiction of the QNN designed for PF analysis, including a feature map, an ansatz, and the measurement. $\vec{x} \in \{(\vec{p}_i, \vec{q}_i) : i = 1, 2, \ldots, n\}$ and $\vec{y} \in \{(\vec{v}_i, \vec{\delta}_i) : i = 1, 2, \ldots, n\}$ respectively represent the input features and output labels, where n is the number of buses.

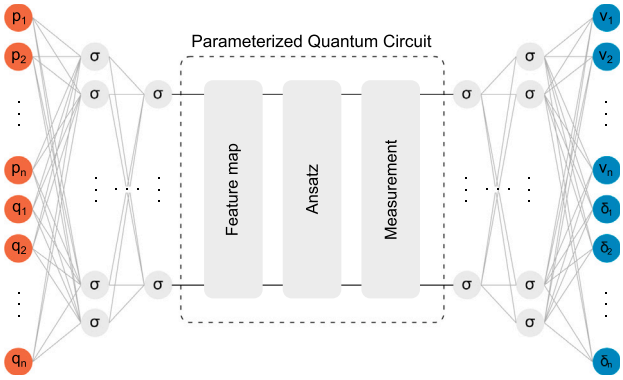

Fig. 5. Schematic depiction of the QCNN designed for PF analysis with a PQC replacing one hidden layer. $\vec{x} \in \{(\vec{p}_i, \vec{q}_i) : i = 1, 2, \ldots, n\}$ and $\vec{y} \in \{(\vec{v}_i, \vec{\delta}_i) : i = 1, 2, \ldots, n\}$ respectively represent the input features and output labels, where n is the number of buses. The data flow starts from a hidden layer, passes through the PQC, and continues to the subsequent hidden layer after the measurement.

architecture has therefore the flexibility to accommodate larger input features and output labels for large-scale power systems, as it is not constrained by the number of available qubits.

In this study, two QCNNs are developed for the 4-bus and 33-bus test systems, each having a PQC as the quantum component sandwiched with classical hidden layers, as depicted in Fig. 5. The number of classical hidden layers is the same as those of the corresponding NNs for the 4-bus and 33-bus test systems. The immediate hidden layer before and after the quantum component has $n \times 2$ neurons, where n is the number of buses. Note that in this setting, the total number of calls to the quantum simulator matches that of the QNN as the qubit count, and the number of shots remains consistent.

5. Results

Experiments are performed on modified versions of 4-bus and 33-bus test systems [29,30]. The chosen systems contain one reference bus with known v and δ and unknown p and q, and PQ buses for which p and q are known, while v and δ are unknown. Note that PV buses are not considered in this work to maintain computational simplicity.

For the two test systems, the dataset comprises a total of 512 data points, which are randomly chosen from a pool of 5000 samples. These samples are systematically generated based on calculated apparent power s and power factor pf for each bus. Initially, p and q are known for a specific scenario of the test systems, as presented in [29,30]. Subsequently, $s = \sqrt{p^2 + q^2}$, and $pf = p/s$ are calculated for each bus. The calculated s is considered as the mean and a deviation of 30% from s is considered as the standard deviation to obtain a normal distribution

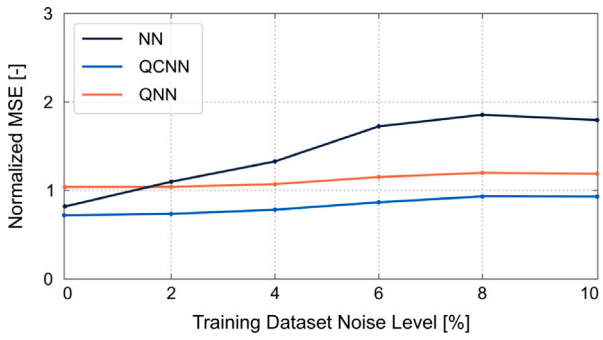

Fig. 6. Illustration of the performance of the NN, QCNN, and QNN based on the MSE obtained for the testing dataset under varying levels of noise in the training dataset for the 4-bus test system. The MSE values are normalized relative to the testing MSE obtained for the QNN at the noise level of 10%.

with 5000 samples for each bus. Finally, $p = s \times pf$ and $q = \sqrt{s^2 - p^2}$ are calculated for all buses and all samples. This approach ensures that the datasets reflect the inherent variability in p and q across the power system.

The input features and output labels of the datasets are respectively $\vec{x} \in \{(\vec{p}_i, \vec{q}_i) : i = 1, 2, \ldots, n\}$ and $\vec{y} \in \{(\vec{v}_i, \vec{\delta}_i) : i = 1, 2, \ldots, n\}$, obtained from the NR. The dataset for each test system is divided into three subsets, allocating 25% for training, 25% for validation, and the remaining 50% for testing. The training dataset intentionally includes only 128 training data points to highlight the enhanced performance of the QNNs. The training process concludes after 1000 epochs for the NN, QCNN, and QNN. The utilized loss function is the MSE (3), the activation function is ReLU, and the Adam optimization algorithm is employed.

5.1. Model performance

The performance of the NN, QCNN, and QNN is systematically evaluated based on (i) generalization ability, (ii) robustness, (iii) impact of training dataset size on generalization ability, (iv) training error, and (v) the stability of the training process. Experiments are done for the 4-bus test system.

5.1.1. Generalization ability

The MSE obtained for the testing dataset for the QNN and QCNN is 41% and 52% lower, respectively, compared to that of the NN.

5.1.2. Robustness

The influence of noisy training dataset is investigated by systematically introducing controlled levels of noise, ranging from 1% to 10%, to both $\vec{x} \in \{(\vec{p}_i, \vec{q}_i) : i = 1, 2, \ldots, n\}$ and $\vec{y} \in \{(\vec{v}_i, \vec{\delta}_i) : i = 1, 2, \ldots, n\}$ of the training points. A certain percentage of the input features and output labels of the training dataset is randomly selected. The corrupted vectors are $\vec{x'} = \vec{x} \pm \vec{r}_x$ and $\vec{y'} = \vec{y} \pm \vec{r}_y$, where $\vec{r}_x$ and $\vec{r}_y$ are vectors of random values between 0 and 1, and 0 and 0.1, respectively. The investigation is based on the testing MSE. The QNN and QCNN exhibit superior robustness against the noisy training dataset compared to the NN, as shown in Fig. 6. The performance of the NN significantly decreases by up to two times as the noise level increases.

5.1.3. Training dataset size

The impact of training dataset size on the performance of the NN is investigated. Fig. 7 shows the changes in the MSE obtained for the testing dataset with varying training dataset sizes. The comparison is made relative to a constant curve, which represents the MSE obtained for the testing dataset for the QCNN using 128 training data points. It is observed that the NN necessitates a training dataset four times larger than that of the QCNN to achieve a performance level that still remains inferior.

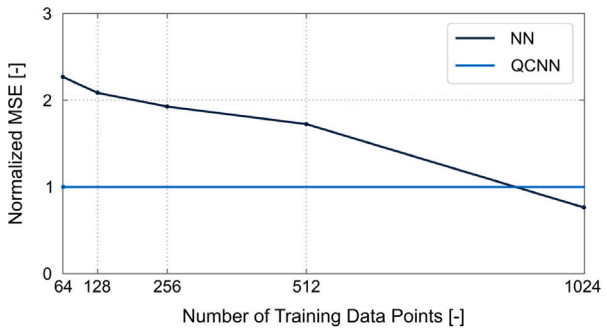

Fig. 7. Illustration of the performance of the NN based on the MSE obtained for the testing dataset for the 4-bus test system under different training dataset sizes, compared to that of the QCNN with 128 training data points relative to which the MSE values are also normalized.

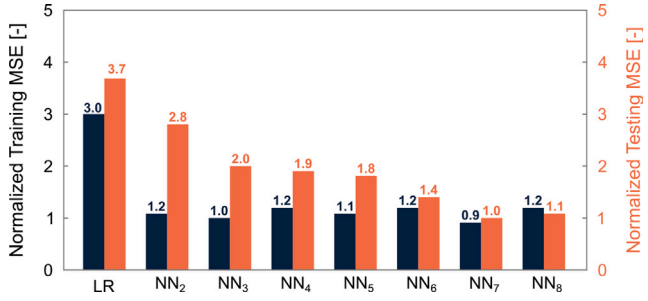

Fig. 8. Illustration of the impact of the NN architecture depth on the training and testing MSE for the 4-bus test system. Linear regression (LR) serves as a benchmark model. NN architectures with varying numbers of hidden layers, ranging from two (NN_2) to eight (NN_8), are assessed. The MSE values are normalized relative to the testing MSE obtained for the NN_7.

5.1.4. Training error

The MSE obtained for the training dataset for the QNN and QCNN after 1000 epochs is 48% and 54%, respectively, less than that of the NN.

5.1.5. Training process stability

The mean and standard deviation of the MSE obtained over 1000 epochs for the training dataset is evaluated. The QNN and QCNN show a respective reduction of 39% and 45% in the mean of the training MSE compared to the classical NN. Similarly, the standard deviation of the training MSE for the QNN and QCNN is lower by 25% and 34%, respectively, compared to the classical NN. Note that the differences in the training processes are due to the differences in the gradient calculation methods employed. For the classical NN and the classical component of the QCNN, the gradients are computed using backpropagation [33]. However, for the pure QNN and the quantum component of the QCNN, gradient calculation is achieved using the parameter shift rule [32].

5.2. Sensitivity analysis for NN

The architecture of the NN, i.e. the number of hidden layers and neurons per hidden layer, but also the hyper-parameters, i.e. the learning rate, the weight decay rate, and dropout percentage, are achieved through sensitivity analysis to ensure a fair comparison between the NN, QCNN, and QNN[1]:

[1] The Optuna Python package is used to conduct an exhaustive search across 2000 unique hyperparameter combinations.

5.2.1. Model architecture

Seven different depths of NN architectures are explored to ascertain the architecture that enhances the generalization ability of the NN. Deep architectures with more than eight hidden layers are not considered due to the small size of both the test system and the training dataset. Across all architectures, the first and last hidden layers consist of $n\times2$ neurons, while the intermediate hidden layers each contain $n\times4$ neurons. A linear regression model (LR) is included in the evaluation as a benchmark. Fig. 8 illustrates the MSE obtained for both the training and testing datasets. It is observed that as the number of hidden layers decreases, the generalization ability of the NN decreases.

5.2.2. Hyper-parameters

An exploration of different learning rates, weight decay rates, and dropout percentages is conducted. The learning rates are explored across a range of values between 1×10^{-1} and 1×10^{-6}. Wight decay rates are considered from 1×10^{-1} to 1×10^{-6}. Dropout percentages are considered between 0% and 2%. The selection of the optimal values is based on the training and testing MSE, along with their relative proximity to ensure generalization ability.

The architecture with seven hidden layers is selected as the basis for evaluating the QCNN and QNN due to its lower MSE obtained for training and testing datasets. The proximity of the training and testing MSE also suggests a lack of overfitting in the proposed architecture. The NN is trained with a learning rate of 1.5×10^{-4}, a weight decay rate of 3×10^{-3}, and a dropout percentage of 0%, with a batch size of 16. The training and testing MSE obtained for the 4-bus test system and a noisy training dataset with 128 training data points are 2.41 and 2.83, respectively.

5.3. Sensitivity analysis for QNN

A PQC serves as the core unit of the QNN architecture. It comprises a feature map, an ansatz, and the measurement, as illustrated in Fig. 9. The feature map is responsible for translating the six input features into quantum states. The ZFeatureMap method from the Qiskit standard library is used, that is, the vacuum state $|000000\rangle$ passes through a layer of Hadamard gates H before each qubit undergoes a z-rotation, i.e. $R_z(h(p,q))$, where $h(\cdot)$ maps the input features $\vec{x}\in\{(\vec{p}_i,\vec{q}_i):i=1,2,\dots,n\}$ to the interval $[-\frac{\pi}{2},\frac{\pi}{2}]$ through the formula $\vec{x'}=\tan^{-1}(\vec{x})$. The RealAmplitude method from the Qiskit standard library is employed as the ansatz. This entails each qubit undergoing a y-rotation $R_y(\vec{w}_i^r):i=1,2,\dots,n\times2$, followed by a CNOT gate, and another y-rotation $R_y(\vec{w}_i^{r'}):i=1,2,\dots,n\times2$. Consequently, the ansatz has $n\times4$ free parameters, where n is the number of buses. From the measurement in the Z-basis, the expectation value for each qubit is obtained within the interval $[-1,1]$. A classical post-processing conversion is then employed to map these values to problem-specific intervals. These intervals are determined based on the lower and upper bounds of the dataset with an additional margin to ensure coverage of extreme values. Accordingly, the intervals for v and δ are determined as $[0.85,1.15]$ and $[-8,8]$, respectively. Details about the sensitivity analysis for the QNN are provided in the following subsections.

5.3.1. Convergence analysis of shots

The error between the predictions obtained from the Aer statevector simulator and those obtained from the shot simulator is investigated, where each shot represents a repetition of the measurement. The results are shown in Fig. 10. It can be observed that the error stabilizes when the shot count reaches 1024 or higher. Therefore, a shot number of 1024 is selected for the training of the QNN and the quantum component of the QCNN.

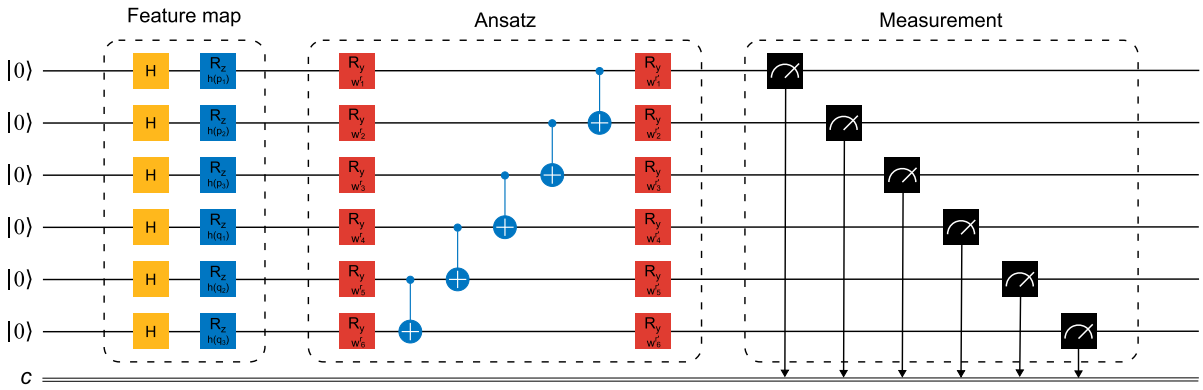

Fig. 9. Detailed representation of the PQC shown in Fig. 4 for the 4-bus test system. $\vec{x} \in \{(\vec{p}_i, \vec{q}_i) : i = 1, 2, \ldots, n\}$ is initially encoded into a quantum state of multiple qubits using the feature map. Then, the data is processed through the ansatz. The feature map utilizes the Hadamard gate H and a rotation around the z-axis R_z on individual qubits. The ansatz applies a y-rotation $R_y(\vec{w}_i^r)$, followed by a CNOT gate, and subsequently another y-rotation $R_y(\vec{w}_i^{r'})$ on each qubit. The expectation value for each qubit is then obtained within the interval of $[-1, 1]$ from the measurement in the Z-basis.

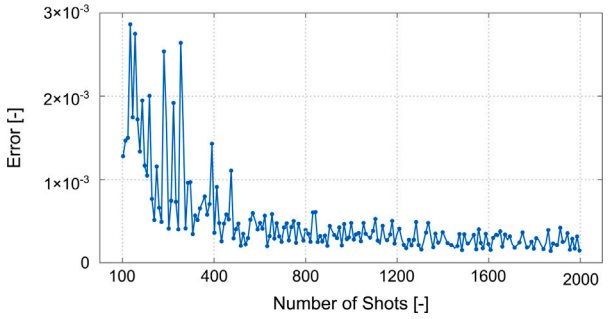

Fig. 10. Illustration of the error between the predictions obtained from the Aer statevector simulator and those from the shot simulator as a function of the number of shots.

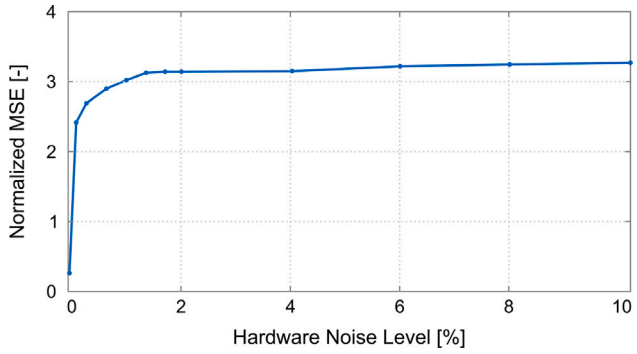

Fig. 11. Illustration of the impact of various noise levels on the performance of the QNN for the 4-bus test system. Measurement error, gate imperfection, depolarizing error, and amplitude-damping error are considered.

5.3.2. Impact of hardware noise

The impact of different sources of noise, including measurement error, gate imperfection, depolarizing error, and amplitude-damping error, is examined. This exploration simulates the real-world behavior of quantum hardware. The noise levels range from 0% to 10% and the analysis utilizes the dataset for the 4-bus test system. The results are depicted in Fig. 11. Details about different error types can be found in Ref. [34].

5.3.3. Impact of the number of qubits

Increasing the number of qubits inherently complicates the training process, particularly for the QCNN which integrates both quantum and classical components. The impact of qubit count on the MSE obtained for the training and testing datasets is therefore studied to highlight the trade-offs between the model complexity and model performance. The analysis uses the dataset for the 4-bus test system. The results, shown in Fig. 12, lead to the selection of six qubits for the QCNN.

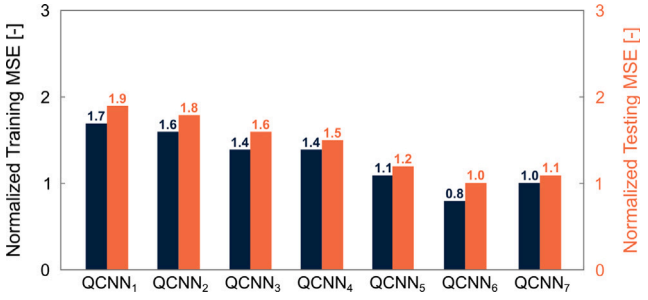

Fig. 12. Illustration of the impact of the qubit count on the training and testing MSE for the 4-bus test system. Different QCNNs with varying numbers of qubits, ranging from one ($QCNN_1$) to seven ($QCNN_7$), are assessed. The MSE values are normalized relative to the testing MSE obtained for the $QCNN_6$.

Table 2
Performance comparison of the NN relative to the results obtained by the NR method for the 4-bus test system.

Approach[a]	mean [p.u]	std [p.u]	mean [degree]	std [degree]
LR	8.85×10^{-1}	6.74×10^{-2}	9.53×10^{-1}	4.02×10^{-2}
NN	1.44×10^{-1}	4.02×10^{-2}	3.18×10^{-1}	3.24×10^{-2}
QCNN	2.67×10^{-3}	1.03×10^{-2}	1.23×10^{-2}	2.4×10^{-2}
QNN	6.1×10^{-3}	3.08×10^{-2}	2.26×10^{-2}	2.98×10^{-2}

[a] LR = linear regression model, NN = classical neural network, QCNN = hybrid quantum–classical neural network, QNN = quantum neural network.

5.4. Power flow analysis

The mean and standard deviation of $\hat{\vec{y}} \in \{(\hat{\vec{v}}_i, \hat{\vec{\delta}}_i) : i = 1, 2, \ldots, n\}$ approximated by the NN, QCNN, and QNN for the 4-bus test system are computed relative to the ground truth data $\vec{y} \in \{(\vec{v}_i, \vec{\delta}_i) : i = 1, 2, \ldots, n\}$ obtained from the NR, see Table 2. A linear regression model (LR) is also included in the evaluation as a benchmark. The mean and standard deviation obtained for the QCNN exhibit the superior performance of the QCNN compared to the QNN, NN, and LR.

An investigation is also done to explore the performance of the NN and QCNN for the 33-bus test system under extreme conditions, where the power system experiences over-voltage or under-voltage situations. According to the results shown in Fig. 13, the QCNN achieves a maximum MSE of 0.011 pu for the testing dataset, while the NN

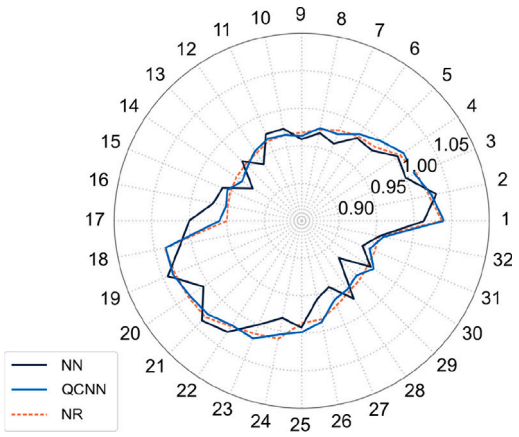

Fig. 13. Comparison of $\vec{v}$ obtained from the NR and $\vec{v}$ obtained from the NN and QCNN for the 33-bus test system under extreme conditions.

obtains a maximum MSE of 0.032 pu for the testing dataset, compared to the ground truth data obtained by the NR. This improvement is significant for power system operations, where unexpected events can induce deviations in v from the nominal value.

6. Discussion

Quantum neural networks (QNNs) are developed for power flow (PF) analysis with the aim of providing fast and accurate estimations compared to the traditional PF solvers, such as the Newton–Raphson method (NR). In this sense, deep learning approaches do not aim to replicate traditional PF solvers. Instead, these approaches serve as surrogate models, wherein complex relationships between inputs and outputs are captured through linear and nonlinear transformations. While the same inputs and outputs can be shared by traditional solvers, e.g. the NR, and surrogate models, e.g. neural networks, their difference lies in the computational process employed to derive the output from the provided input. Generally, the success of a surrogate model depends on the size and quality of historical data on inputs and outputs. In contrast, traditional solvers rely on mathematical models to obtain outputs from inputs. Note that the output of QNNs can serve as an initial estimate, which can then be further refined by conducting a few iterations of the NR method to converge towards a solution sufficiently fast.

In this work, quantum simulations are conducted using the Aer statevector simulator of Qiskit. However, real quantum hardware, such as that provided by IBM, remains an available option for future research and experimentation. Note that the current high noise levels in today's NISQ hardware make it impractical to implement the proposed QNN approaches. However, it is expected that as NISQ hardware evolves and early Fault-Tolerant (FT) computers with reduced noise levels and a moderate number of logical qubits emerge, the proposed approaches will become viable for successful execution.

Although this work shows the potential of QNNs for PF analysis, and their superior performance compared to classical NNs, further investigation is needed to fully understand the added value of using such complex approaches. Note also that the superior performance of the hybrid quantum–classical neural network (QCNN) in terms of generalization, robustness, training dataset size needed, training error, and training process stability compared to the classical NN can be attributed to the capacity of the QCNN to leverage the strengths of both classical and quantum paradigms, which makes it a more robust choice, particularly in the NISQ era. On the other hand, the limited performance of the classical NN can be attributed to the limited number of training data points and the noises introduced into the training dataset.

7. Conclusion

This paper systematically investigates the application of quantum neural networks for power flow analysis. A comprehensive comparison of a classical neural network (NN), a hybrid quantum–classical neural network (QCNN), and a quantum neural network (QNN) is performed. The comparison is based on (i) generalization ability, (ii) robustness, (iii) training dataset size needed, (iv) training error, and (v) training process stability. Experimentation and sensitivity analyses are conducted on 4-bus and 33-bus test systems. The results indicate the superiority of the QCNN over the NN and QNN.

The QNN and QCNN exhibit enhanced generalization ability compared to the NN by 41% and 52%, respectively, when trained on a small noisy training dataset of 128 data points. In terms of robustness against noisy training data, the QNN and QCNN outperform the NN. In addition, it is observed that the NN necessitates a training dataset approximately four times larger than that of the QCNN to achieve a performance that still remains inferior. The training error obtained for the QNN and QCNN is 48% and 54% less than that of the NN. Furthermore, the QNN and QCNN show a more stable training process compared to the NN.

CRediT authorship contribution statement

Zeynab Kaseb: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft. **Matthias Möller:** Conceptualization, Supervision, Writing – review & editing, Validation. **Giorgio Tosti Balducci:** Conceptualization, Validation. **Peter Palensky:** Funding acquisition. **Pedro P. Vergara:** Funding acquisition, Supervision, Writing – review & editing, Validation.

Declaration of competing interest

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

Data availability

Data will be made available on request.

Acknowledgments

This work is part of the DATALESs project (with project number 482.20.602) jointly financed by the Netherlands Organization for Scientific Research (NWO), and the National Natural Science Foundation of China (NSFC). This work used the Dutch national e-infrastructure with the support of the SURF Cooperative using grant number EINF-6569. Acknowledgments go to these organizations for their support.

References

[1] O.D. Montoya, W. Gil-González, A. Garces, Numerical methods for power flow analysis in DC networks: State of the art, methods and challenges, Int. J. Electr. Power Energy Syst. 123 (2020).
[2] K. Baker, Solutions of DC OPF are never AC feasible, 2019.
[3] J.S. Giraldo, O.D. Montoya, P.P. Vergara, F. Milano, A fixed-point current injection power flow for electric distribution systems using Laurent series, Electr. Power Syst. Res. 211 (2022).
[4] T. Pham, X. Li, Neural network-based power flow model, in: 2022 IEEE Green Technologies Conference, GreenTech.
[5] B. Donon, R. Clément, B. Donnot, A. Marot, I. Guyon, M. Schoenauer, Neural networks for power flow: Graph neural solver, Electr. Power Syst. Res. 189 (2020).
[6] S.M. Miraftabzadeh, F. Foiadelli, M. Longo, M. Pasetti, A survey of machine learning applications for power system analytics, in: 2019 IEEE International Conference on Environment and Electrical Engineering, EEEIC.

[7] F. Feng, Y. Zhou, P. Zhang, Quantum power flow, IEEE Trans. Power Syst. 36 (2021).
[8] L. von Rueden, S. Mayer, K. Beckh, B. Georgiev, S. Giesselbach, R. Heese, B. Kirsch, M. Walczak, J. Pfrommer, A. Pick, R. Ramamurthy, J. Garcke, C. Bauckhage, J. Schuecker, Informed machine learning - A taxonomy and survey of integrating prior knowledge into learning systems, IEEE Trans. Knowl. Data Eng. (2021).
[9] C.C. McGeoch, Adiabatic Quantum Computation and Quantum Annealing Theory and Practice, Springer Cham, 2014.
[10] F. Feng, P. Zhang, Y. Zhou, Z. Tang, Quantum microgrid state estimation, Electr. Power Syst. Res. 212 (2022).
[11] S. Golestan, M. Habibi, S. Mousazadeh Mousavi, J. Guerrero, J. Vasquez, Quantum computation in power systems: An overview of recent advances, Energy Rep. 9 (2023).
[12] F. Feng, Y. Zhou, P. Zhang, Noise-resilient quantum power flow, 2022.
[13] T. Morstyn, Annealing-based quantum computing for combinatorial optimal power flow, IEEE Trans. Smart Grid 14 (2023).
[14] M. Schuld, I. Sinayskiy, F. Petruccione, The quest for a Quantum Neural Network, Quantum Inf. Process. 13 (2014).
[15] K. Beer, D. Bondarenko, T. Farrelly, T.J. Osborne, R. Salzmann, D. Scheiermann, R. Wolf, Training deep quantum neural networks, Nature Commun. 11 (2020).
[16] K. Beer, Quantum neural networks, 2022, Institutionelles Repositorium der Leibniz Universitat Hannover.
[17] R. Eskandarpour, K. Ghosh, A. Khodaei, L. Zhang, A. Paaso, S. Bahramirad, Quantum computing solution of DC power flow, 2020.
[18] Y. Zhou, P. Zhang, Noise-resilient quantum machine learning for stability assessment of power systems, 2021.
[19] B. Sævarsson, S. Chatzivasileiadis, H. Jóhannsson, J. Østergaard, Quantum computing for power flow algorithms: Testing on real quantum computers, 2022.
[20] N. Nikmehr, P. Zhang, M.A. Bragin, Quantum distributed unit commitment: An application in microgrids, IEEE Trans. Power Syst. 37 (2022).
[21] H. Jing, Y. Wang, Y. Li, Data-driven quantum approximate optimization algorithm for power systems, Commun. Eng. 2 (2023).
[22] F. Amani, R. Mahroo, A. Kargarian, Quantum-enhanced DC optimal power flow, in: 2023 IEEE Texas Power and Energy Conference, TPEC.
[23] A. Ajagekar, F. You, Quantum computing based hybrid deep learning for fault diagnosis in electrical power systems, Appl. Energy 303 (2021).
[24] P. Halffmann, P. Holzer, K. Plociennik, M. Trebing, A quantum computing approach for the unit commitment problem, 2022.
[25] P.W. Sauer, M.A. Pai, J.H. Chow, Power System Dynamics and Stability with Synchrophasor Measurement and Power System Toolbox, Wiley, 2017.
[26] H.T. Alrikabi, I. A. Aljazaery, J.S. Qateef, A.H.M. Alaidi, R.M. Al_airaji, Face patterns analysis and recognition system based on quantum neural network QNN, Int. J. Interact. Mob. Technol. (iJIM) 16 (2022).
[27] A. Sagingalieva, M. Kordzanganeh, N. Kenbayev, D. Kosichkina, T. Tomashuk, A. Melnikov, Hybrid quantum neural network for drug response prediction, Cancers 15 (2023).
[28] E. Paquet, F. Soleymani, QuantumLeap: Hybrid quantum neural network for financial predictions, Expert Syst. Appl. 195 (2022).
[29] J.J. Grainger, W.D. Stevenson Jr., Power System Analysis, McGraw-Hill, Inc., 1994.
[30] M. Baran, F. Wu, Network reconfiguration in distribution systems for loss reduction and load balancing, IEEE Trans. Power Deliv. 4 (1989).
[31] J. Arrillaga, B. Smith, AC-DC Power System Analysis, Institution of Electrical Engineers, 1998.
[32] K. Mitarai, M. Negoro, M. Kitagawa, K. Fujii, Quantum circuit learning, Phys. Rev. A 98 (2018).
[33] J. Fang, M. Xu, H. Chen, B. Shuai, Z. Tu, J. Tighe, An in-depth study of stochastic backpropagation, 2022.
[34] H.-L. Huang, X.-Y. Xu, C. Guo, G. Tian, S.-J. Wei, X. Sun, W.-S. Bao, G.-L. Long, Near-term quantum computing techniques: Variational quantum algorithms, error mitigation, circuit compilation, benchmarking and classical simulation, Sci. China Phys. Mech. Astron. 66 (2023).