Quantum Deep Learning for Energy-Efficient Power Control in Wireless Communication Systems

Su Fong Chien*, Heng Siong Lim‡, Alessio Zappone§ and Charilaos C Zarakovitis†

***Abstract*—The advent of quantum technology promises major leaps forward in addressing the ever-during problems of wireless resource control and system network optimisation. Solutions to these problems have, so far, been proven paramount for improving key performance measures, like the energy-efficiency, spectral-efficiency, transmission latency, etc., at close-to-zero operational time, yet, their transition from theory to practice is burdened by their high implementation complexity. This paper aims at directing and implementing the energy-efficiency resource allocation problem by introducing unique quantum neural network modelling with innovative quantum deep-learning process for optimizing the power allocations in real-time. We demonstrate that our quantum deep-learning solution requires a much shorter training time compared to classical artificial neural network approaches, which significantly boosts the computational speed with the guaranteed convergence, reliability, and practicability.**

***Index Terms*— energy-efficiency, quantum computing, quantum deep neural networks, quantum entanglement, quantum machine learning, quantum superpositions, resource allocation.**

I. INTRODUCTION

The intelligence of resolving complex problems in engineering, science, finance, etc., has become increasingly dependent on Machine Learning methods and, especially, deep-learning approaches driven by Artificial Neural Networks (ANNs). In the communications engineering regime, for example, the studies in [1]-[2] have recently utilized deep-learning to resolve the Non-deterministic Polynomial-time (NP-hard) problem of energy efficiency by means of training an ANN architecture with a set of optimal power allocations, so as, the system can learn how to autonomously predict the optimal power control policy with respect to the varying wireless channel conditions.

Yet, by leveraging deep-learning to Quantum deep-learning, ANNs can produce results not only comparable, but also far superior to the existing studies and findings. This is mainly due to the ability of Quantum deep-learning in performing computationally-intensive tasks by amalgamating the versatility of neural networks with the computing power of quantum neurons [3]-[4]. Evidently, the Google's Sycamore 54-qubit quantum processor has been recently shown to resolve in 200 seconds a complex task, which takes up to 10,000 years to be resolved with state-of-the-art classical super-computers (Oct 2019). As such, Quantum deep-learning based on ANN rationale have become top research topic, with Industry and Academia to seek for more sophisticated ways in maximising the computational potentials of quantum-oriented algorithms.

Within this direction, we notice that the artificial perceptron unit, from which ANNs are constructed, cannot extend to the actual capacity of the quantum perceptron. This is because the ANN perceptron activation functions are non-linear in nature and do not exactly match with the mathematical structure of quantum rationale, which is described by linear operations. That means, although classical ANN training can exploit the advantages of quantum information, its complexity can be further and sufficiently decreased upon embedding it into Quantum Neural Network (QNN) formalism. Notably, literature reports quite limited QNN architectures that have been proven to train the system based of Quantum deep-learning. For instance, the QNN subroutine in [5] attains storing intermediate results in quantum random access memory, which speeds up the training time compared to ANN algorithms. The study in [6], models the quantum neurons as natural generalizations of perceptrons in feed-forward QNN machines, while authors in [7] use a qubit-circuit setup to facilitate the system to learn classical data assisted by quantum algorithm.

Bearing the quantum technology landscape and its horizon, this paper introduces a unique QNN modelling tailored to execute Quantum deep-learning towards resolving highly-complex problems in wireless communications engineering. Our focus lays on optimizing the power control of the energy efficiency problem, which is a type of sum-of-ratios problem with NP-hard computational complexity meaning that it belongs to the hardest class of problems to resolve and integrate in practical system setting. For the sake of clarity, we initially show how the optimal power control policy can be approached using traditional ANN-oriented Quantum deep-learning. We then present the proposed QNN deep-learning modelling, which innovates by facilitating the computation of the parameter matrices of each perceptron unitary operator in a layer-by-layer fashion (i.e. without any need to rely on additional unitary operators of the full quantum system) and contributes in a threefold manner highlighted in the following.

i) The proposed QNN model computes the trace over all qubits accounting the total number of perceptrons over all system layers, meaning that it introduces sufficiently smaller step size than ANN training, which typically computes the trace of limited qubits accounting the perceptron that acts on the running and previous layers. Thereby, we substantially speed up convergence towards the optimal power control policy;

ii) To calculate the training function at each QNN training step, our model formulates the non-unitary stochastic transition mapping of the overall system by considering the system parameters of two layer only, meaning that the size of the parameter matrices depends only on the width of the network, rather than the width of each layer as required in ANN. Thereby, we save considerable memory resources.

iii) Due to its small step size and light-weight mapping, our QNN training process can be approached in similar algorithmic logic as in the classical ANN systems (i.e. by following a quantum analogue of back-propagation). Thereby, we

* Advanced Informatics Lab, MIMOS Berhad, Technology Park Malaysia, Kuala Lumpur, 57000, Malaysia (sf.chien@mimos.my)

‡ Faculty of Engineering and Technology, Multimedia University, Melaka, Malaysia (hslim@mmu.edu.my)

§ University of Cassino and Southern Lazio, Cassino, 03043, Italy (alessio.zappone@unicas.it)

† Institute of Informatics and Telecommunications, National Centre for Scientific Research "DEMOKRITOS" (NCSRD), Athens, 153 10, Greece (c.zarakovitis@iit.demokritos.gr)

develop a highly practicable QNN-oriented deep-learning paradigm that can be implemented by classical computers.

To the best of our knowledge, our QNN modelling is not only unique, but also the first to consider resolving highly-complex problems in the wireless communications regime.

The remainder of the paper is as follows. Section II formulates the network system modelling and the energy-efficiency problem of power control. Section III and Section IV present the optimisation procedure of the problem at hand using generalized ANN approach and the proposed QNN paradigm, respectively. Section V evaluates our findings using numerical simulations, with Section VI to conclude the paper.

II. SYSTEM MODEL AND PROBLEM FORMULATION

To identify the key parameters of the proposed QNN training process with respect to those utilised in ANN approaches, we draw our focus on a widely-adopted system modelling, which structurizes on the uplink transmissions of a multi-cell network of total bandwidth B and L total number of users serviced by M total number of Base Stations (BSs) each one equipped with N antennas. Assuming standard interference effects between the channels of user $i = 1, 2, ..., L$ and user $j \neq i$ at each BS $m = 1, 2, ..., M$ and by recalling the Shannon capacity theorem, we can describe the maximum achievable data rate of user i at its associated BS m_i as[1]

$$R_i = B \cdot \log_2 \left(1 + \frac{(\|\mathbf{h}_{m_i}\|^2 \cdot p_i)/\sigma_i^2}{1 + \sum_{j \neq i} \left(|\mathbf{h}_{m_i} \cdot \mathbf{h}_{m_{ij}}|^2 \cdot p_j \right)/\sigma_i^2} \right), \text{ (in bit)} \quad (1)$$

where (i) $\mathbf{h}_{m_{ij}} \in \mathbb{C}^N$ represents the channel association[2] between the j-th user and the BS m_i, (ii) σ_i^2 the thermal power of the zero-mean circularly symmetrical complex Gaussian channel noise, and (iii) p_i and p_j the transmitted powers from the i-th and j-th user to BS m_i, respectively. In view of (1), we formulate the most inclusive, yet, hardest-to-optimize expression between the available energy-efficiency metrics, so called, the Weighted Sum Energy Efficiency (WSEE), as [8]

$$WSEE_i = \frac{w_i \cdot R_i}{\mu_i \cdot p_i + C_i}, \text{ (in bit per Joule)} \quad (2)$$

where (i) μ_i is the coefficient of the efficiency of the transmit power amplifier, (ii) C_i the total circuit power consumption of user i and its associated BS m_i, and (iii) $w_i \geq 0$ the priority weight of each i-th link between users and BSs, i.e., considering that some users may perquisite for higher data rate than others, with $w = \{w_1, ..., w_i, ..., w_j, ..., w_L\}$. Using (2) and by setting that the transmitted power of each user is subject to an average transmitted power threshold, i.e., $p_i \leq P_i$ with $\boldsymbol{p} = [p_1, ..., p_i, ...p_j, ...p_L]$ and $\boldsymbol{P} = [P_1, ..., P_i, ..., P_j, ...P_L]$, we formulate the (generic) WSEE optimisation problem as

$$\begin{aligned} &\max_{\boldsymbol{p}} \sum_{i=1}^{L} WSEE_i \\ &\textit{subject to}: \ 0 \leq p_i \leq P_i, \ \forall i = 1, ..., i..., j..., L. \end{aligned} \quad (3)$$

Problem (3) is a type of sum-of-ratios problem with Non-deterministic Polynomial-time hardness (NP-hard) in its computational complexity meaning that it belongs to the hardest class of problems to resolve explicitly and integrate in practice. Traditionally, such complexities are tracked by transforming problems like (3) into simpler forms using variable relaxation methods [8], which, on the other hand, introduce considerable truncation error that compromises the convergence and accuracy of their solutions. To bypass such issues, the recent studies in [1]-[4], approach NP-hard types of problems with ANN modelling to obtain their solutions by anticipating the training result of the neural network. For the sake of clarity, in the next Section III we build such ANN model by elaborating on the parameters of the considered WSEE system modelling.

III. ARTIFICIAL NEURAL NETWORK ARCHITECTURE FOR THE ENERGY EFFICIENCY PROBLEM

The aim is to resolve problem (3) by deriving the power allocations $\boldsymbol{p}$ as closed-form functions of the propagation channels $\mathbf{h}_{m_i}$, $\mathbf{h}_{m_{ij}}$ and the power threshold parameters $\boldsymbol{P}$, so as, the computation of the optimal power allocation $\boldsymbol{p}^*$ can have a negligible complexity, i.e., to track the fading channel variations and update in real-time. To do so, we compact the numerator and denominator of the data rate expression in (1) by defining the channel realization vectors $\boldsymbol{\alpha} = [\alpha_1, ..., \alpha_i, ..., \alpha_L]$ and $\boldsymbol{\beta} = [\beta_1, ..., \beta_i, ..., \beta_L]$, with $\alpha_i = \|\mathbf{h}_{m_i}\|^2/\sigma_i^2$ and $\beta_{ij} = |\mathbf{h}_{m_i} \cdot \mathbf{h}_{m_{ij}}|^2/\sigma_i^2$, and re-structure the WSEE problem (3) using the map

$$\mathcal{F} : \mathbf{a} = (\alpha_i, \beta_{ij}, P_i)_{ij} \in \mathbb{R}^{L \cdot (L+1)} \mapsto \boldsymbol{p}^* \in \mathbb{R}^L, \quad (4)$$

where $\boldsymbol{p}^*$ the vector with elements the optimal power allocation coefficients of the L users corresponding to the M BS indexing, and $\mathbb{R}$ the set of real numbers. At this point, we recall the ANN universal approximation property, which yields that the input-output relationship of a feed-forward neural network with fully-connected layers is capable for emulating any continuous function [3]. Indeed, as $\boldsymbol{p} \geq 0$, the WSEE function in (2) is continuous, which facilitates approximating the map in (4) with an ANN model of fully-connected feed-forward layers, i.e., an input layer based on the realisation of the parameter vector $\mathbf{a}$ and an output layer based on the power allocation vector $\mathbf{p}$ that, in essence, stands as an estimate of the optimal power allocation vector $\boldsymbol{p}^*$ corresponding to input $\mathbf{a}$. Note that due to feed-forward not only these two input-output layers are created, but also K additional "hidden" neural layers are generated as well. To enumerate all neural layers, we label the input layer as *layer-0*, the first "hidden" layer as *layer-1* with N_1 neurons, the k-th "hidden" layer as *layer-k* with N_k neurons, and so on, until the output *layer-*$K+1$, which includes N_{K+1} neurons. In such fully-connected ANN system, each layer $k = 1, ..., K+1$ has N_k neurons in total, with each neuron n to be able computing[2]

$$\zeta_k(n) = f_{nk}\left(\boldsymbol{\gamma}_{nk}^T \cdot \boldsymbol{\zeta}_{k-1} + \delta_{nk} \right), \quad (5)$$

where (i) $\boldsymbol{\zeta}_k = \left(\zeta_k(1), \ldots, \zeta_k(N_{k+1})\right)$ represents the $N_{k+1} \times 1$ output vector of layer k, (ii) $\boldsymbol{\gamma}_{nk} \in \mathbb{R}^{N_{k+1}}$ the neuron-dependent weights, (iii) $\boldsymbol{\delta}_{nk} \in \mathbb{R}$ the neuron-dependent bias terms, and (iv) f_{nk} the activation function of neuron n in layer-k. From (5) we see that although each neuron can perform simple operations, combining the processing of multiple neurons in single ANN, the system can perform complex tasks and obtain the overall input-output map in (4) that can approximate virtually any type of optimisation function. The challenge, however, is how to configure the weights $\{\boldsymbol{\gamma}_{nk}\}$ and bias terms $\{\boldsymbol{\delta}_{nk}\}$ for achieving a desired approximation accuracy.

[1]More sophisticated modellings of the channel interference, as well as (later on) of the circuit power consumption are available in relevant literature and can be considered to enrich the data rate expression in (1) and the energy-efficiency expression in (2), respectively, yet, such insights are out of the scope of this work, which aims to describe the methodology of applying the new QNN approach on generic wireless communications system setting.

[2]The operators $\|\cdot\|$, $|\cdot|$, $(\cdot)^T$ $(\cdot)^\dagger$ denote the L^2-norm, absolute value transpose, matrix transpose and the Hermitian conjugate, respectively, while typeset in bold (e.g. $\mathbf{h}$) expresses variables in matrix form.

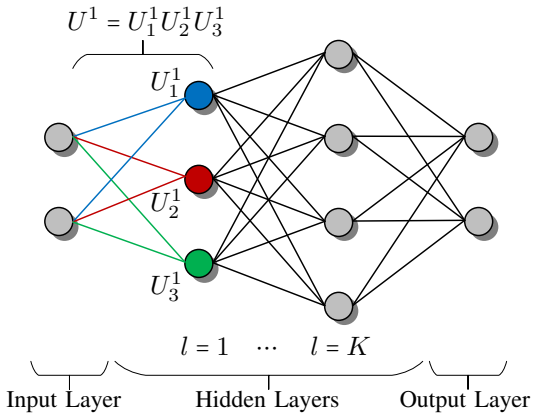

Fig. 1: Illustration of the channel transition between two layers during the ANN training process.

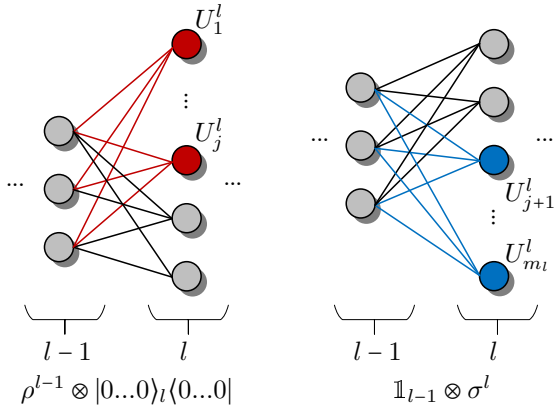

Fig. 2: Illustration of the channel transition between two layers during the QNN training process.

This can be resolved by training the considered ANN system using a supervised procedure, where a certain training set of optimal power allocations can be deployed as the collection of channel realisations $\{\boldsymbol{\alpha}_n\}_{n=1}^{N_T}$ and the corresponding optimal power allocations $\{\boldsymbol{p}_n^*\}_{n=1}^{N_T}$ of N_T training tuples $(\boldsymbol{a}_n, \boldsymbol{p}_n^*)$. In other words, the training set contains $\{\boldsymbol{p}_n\}_{n=1}^{N_T}$ examples of desired power allocation vectors corresponding to some possible configurations of system parameters $\boldsymbol{a}_n$, whereas, by exploiting these examples, the system can learn to predict the power allocation for new realizations of $\boldsymbol{a}_n$ that are not included in the running training set. More specifically, by setting $\mathcal{L}$ the error measure between actual and desired outputs in (5) the ANN training problem can be written as

$$\min_{\boldsymbol{\gamma},\boldsymbol{\delta}} \tfrac{1}{N_T} \textstyle\sum_{n=1}^{N_T} \mathcal{L}\left(\boldsymbol{p}_n(\boldsymbol{\gamma},\boldsymbol{\delta}), \boldsymbol{p}_n^*\right). \tag{6}$$

Problem (6) can be straightforward resolved by recently-announced techniques based on the Stochastic Gradient Descent (SGD) method [1], [4]. That is, for generating the training set in (6), the generic WSEE maximisation problem (3) should be reformulated by means of normalising its maximum transmit power coefficients $\{p_i\}$, which requires to transform the variable p_i into $p_i \rightarrow \tilde{p}_i \cdot P_i$, for all $i = 1, ..., L$ for constraining these power coefficients to lie within the closed interval $[0, 1]$. Using the variable transformation $\tilde{p}_i \cdot P_i$, the WSEE expression in (2) becomes $\widetilde{EE}_i(\tilde{p}_i) = \frac{\tilde{R}_i}{\mu_i \cdot \tilde{p}_i + P_{c,i}}$ and the generic WSEE problem (3) can be rewritten into its equivalent form (i.e. suitable for applying SGD) as

$$\max_{\tilde{p}} \textstyle\sum_{i=1}^{L} w_i \frac{B \cdot log\left(1 + \frac{\tilde{\alpha}_i \cdot \tilde{p}_i}{1 + \sum_{j \neq i} \tilde{\beta}_{i,j} \cdot \tilde{p}_j}\right)}{\mu_i \cdot P_i \cdot \tilde{p}_i + P_{c,i}} \tag{7}$$
$$\textit{subject to}: \; 0 \leq \tilde{p}_i \leq 1, \forall i = 1, 2, \ldots, L,$$

where $\tilde{\alpha}_i = \alpha_i \cdot P_i$, $\tilde{\beta}_i = \beta_i \cdot P_i$, for all $i = 1, ..., L$. In problem (7), the normalized ANN training set is $S_T = \{(\tilde{\mathbf{a}}, \tilde{\mathbf{p}}_n^*) | n = 1, ..., N_T\}$ with the parameter vector $\tilde{\mathbf{a}} = (\tilde{\alpha}_i, \tilde{\beta}_{i,j}; P_i)_{i;j}$ to be able to conveniently resolved using the improved branch-and-bound method proposed in [4]. After the training phase, the resulting ANN input-output relationship provides an approximation of the optimal power allocation policy, that can be written in closed-form by considering the composition of the affine combinations and activation functions of all neurons. As shown in [1], [4], such approach provides accurate results with a limited complexity, which makes NP-hard problem optimisation (like the ANN-driven WSEE optimization) applicable in close-to-real-time, i.e., far faster and reliably than conventional relaxations discussed in previous Section II.

IV. DEEP QUANTUM NEURAL NETWORK APPROACH FOR THE ENERGY EFFICIENCY PROBLEM

Notably, upon embedding the proposed ANN approach into a QNN quantum formalism, it is anticipated to resolve the power allocations of the generic WSEE problem (3) much more reliably and fast, i.e., in actual-real-time. This is because upon realising the ANN training process as a variational quantum circuit built on a continuous variable architecture, the training process can be enabled to encode quantum information in continuous degrees of freedom, i.e., by exploiting the amplitudes of the propagation channels $\mathbf{h}_{m_i}$ and $\mathbf{h}_{m_{ij}}$. That means the feed-forward neural network can switch its building blocks from classical- to quantum-neurons, which, in turn, enterprise the computations of the power allocations by considering not only the system parameters of their own k layer, but on all the $K+1$ layers. Such feature is known as Universal Quantum Computation and enables fast optimization with minimum memory demands because the number of the required qubits depends only on the width of the network at hand.

In this respect, we model the QNN training as a quantum circuit of multiple quantum perceptrons, where each perceptron is deemed as a general unitary operator acting on m input qubits and n output qubits. That means, each perceptron can (i) be built using L hidden layers of qubits, (ii) act on an initial input state ρ^{in} of qubits, and (iii) produce an output state ρ^{out} of qubits that can be represented by

$$\rho^{out} \equiv tr_{in,hidden}\left(\mathcal{U}\left(\rho^{in} \otimes |0\cdots 0\rangle_{out,hidden}\langle 0\cdots 0|\right)\mathcal{U}^\dagger\right), \tag{8}$$

where $\mathcal{U} = U^{out} U^L U^{L-1} \cdots U^1$ is the QNN quantum circuit and $\mathcal{U}^\dagger$ the layer-specific perceptron expressed as the unitary operator of the product of the qubits between adjusting layers, e.g., perceptron U^l reflects to the qubits between layer-$(l-1)$ and layer-l, with $l = 1, ..., L$. Note that as perceptrons are arbitrary unitary operators, they may not commute within the QNN, e.g., as shown in Fig.1, meaning that the order of their operations can be defined as the composition of a sequence of completely positive layer-to-layer ε^l transition maps, i.e.,[2]

$$\rho^{out} = \varepsilon^{out}\left(\varepsilon^L\left(\ldots\ \varepsilon^2\left(\varepsilon^1\left(\rho^{in}\right)\right)\ldots\right)\right), \text{ with} \tag{9}$$
$$\varepsilon^l\left(X^{l-1}\right) \equiv tr_{l-1}\left(\textstyle\prod_{j=m_l}^{1} \mathcal{U}_j^l \left(X^{l-1} \otimes |0\cdots 0\rangle_l\langle 0\cdots 0|\right) \prod_{j=1}^{m_l} \mathcal{U}_j^{l\dagger}\right).$$

For example, $\mathcal{U}_j^l$ in (9) is the j-th perceptron acting on previous layers $l-1$ and the current layer l, while m_l is the total number of perceptrons acting on layers $l-1$ and l. On this basis, the QNN mapping in (9) facilitates the information

Algorithm 1 Pseudo-code of the proposed quantum deep-learning process.

1 **Initialization**:
2 Initialize Unitary $\mathcal{U}_j^l$ randomly $\forall j, l$
3 **Feed-forward**:
4 For each batch of $(|\psi_x^{in}\rangle, |\psi_x^{out}\rangle), \forall x \in Batchsize$, $l \in L$
5 (i) Apply the channel ε^l to the output state of layer $l-1$
6 (ii) Tensor ρ_x^{l-1} with layer l in state $|0\cdots 0\rangle_l$ and apply $\mathcal{U}^l = \mathcal{U}_{m_l}^l \cdots \mathcal{U}_1^l$:
7 (iii) Trace out layer $l-1$ and store ρ_x^l
8 **Update Network**:
9 (i) Compute the parameter matrices K_j^l
10 (ii) Update each unitary $\mathcal{U}_j^l$ in which $\mathcal{U} \rightarrow e^{i\epsilon K}\mathcal{U}$
11 **Repeat**:
12 Go to step-3 until the training function reaches its maximum.

data to propagate from the input to the output, which naturally works as a feed-forward neural network, thus, it corresponds to the classical ANN mapping in (4) of the generic WSEE problem (3). More importantly, adopting the mapping in (9), the QNN training can be performed in similar algorithmic logic as in the classical ANN system based on the mapping in (4), i.e., by following a quantum analogue of the, so-called, *back-propagation algorithm* [9]. Recall that back-propagation aims at facilitating the training phase to obtain the output density matrix ρ_x^{out} as closer to the desired output $|\psi_x^{out}\rangle$, which corresponds to the input $|\psi_x^{in}\rangle$. As such, we integrate back-propagation of (9) by setting the training function

$$C = \frac{1}{N} \cdot \sum_{x=1}^{N} \langle \psi_x^{out} | \rho_x^{out} | \psi_x^{out} \rangle, \tag{10}$$

which is known in the field of quantum information theory as *the fidelity function* [9]. Particularly, function C in (10) helps to (i) estimate the perceptrons as quantum states averaged over the training data, and (ii) quantify the difference between the ρ^{out} output and a certain (desired) output by varying between 0 (worst estimation) and 1 (best estimation). For example, C can be minimized by updating the perceptrons unitary operators in the QNN training (9), i.e., $U \rightarrow e^{i\epsilon \mathbf{K}} U$, where $\mathbf{K}$ is the matrix with all system parameters of the corresponding perceptron unitary operator and ϵ the step size of the SGD algorithm. In view of (10), we can define the QNN learning rate η and the update matrices $\mathbf{K}_j^l$ for each perceptron j at layer l as

$$\mathbf{K}_j^l = \eta \cdot \frac{2^{m_{l-1}}}{N} \cdot \sum_{x=1}^{N} tr_{rest} M_j^l, \quad \text{with} \tag{11}$$

$$M_j^l = \left[\prod_{i=j}^{1} \mathcal{U}_i^l \left(\rho_x^{l-1,l} \right) \prod_{i=1}^{j} \mathcal{U}_i^{l\dagger}, \prod_{i=j+1}^{m_l} \mathcal{U}_i^{l\dagger} \left(\mathbb{I}_{l-1} \otimes \sigma_x^l \right) \prod_{i=m_l}^{j+1} \mathcal{U}_i^l \right],$$

$$\rho_x^{l-1,l} = \rho_x^{l-1} \otimes |0\cdots 0\rangle_l \langle 0\cdots 0|,$$

$$\sigma_x^l = \mathcal{F}^{l+1}\left(\ldots \mathcal{F}^{out}\left(|\psi_x^{out}\rangle\langle\psi_x^{out}| \right) \ldots \right)$$

and $\mathcal{F}^l$ the join channel to ε^l, which is the transition channel from layer $l+1$ to layer l, e.g., as illustrated in Fig.2. Relying on (10)-(11) and the feed-forward rationale of the SGD algorithm, we present in Algorithm 1 the pseudo-code to structurize the QNN deep-learning that derives the optimal power allocations of the WSEE problem (3). Notice the key contribution of Algorithm 1 that computes the trace over all qubits (i.e. in line-9) independently from $\mathcal{U}_j^l$, which gives rise to two important features for QNN operation. First, recalling that one of the merits of SDG rationale is that the parameter matrices K can be computed layer-by-layer without any need to rely on additional unitary operators of the full quantum system [6], Algorithm 1 has smaller step size ϵ than the ANN training in (6), which speeds up QNN convergence towards deriving the optimal WSEE power allocations. Second, by coordinating (10) with SGD, the computation of (9) at each step of Algorithm 1 involves only two layers meaning that the size of the K matrices in the calculation of (10) depends only on the width of the network, rather than the width of each layer as required by the respective ANN training function in (5). This latter feature results in considerable memory savings and makes the proposed QNN deep-learning process highly practicable to apply in realistic system setting and scaling. To demonstrate the practicability of our new QNN approach, ext Section V evaluates Algorithm 1 using numerical simulations.

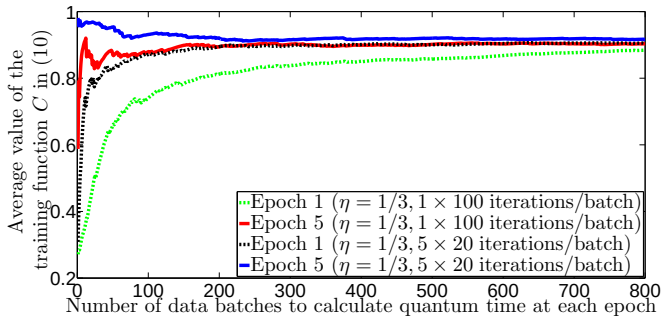

Fig. 3: Average value of the training function versus the number of data batches to calculate the quantum time for each epoch

V. NUMERICAL EVALUATIONS

To simulate the QNN deep-learning process in (11) towards optimizing the WSEE power allocation problem (3), we use a classical desktop computer with typical processing capacity (i.e. Intel i7 with 16 GB of RAM), which, in principle, is not designed to handle more than a handful of qubits. That means, with mainstream equipment we can execute Algorithm 1 for a four-layer QNN system of 16 neurons in total (i.e. 4 neurons for the input layer, 8 neurons for the two hidden layers and 4 neurons for the output layer), while the addition of more neurons is subject to future quantum-based equipment. Yet, even under conventional equipment, we will demonstrate that such four-layer QNN system can attain much higher performance than ANN systems of bigger size, i.e., consisted by seven layers and 28 neurons in total. Also, for the fair simulation comparisons, we deliberate over a realistic system setting similar to [4], which provides a generic quantum Matlab coding of the, so called, globally optimal energy efficient power control framework in interference networks.

On this basis, we translated the provided by [4] Matlab coding (available online at [10]) to Python coding for producing a *training module* composed of 102,000 training samples for P_{max} = -30,...,20 dB in 1 dB step and 10,200 validation samples, i.e., the 10% of the overall training samples. In the produced training module, we split the total number of iterations N_T into R rounds, so as, the parameter matrices $\mathbf{K}_j^l$ in (11) can be re-initialized at the beginning of each feed-forward round of execution. Such matrix re-initialization ensures that the QNN training function C in (10) can increase with smaller step size (i.e. more rapidly) than the ANN training function $\mathcal{L}(\boldsymbol{p_n}(\boldsymbol{\gamma}, \boldsymbol{\delta}), \boldsymbol{p_n^*})$ in (6) because each matrix K_j^l includes all the necessary unitary operators of the full quantum system, rather than additional unitary operators corresponding to each perceptron as done in ANN. Having defined the rationale of our system setting, we proceed by updating the perceptron unitary using $\mathbf{U} \rightarrow e^{i\epsilon \mathbf{K}} \cdot \mathbf{U}$ and performing a random re-initialization and reshuffling with each batch size of 128 samples of data. We arrange [5,4,3,2] neurons over the

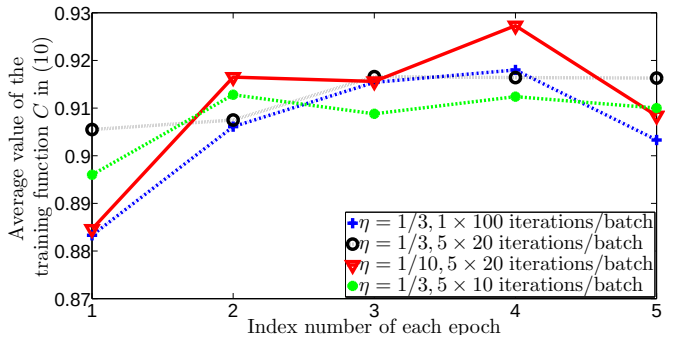

Fig. 4: Average value of the training function versus the index number of each epoch.

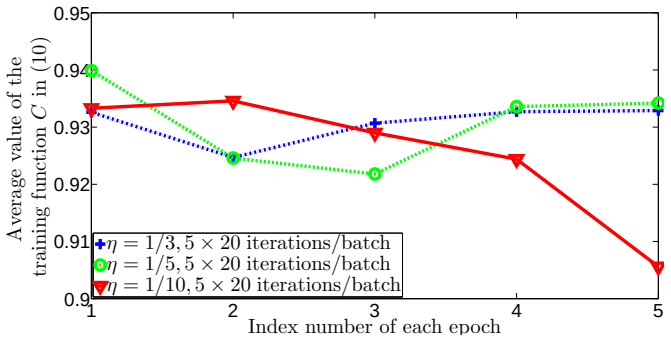

Fig. 5: Average value of the training function versus the index number of each epoch using training perceptron unitary operator.

considered four-layer QNN architecture, which consequences to 5 input perceptrons with 32 states and 2 output perceptrons with 4 states. Also, we arrange the original information data over 17 sets of input data, and set zeros for the rest of the input state, so as, there is no need to add offset data for the output perceptrons as the original output set is exactly equal to 4. Notice that this evaluation architecture is equivalent to the four-layer classical training process architecture with [32,16,8,4] neurons, which, without loss of generality, converges in $5 \times 800 \times 100 = 400,000$ iterations.

In Fig.3 we investigate the role of the parameter matrices $\mathbf{K}_j^l$ in the QNN training process. Particularly, we set $N_T = 100$, learning rate $\eta = 1/3$, step size $\epsilon = 0.1$ and 100 iterations to study the effect of $\mathbf{K}_j^l$ on the value of the training function when it is initialised once and multiple times per step. We observe that when $\mathbf{K}_j^l$ is initialised, Algorithms 1 can always produce lower cost values than those with 5 re-initializations across the number of batches, either for the first epoch or the fifth epoch. We also observe that by initializing multiple matrices $\mathbf{K}_j^l$, Algorithm 1 can attain its maximum cost value much faster than initializing a single matrix for the whole training session. As result, the training function converges towards a steady state (optimal power allocation result) after 300 batches of data for multiple re-initializations.

In Fig.4, we examine the impact of the learning rate for the wireless communications data set. From the figure we see that the learning rate $\eta = 1/10$ is too small to get stable unitary matrices $\mathbf{U}$ compared to $\eta = 1/3$ after 3 training epochs. We also see that by re-initializing multiple K matrices for 50 and 100 iterations within the training module, we can attain stable results (i.e. the WSEE optimal powers) after 3 epochs only (considerably fast). However, we notice that with smaller learning rate there are chances to attain higher cost value at the price of increasing the epoch time, which may consume a longer total training time.

Finally, in Fig 5. we extend the globally optimal power control framework of [6] by adding training perceptron unitary operator to test the reliability of the proposed QNN architecture. For this experiment we use 10,200 samples of new data for validation to observe that the trained perceptron unitary operator with learning rate of either $\eta = 1/3$ or $1/5$ can provide better and stabler cost values after 4 epochs. We also notice from Fig.5 that the learning rate $\eta = 1/10$ is too small for Algorithm 1 for attaining a stable state in a short time, which matches with our previous observations in Fig.3 and implies that there is an optimal value of η to be chosen during the training process, so as, Algorithm 1 can reach its actual minimum time of convergence. Nevertheless, even with small η, the results of Fig.5 show that the training cost value and the validation cost value are almost comparable, which confirms the reliability of the proposed QNN architecture.

VI. CONCLUSION

This paper studied the perspective of adapting the quantum logic in wireless communication networks by proposing unique quantum neural network architecture for resolving the power control in energy-efficiency optimization problems. We formulated the generic energy-efficiency problem and show that its NP-hard complexity can be significantly relaxed with the use of quantum neural network setting rather than using traditional artificial neuron network approach. We further showed that the quantum-oriented algorithm can be not only faster, but also more reliable towards converging to stable (optimal) states. The work concludes that by fine-tuning the learning rate and training epoch time, quantum-oriented algorithms can be further enhanced in terms of stability and speed.

ACKNOWLEDGEMENT

The authors would like to thank Robert Salzmann at the Centre for Quantum Information and Foundations, University of Cambridge for his technical assistance.

REFERENCES

[1] A. Zappone, M. Di Renzo, and M. Debbah, "Wireless networks design in the era of deep learning: Model-based, AI-based, or both?", *IEEE Trans. Commun.*, vol 67, no. 10, pp. 7331 - 7376, Oct 2019

[2] A. Zappone, M. Di Renzo, and M. Debbah, T. T. Lam, and X. Qian, "Model-aided wireless artificial intelligence: Embedding expert knowledge in deep neural networks towards wireless systems optimization", *IEEE Vehic. Technol. Magazine*, vol. 14, no. 3, pp. 60-69, Mar 2019

[3] M. Leshno, V. Y. Lin and S. Schocken, "Multilayer Feed-Forward Networks with a Non-polynomial Activation Function can Approximate any Function", *Neural Networks*, vol. 6, no. 6, pp. 861-867, 1993

[4] B. Matthiesen, A. Zappone, K. L. Besser, E. A. Jorswieck and M. Debbah, "A Globally Optimal Energy-Efficient Power Control Framework and its Efficient Implementation in Wireless Interference Networks", 2019 [Online]: arXiv:1812.06920

[5] J. Allcock, C. Hsieh, I. Kerenidis and S. Zhang, "Quantum Algorithms for Feed-Forward Neural Networks", 2019 [Online]: arXiv:1812.03089

[6] K. Beer, D. Boundarenko, T. Farrelly, T. J. Osbotne, R.t Salzmann, and R. Wolf, "Efficient Learning for Deep Quantum Neural Networks", 2019 [Online]: arXiv:1902.10445

[7] L. Gyongyosi and S. Imre, "Training Optimization for Gate-Model Quantum Neural Networks", *J. Nature Res.*, Sci Rep 9, no. 12679, Sep 2019

[8] C. C. Zarakovitis, Q. Ni and J. Spiliotis, "New Energy Efficiency Metric With Imperfect Channel Considerations for OFDMA Systems," *IEEE Wireless Commun. Letters*, vol. 3, no. 5, pp. 473-476, Oct 2014

[9] M. A. Nielsen and I. L. Chuang , "Quantum Computation and Quantum Information", Cambridge Univ. Press, 2010

[10] Generic Matlab coding for simulating quantum deep learning processes [Online]: https://github.com/R8monaW/DeepQNN