Machine Learning manuscript No.
(will be inserted by the editor)

Hyperparameter Importance and Optimization of Quantum Neural Networks Across Small Datasets

Charles Moussa · Yash J. Patel · Vedran Dunjko · Thomas Bäck · Jan N. van Rijn ·

the date of receipt and acceptance should be inserted later

Abstract As restricted quantum computers become available, research focuses on finding meaningful applications. For example, in quantum machine learning, a special type of quantum circuit called a quantum neural network is one of the most investigated approaches. However, we know little about suitable circuit architectures or important model hyperparameters for a given task. In this work, we apply the functional ANOVA framework to the quantum neural network architectures to analyze which of the quantum machine learning hyperparameters are most influential for their predictive performance. We restrict our study to 7 open-source datasets from the OpenML-CC18 classification benchmark, which are small enough for simulations on quantum hardware with fewer than 20 qubits. Using this framework, three main levels of importance were identified, confirming expected patterns and revealing new insights. For instance, the learning rate is identified as the most important hyperparameter on all datasets, whereas the particular choice of entangling gates used is found to be the least important on all except for one dataset. In addition to identifying the relevant hyperparameters, for each of them, we also learned data-driven priors based on values that perform well on previously seen datasets, which can then be used to steer hyperparameter optimization processes. We utilize these priors in the hyperparameter optimization method hyperband and show that these improve performance against uniform sampling across all datasets by, on average, 0.53%, up to 6.11%, in cross-validation accuracy. We also demonstrate that such improvements hold on average regardless of the configuration hyperband is run with. Our work introduces new methodologies for studying quantum machine learning models toward quantum model selection in practice. All research code is made publicly available.

Keywords Hyperparameter importance · Quantum Neural Networks · Quantum Machine Learning · Hyperparameter Optimization.

C. Moussa, Y. J. Patel, V. Dunjko, T. Bäck, J. N. van Rijn
LIACS, Leiden University, Niels Bohrweg 1, 2333 CA Leiden, Netherlands
E-mail: c.moussa@liacs.leidenuniv.nl

1 Introduction

Quantum computers have the potential to efficiently solve computational problems believed to be intractable for classical computers, such as factoring [66] or simulating quantum systems [18]. In the current noisy intermediate-scale quantum era [55], fault-tolerant quantum algorithms face many limitations (e.g., the number of qubits, decoherence, etc.). Consequently, hybrid quantum-classical algorithms were introduced to target practical applications under these constraints, such as chemistry [45], combinatorial optimization [16], and machine learning [2]. In exceptional cases, quantum models can exhibit exponential advantages over classical ones [30,38,59,70]. More theoretical works also study these models from a generalization perspective [11]. Quantum neural networks are quantum circuits with adjustable parameters that have been used to tackle regression [43], classification [23], generative adversarial learning [76], and reinforcement learning tasks [30, 68].

However, the value of such quantum models is still to be delved into in any larger-scale methodical fashion and on real-world datasets [22,54]. Presently, standard practices from machine learning, such as large-scale benchmarking, hyperparameter importance, and analysis, have been challenging tools to use in the quantum community [63]. Numerous ways to design quantum circuits for machine learning tasks call for hyperparameter optimization and other methods from the field of automated machine learning [28]. However, more intuition is needed on which quantum machine learning hyperparameters are important to optimize and which are less important for efficient hyperparameter optimization [8,17,44].

In order to broach this question, we employ functional ANOVA [26,69], a statistical framework which can be used for assessing hyperparameter importance. To obtain more general results, we follow and extend the methodologies provided by [56, 65], who employed functional ANOVA across datasets. Our work distinguishes itself from the previous work by applying it to the challenging case of simulated quantum neural networks, which come at a significantly increased computational cost compared to the conventional models.

We selected a subset of several low-dimensional datasets from the OpenML-CC18 benchmark [7] matching the current scale of simulations of quantum hardware. We defined a configuration space consisting of ten hyperparameters identified based on a literature study. We then apply the functional ANOVA framework across datasets and extend it with a critical verification step of the internal surrogate models. This results in a ranking of the importance of hyperparameters across datasets. We also perform an extensive experiment to verify whether the importance ranking of hyperparameters holds in practice. Our main findings align with existing knowledge and reveal new insights. For instance, setting the learning rate well is found to be the most critical hyperparameter, whereas the particular choice of entangling gates used is identified as the least important on all except one dataset.

Finally, we demonstrate the usefulness of our insights on hyperparameter importance within a hyperparameter optimization procedure. Following the methodology used in [56], we learned data-driven priors based on values that achieve good performance on previously seen datasets for each hyperparameter. We utilize these priors in the hyperparameter optimization method hyperband, and compare it to the original version of hyperband, which samples hyperparameter configurations uniformly across the input space. We show that the data-driven priors improve

performance by 0.53%, up to 6.11%. Extending from [56], we also demonstrate that such improvements hold on average regardless of the configuration of hyperband. Indeed, across various instantiations of hyperband considered in our study, we improve over the original version of hyperband in 77.2% of cases when sampling according to the data-driven priors.

This paper is an invited extended edition to the original version [48], including the additional methodology and experiments on the data-driven priors for quantum neural networks. We make all of our results and experimental scripts publicly available.[1]

2 Background

This section introduces the necessary background on functional ANOVA, quantum computing, and quantum circuits with adjustable parameters for supervised learning.

2.1 Functional ANOVA

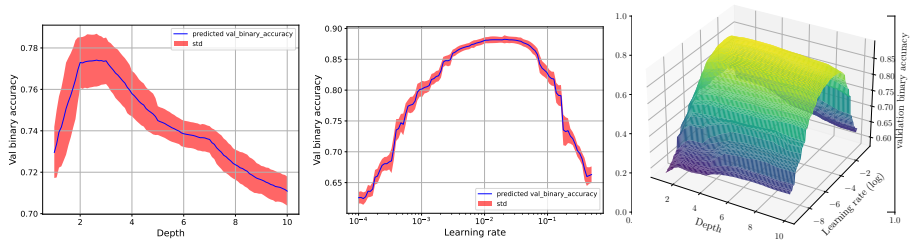
When applying a new machine learning algorithm to solve a specific task, it is not known a priori which hyperparameters to optimize, what are the good ranges for these hyperparameters to sample from, and which values in these user-defined ranges are suitable to get high performance. Several techniques exist that assess hyperparameter importance, such as forward selection [27], ablation analysis [4], local parameter importance [6], and functional ANOVA [26,60]. These techniques are typically used as a post-hoc procedure, stating which hyperparameters were most influential after executing the hyperparameter optimization process. The work of [56] showed that these findings could generalize across datasets.

We first introduce the relevant notation, based on the work by Hutter *et al.* [26].

Let A be a machine learning algorithm that has n hyperparameters with domains $\Theta_1, \ldots, \Theta_n$ and *configuration space* $\boldsymbol{\Theta} = \Theta_1 \times \ldots \times \Theta_n$. An instantiation of A is a vector $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_n\}$ with $\theta_i \in \Theta_i$, for all $i \in [n] = \{1, \ldots, n\}$ (this is also called a *configuration* of A). A partial instantiation of A is a vector $\boldsymbol{\theta}_U = \{\theta_{i_1}, \ldots, \theta_{i_k}\}$ with a subset $U = \{i_1, \ldots, i_k\} \subseteq N = [n]$ of the hyperparameters fixed, and the values for other hyperparameters unspecified. Note that $\boldsymbol{\theta}_N = \boldsymbol{\theta}$.

Functional ANOVA is based on the concept of a marginal of a hyperparameter, i.e., how a given value for a hyperparameter performs, averaging over all possible combinations of the other hyperparameters' values. The *marginal performance* $\hat{a}_U(\boldsymbol{\theta}_U)$ is described as the average performance of all complete instantiations $\boldsymbol{\theta}$ that have the same values for hyperparameters that are in $\boldsymbol{\theta}_U$. As an illustration, Fig. 1 shows marginals for two hyperparameters of a quantum neural network and their union (see also Fig. 9 in Appendix A). As the number of terms to consider for the marginal can be very large, the authors of [26] used tree-based surrogate regression models to calculate the average performance efficiently. Surrogate models operate on the meta-level: they take as input the hyperparameter values of a certain configuration, and map this to a given performance score of this

[1] See: `https://github.com/chMoussa/prior_qnn_surrogate_search`.



(a) (b) (c)

Fig. 1: Marginals for a quantum neural network with validation accuracy as performance on the banknote-authentication dataset. Marginal plots of other datasets are shown in Fig. 9 in Appendix A. The hyperparameters correspond to the number of layers, also known as depth (a), the learning rate used during training (b), and their combination (c). The hyperparameter values for the learning rate are on a log scale. When considered individually, we see, for instance, that depth and learning rate should be set at a reasonable price for better performances. However, when grouped together, the learning rate seems most influential.

configuration [14]. As such, it can be used to assess the scores of configurations that we have not observed directly, serving as a surrogate for running the actual model and determining its performance. Such a model yields predictions $\hat{y}$ for the performance measure p of arbitrary hyperparameter settings.

Functional ANOVA determines how much each hyperparameter (and each combination of hyperparameters) contributes to the variance of $\hat{y}$ across the algorithm's hyperparameter space $\boldsymbol{\Theta}$, denoted $\mathbb{V}$. Intuitively, it assumes that a hyperparameter is highly important to the performance measure if its marginal has a high variance and vice versa. Due to the marginalizing over all possible hyperparameter values and combinations, it gives a global overview of the important hyperparameters (opposed to for example ablation analysis, which gives a more local view) [5]. Functional ANOVA has been used for studying the importance of hyperparameters of standard machine learning models such as support vector machines, random forests, Adaboost, and residual neural networks [56,65]. We refer to [26] for a complete description.

2.2 Supervised learning with Parameterized Quantum Circuits

2.2.1 Basics of quantum computing

In quantum computing, computations are performed by manipulating qubits, similar to classical computing with bits. A system of n qubits is represented by a 2^n-dimensional complex vector in the Hilbert space $\mathcal{H} = (\mathbb{C}^2)^{\otimes n}$. This complex vector describes the state of the system $|\psi\rangle \in \mathcal{H}$ and is of unit norm $\langle\psi|\psi\rangle = 1$. The bra-ket notation is used to describe vectors $|\psi\rangle$, their conjugate transpose $\langle\psi|$ and inner-products $\langle\psi|\psi'\rangle$ in $\mathcal{H}$. Single-qubit computational basis states are given by $|0\rangle = (1,0)^T, |1\rangle = (0,1)^T$, and their tensor products describe general computational basis states, e.g., $|10\rangle = |1\rangle \otimes |0\rangle = (0,0,1,0)$.

The quantum state is modified with unitary operations or gates U acting on $\mathcal{H}$. This computation can be represented by a quantum circuit (see Fig. 2). When a gate U acts non-trivially only on a subset $S \subseteq [n]$ of qubits, we denote such operation $U \otimes \mathbb{1}_{[n]\setminus S}$. In this work, we use, the Hadamard gate H, the single-qubit Pauli gates X, Y, Z and their associated rotation gates R_X, R_Y, R_Z:

$$\begin{aligned} H = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}, Z = \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}, R_Z(w) = \exp\Big(-i\frac{w}{2}Z\Big),\\ Y = \begin{pmatrix}0 & -i\\ i & 0\end{pmatrix}, R_Y(w) = \exp\Big(-i\frac{w}{2}Y\Big), X = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}, R_X(w) = \exp\Big(-i\frac{w}{2}X\Big), \end{aligned} \tag{1}$$

The rotation angles are denoted by $w \in \mathbb{R}$ (typically restricted in the range of $[-\pi, \pi]$). The matrix form of a 2-qubit controlled-Z gate ⁞ = $\mathrm{diag}(1, 1, 1, -1)$ and the $\sqrt{\mathrm{iSWAP}}$ (also denoted as sqiswap) gate is given by

$$\frac{1}{\sqrt{2}}\begin{pmatrix}\sqrt{2} & 0 & 0 & 0\\ 0 & 1 & i & 0\\ 0 & i & 1 & 0\\ 0 & 0 & 0 & \sqrt{2}\end{pmatrix}. \tag{2}$$

Measurements are carried out at the end of a quantum circuit to obtain bitstrings. Such measurement operation is described by a Hermitian operator O called an observable. Its spectral decomposition $O = \sum_m \lambda_m P_m$ in terms of eigenvalues λ_m and orthogonal projections P_m defines the outcomes of this measurement, according to the Born rule: a measured state $|\psi\rangle$ gives the outcome λ_m and gets projected onto the state $P_m |\psi\rangle / \sqrt{p(m)}$ with probability $p(m) = \langle\psi| P_m |\psi\rangle = \langle P_m\rangle_\psi$. The expectation value of the observable O with respect to $|\psi\rangle$ is $\mathbb{E}_\psi[O] = \sum_m p(m)\lambda_m = \langle O\rangle_\psi$. We refer to [51] for more basic concepts of quantum computing and follow with parameterized quantum circuits.

2.2.2 Parameterized Quantum Circuits

A parameterized quantum circuit (also called *ansatz*) can be represented by a quantum circuit with adjustable real-valued parameters $\mathbf{w}$. A unitary $U(\mathbf{w})$ then defines the quantum circuit by acting on a fixed n-qubit state (e.g., $|0^{\otimes n}\rangle$). The ansatz may be constructed by exploiting the nature of the problem (typically the case in chemistry [45] or optimization [16]) or with a problem-independent generic construction. The latter are often designated as *hardware-efficient-ansatz*.

For a machine learning task, the unitary $U(\mathbf{w})$ encodes an input data instance $x \in \mathbb{R}^d$ and is parameterized by a trainable vector $\mathbf{w}$. Many designs exist, but hardware-efficient parameterized quantum circuit [31] with an alternating-layered architecture is often considered in quantum machine learning when no information on the structure of the data or the problem is known. This architecture is depicted in an example presented in Fig. 2 and essentially consists of an alternation of encoding unitaries U_{enc} and variational unitaries U_{var}. In the example, U_{enc} is composed of single-qubit rotations R_X, and U_{var} of single-qubit rotations R_Y, R_Z and entangling Ctrl-Z gates, represented as ⁞ in Fig. 2, forming the entangling part of the circuit. Such an entangling part denoted U_{ent} can be defined by connectivity between qubits.

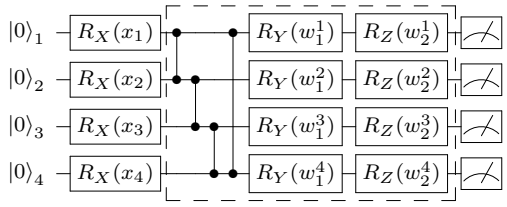


Fig. 2: Parameterized quantum circuit architecture example with 4 qubits and ring connectivity (qubit 1 is connected to 2, 2 to 3, 3 to 4, and 4 to 1 makes a ring). The first layer of R_X is the encoding layer U_{enc}, taking a data instance $x \in \mathbb{R}^4$ as input. It is followed by the entangling part with Ctrl-Z gates. Finally, a variational layer U_{var} is applied. Eventually, we do measurements to be converted into predictions for a supervised task. The dashed part can be repeated many times to increase the expressive power of the model.

These parameterized quantum circuits are similar to neural networks where the circuit architecture is fixed, and the gate parameters are optimized by a classical optimizer such as gradient descent. Hence, they have also been termed quantum neural networks. The parameterized layer can be repeated multiple times, which increases its *expressive power* like neural networks [67]. The data encoding strategy (such as reusing the encoding layer multiple times in the circuit - a strategy called *data reuploading*) also influences the latter [52, 64].

The user can define the observable(s) and the post-processing method to convert the circuit outputs into a prediction in the case of supervised learning. In practice, observables based on the single-qubit Z operator are used. When applied on $m \leq n$ qubits, the observable is represented by a $2^m - 1$ square diagonal matrix with $\{-1, 1\}$ values and is denoted $\mathcal{O} = Z \otimes Z \otimes \cdots \otimes Z$.

Having introduced parameterized quantum circuits, we present the hyperparameters of the models, the configuration space, and the experimental setup for our functional ANOVA-based hyperparameter importance study.

2.2.3 Related works

To apply functional ANOVA for performing hyperparameter importance, we performed an extensive literature review on parameterized quantum circuits for machine learning [2, 23, 24, 29, 30, 37, 39, 42, 43, 54, 62, 68, 71, 72, 73, 76] as well as quantum machine learning software [1, 3, 10]. This resulted in a set of hyperparameters and configuration space presented in Section 3.1. Several works also study the performances of quantum neural networks on binary classification tasks, but more for benchmarking purposes [40, 62]. Concerning hyperparameter optimization, several directions are taken, from quantum architecture search [13, 75], to developing hyperparameter optimization techniques [58], or applying concepts from automated machine learning to quantum models [19]. In our case, we use insights from the hyperparameter importance study to steer a hyperparameter optimization procedure, which can be considered an automated machine learning concept.

3 Methods

In this section, we describe the model with its hyperparameters and define our methodology.

3.1 Hyperparameters, configuration space and simulations

Many parameterized quantum circuit designs have been introduced based on motivated research questions and contributions or the addressed problem. We first carried out an extensive literature review on parameterized quantum circuits for machine learning [2,23,24,29,30,37,39,42,43,54,62,68,71,72,73,76] as well as quantum machine learning software [1,3,10]. We aggregated and translated the design choices into a set of hyperparameters and configuration space, resulting in a list of 10 hyperparameters, presented in Table 1. We choose them, so we balance between having well-known hyperparameters that are expected to be essential and less considered ones in the literature. For instance, many use Adam [32] as the optimization algorithm whose learning rate should usually be well set. In contrast, the choice of the entangling gate is generally fixed.

From the literature, we expect the data encoding strategy/circuit to be essential. We set two main forms for U_{enc}. The first one is the hardware-efficient $\bigotimes_{i=1}^{n} R_X(x_i)$. The second form from [3,29,23], translates to an instantaneous quantum polynomial (IQP) circuit and is formulated as:

$$U_{\text{enc}}(\boldsymbol{x}) = U_z(\boldsymbol{x}) H^{\otimes n} \tag{3}$$

$$U_z(\boldsymbol{x}) = \exp\left(-\mathrm{i}\pi \left[\sum_{i=1}^{n} x_i Z_i + \sum_{\substack{j=1, \\ j>i}}^{n} x_i x_j Z_i Z_j \right] \right). \tag{4}$$

We can also make a model more expressive using data-reuploading [52,64,30, 68] or by pre-processing the input [64] (sometimes used in encoding strategies where input features are fed into Pauli rotations). In this work, as a possible pre-processing technique, we choose a usual activation function tanh. Its range is $[-1, 1]$, similar to the data features during training after the normalization step.

The list of hyperparameters we take into account is non-exhaustive. Therefore, it can be extended at will, at the cost of more software engineering and a budget for running experiments. All the quantum circuits were implemented using the Cirq library [20] and numerically simulated with the TensorFlow Quantum library [10] using an exact statevector simulator (without quantum noise). An estimated total compute time for all the simulations performed was 31 000 CPU-hours.

3.2 Assessing Hyperparameter Importance

Having set the previous list of hyperparameters and the configuration space, we perform the hyperparameter importance analysis using functional ANOVA. Firstly, we sample various random configurations per dataset and apply the models to measure their performance according to a measure we are interested in, in this

Table 1: List of hyperparameters considered for hyperparameter importance for a quantum neural network, as we named them in our TensorFlow Quantum code.

Hyperparameter	Values	Description
Adam learning rate	$[10^{-4}, 0.5]$ (log)	The learning rate with which the quantum neural network starts training. The range was taken from the automated machine learning library Auto-sklearn [17]. We uniformly sample, taking the logarithmic scale.
batch size	16, 32, 64	Number of samples in one batch of Adam used during training
depth	$\{1, 2, \cdots, 10\}$	Number of variational layers defining the circuit
is data_encoding hardware efficient	True, False	Whether we use the hardware-efficient circuit $\bigotimes_{i=1}^{n} R_X(x_i)$ or an IQP circuit defined in Eq.3 to encode the input data.
use reuploading	True, False	Whether the data encoding layer is used before each variational layer or not.
have less rotations	True, False	If True, only use layers of R_Y, R_Z gates as the variational layer. If False, add a layer of R_X gates.
entangler operation	cz, sqiswap	Which entangling gate to use in U_{ent}
map type	ring, full, pairs	The connectivity used for U_{ent}. The ring connectivity uses an entangling gate between consecutive indices $(i, i+1), i \in \{1, \ldots, n\}$ of qubits. The full one uses a gate between each pair of indices $(i, j), i < j$. Pairs connect even consecutive indices first, then odd consecutive ones.
input activation function	linear, tanh	Whether to input $\tanh(x_i)$ as rotations or just x_i.
output circuit	2Z, mZ	The observable(s) used as output(s) of the circuit. If 2Z, we use all possible pairs of qubit indices defining $Z \otimes Z$. If mZ, the tensor product acts on all qubits. Note that we do not use single-qubit Z observables, although they are often used in the literature. Indeed, they are provably not using the entire circuit when it is shallow. Hence we decided to use $Z \otimes Z$ instead. Also, a single neuron layer with a sigmoid activation function is used as a final decision layer similar to [62].

case, predictive accuracy. The sampled configurations and performances are used to train a tree-based surrogate model (i.e., a random forest [9]) that can map any configuration to the performance measure. Each hyperparameter value is given as input, and the output is predictive accuracy. On top of the hyperparameters, we can also add the number of epochs to make the surrogate aware of specific budgets.

Next, we verify the performance of the surrogate models. We utilize regression metrics commonly used in surrogate benchmarks [15] to discard datasets where surrogates perform poorly, as they can deteriorate the quality of the results. Per dataset, the R2 score is calculated over the actual performance per configuration versus the predicted performance per configuration. If this score is too low (below 0.75), the surrogate model is not accurate enough, and the dataset is excluded from further analysis. Finally, we obtain the marginal contribution of each hyperparameter across all datasets, which can be used to infer a ranking of their general

importance. A verification step similar to [56] is carried out to confirm the inferred ranking previously obtained. We explain such a procedure in the following section.

3.3 Verifying Hyperparameter Importance

In order to verify the hyperparameter importance ranking, the authors of [56] proposed a random search procedure in which one hyperparameter is fixed to a given value, and all other hyperparameters are optimized. The assumption is that when an important hyperparameter is fixed to a given value, the result of the optimization procedure is worse than when an unimportant hyperparameter is fixed to a given value. When fixing a hyperparameter to a given value, the procedure is repeated several times with different values to avoid bias, and as such, the optimization procedure is carried out several times. Formally, for each hyperparameter θ_j, we measure $y^*_{j,f}$ as the result of a random search for maximizing the metric, fixing θ_j to a given value $f \in F_j, F_j \subseteq \Theta_j$. For categorical θ_j with domain Θ_j, $F_j = \Theta_j$ is used. For numeric θ_j, following [56], we use a set of 10 values spread uniformly over θ_j's range. We then compute $y^*_j = \frac{1}{|F_j|} \sum_{f \in F_j} y^*_{j,f}$, representing the score when not optimizing hyperparameter θ_j, averaged over fixing θ_j to various values it can take. Hyperparameters with lower values for y^*_j are assumed to be more important since the performance should deteriorate more when set sub-optimally.

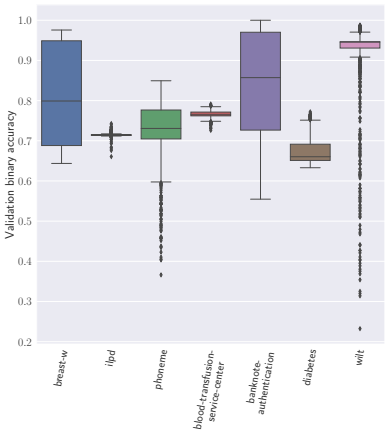


Fig. 3: Performances of 1 000 quantum machine learning models defined by different configurations of hyperparameters over each dataset. The metric of interest we use is the 10-fold cross-validation accuracy. We take the best-achieved metric per model trained over 100 epochs.

In our study, we extend the verification step to be used on the scale of quantum machine learning models. As quantum simulations can be costly, we use the predictions of the surrogate instead of fitting new quantum models during the verification experiment. The surrogates yield predictions $\hat{y}$ for the performance of arbitrary hyperparameter settings sampled during a random search, serving to compute $y^*_{j,f}$. This is also the reason why we performed a second phase evaluation of the built-in surrogates. Surrogates that perform poorly can reduce the quality of built marginals, resulting in inferred conclusions with potential bias.

3.4 Deriving Data-driven Priors for Hyperparameter Optimization

From the functional ANOVA framework, we obtain insights into which hyperparameters are important to optimize and which are not at the hyperparameter level. Additionally, we will explore ways to utilize well-performing hyperparameter values across datasets. The authors of [56] demonstrated a procedure for this that learns

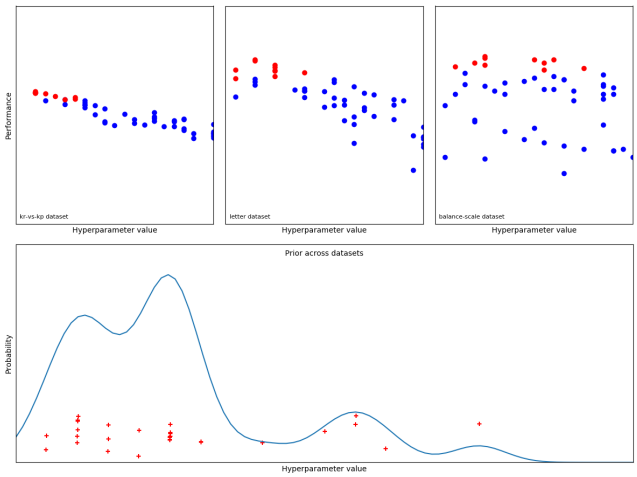


Fig. 4: Procedure outlining how we learn data-driven priors, following [56]. The top pane shows three individual datasets (utilized for training). Each dot here represents a configuration that was run on these datasets, with the red dots being the best-performing configurations. The bottom pane shows how a kernel density estimator can be utilized to learn data-driven priors inferred from the best configurations. This figure shows how this is done for a single hyperparameter, although it can be generalized to multi-dimensional configuration spaces.

data-driven priors across datasets from good hyperparameter values. The priors are then used within a hyperparameter optimization procedure such as hyperband [36].

Figure 4 outlines this procedure. For each dataset, we utilize past experiments and determine which configurations yielded the best performance. For each dataset not used for evaluation, we identify the best-N (N to be determined by the user, but we found that 10 and 20 works) performing configurations in terms of the metric that we are interested in (e.g., predictive accuracy). Per hyperparameter, we now gather the values that ended up in the best configurations. We fit a 1-dimensional distribution per hyperparameter [35]. For numeric hyperparameters, we use a kernel density estimator; for categorical hyperparameters, we sample from a multinomial distribution (or Bernoulli in case of 2 possible values) whose parameters are set according to the frequencies of values. This distribution now represents a learned prior of good hyperparameter values and can be used to sample configurations in a hyperparameter optimization process.

4 Dataset and inclusion criteria

We consider classical datasets to apply the machinery of quantum neural networks and investigate the importance of the hyperparameters that were introduced before. Similarly to [56], we use datasets from the OpenML-CC18 benchmark suite [7]. We adhere to a commonly used trend in the quantum community in our study; i.e., we only consider the datasets whose number of features is equal to the number of qubits available under the current scale of quantum simulations on a server. We limit this study to the case where the number of features is less than 20 after pre-processing,

Table 2: List of datasets used in this study. The number of features is obtained after a usual pre-processing used in machine learning methods, such as one-hot encoding.

Dataset	OpenML Task ID	Number of features	Number of instances
breast-w	15	9	699
diabetes	37	8	768
phoneme	9952	5	5 404
ilpd	9971	11	583
banknote-authentication	10093	4	1 372
blood-transfusion-service-center	10 101	4	748
wilt	146820	5	4 839

as simulating quantum machine learning algorithms is computationally expensive.[2] Hence, our first step was identifying which datasets fit this criterion. We include all datasets from the OpenML-CC18 with 20 or fewer features after categorical features have been one-hot-encoded and constant features are removed. Afterwards, the input variables are scaled to unit variance as a normalization step. Finally, the scaling constants are calculated on the training data and applied to the test data.

The final list of datasets is given in Table 2. In total, 7 datasets fitted the criterion considered in this study. For all of them, we picked the OpenML Task ID giving the 10-fold cross-validation task. A quantum model is then applied using the latter procedure with the aforementioned pre-processing steps.

5 Results of Hyperparameter Importance

In this section, we present the results of the hyperparameter importance study.

5.1 Performance distributions per dataset

We independently sampled 1 000 hyperparameter configurations for each dataset and ran the simulation of quantum models for 100 epochs as budget. We recorded the best validation accuracy obtained over 100 epochs as a performance measure. Fig. 3 shows the distribution of the 10-fold cross-validation accuracy obtained per dataset. We observe the impact of hyperparameter optimization by the difference between the least performing and the best model configuration. For instance, on the wilt dataset, the best-performing model gets an accuracy close to 1, and the least-performing model obtains a performance below 0.25. We can also see that some datasets present a smaller spread of performances. For example, ilpd and blood-transfusion-service-center are in this case. Hyperparameter optimization does not seem to have a real effect because most hyperparameter configurations give the same result. As such, the surrogates could not differentiate between various configurations. For most datasets, hyperparameter optimization is vital for getting high performances per dataset and detecting datasets where the importance study can be applied.

[2] A 10-fold cross-validation run in our experiment takes on average 262 minutes for 100 epochs with Tensorflow Quantum [10].

Table 3: Results of the step in which we validate the surrogate models. This table shows performances of the surrogate models built within functional ANOVA over a 10-fold cross-validation procedure. We present the average coefficient of determination (R2), root mean squared error (RMSE), and Spearman's rank correlation coefficient (CC). These are standard regression metrics for benchmarking surrogate models on hyperparameters [15]. The surrogates over ilpd and blood-transfusion-service-center obtain low scores (less than .75 R2). Hence we remove them from the study.

Dataset	R2 score	RMSE	CC
breast-w	0.8663	0.0436	0.9299
diabetes	0.7839	0.0155	0.8456
phoneme	0.8649	0.0285	0.9282
ilpd	0.1939	0.0040	0.4530
banknote-authentication	0.8579	0.0507	0.9399
blood-transfusion-service-center	0.6104	0.0056	0.8088
wilt	0.7912	0.0515	0.8015

5.2 Surrogate verification

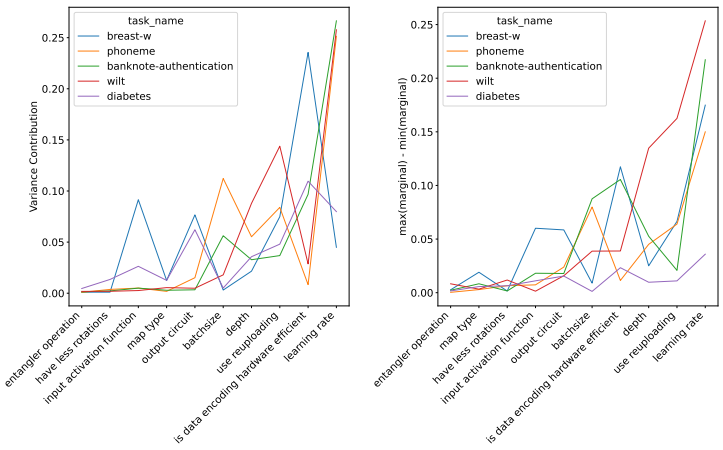
Functional ANOVA relies on an internal surrogate model to determine the marginal contribution per hyperparameter. If this surrogate model is not accurate, this can severely limit the conclusions drawn from functional ANOVA. Surrogate models are trained to map the configuration to a particular performance score. Each hyperparameter value is given as an input, and the output is the performance measure of interest, in this case, predictive accuracy. As such, in this experiment, we measure the performance of the surrogates without information on the number of epochs. In this experiment, we verify whether the hyperparameters can explain the performances of the models. Table 3 shows the performance of the internal surrogate models. We notice low regression scores for the two datasets (less than 0.75 R2 scores). Hence we remove them from the analysis.

5.3 Marginal contributions

For functional ANOVA, we used 128 trees for the surrogate model. Fig. 5(a,b) shows the marginal contribution of each hyperparameter over the remaining 5 datasets. We distinguish visually 3 primary levels of importance. According to these results, the learning rate, depth, data encoding circuit, and reuploading strategy are critical. These results are in line with our expectations. According to functional ANOVA, the entangler gate, connectivity, and whether we use R_X gates in the variational layer are the least important. Hence, our results reveal new insights into these hyperparameters that are not considered in general.

5.4 Verification of important hyperparameters

In line with the work of [56], we perform an additional verification experiment that verifies whether functional ANOVA outcomes align with our expectations. However, the verification procedure involves an expensive, post hoc analysis: a



(a) (b)

Fig. 5: The marginal contributions per dataset are presented as (a) the variance contribution and (b) the difference between the minimal and maximal value of the marginal of each hyperparameter. The hyperparameters are sorted from the least to the most important using the median. We distinguish from the plot 3 primary levels of importance.

random search procedure fixing one hyperparameter at a time. Therefore, as our quantum simulations are costly, we used the surrogate models fitted on the current dataset considered over the 1 000 configurations obtained initially to predict the performances one would obtain when presented with a new configuration. Fig. 6 shows the average rank of each run of random search, labeled with the hyperparameter whose value was fixed to a default value. A high rank implies poor performance compared to the other configurations, meaning that tuning this hyperparameter would have been important. We again witness the 3 levels of importance, with almost the same order obtained. However, the input_activation_function is found to be more important while the batch size is less. More simulations with more datasets may be required to validate the importance. However, we empirically retrieve the importance of well-known hyperparameters while considering less important ones. Hence functional ANOVA becomes an interesting tool for quantum machine learning in practice. Next, we demonstrate this by using the obtained results to improve a hyperparameter optimization algorithm.

5.5 Efficiency of Data-driven Priors for hyperparameter optimization

We will utilize the performance data to build data-driven priors over what are good hyperparameter values across datasets. The data-driven priors can then be utilized in hyperparameter optimization methods, such as hyperband [36], to replace uniform sampling strategies. We will use the experimental data to obtain a set of data-driven priors (kernel density estimators and multinomial) over hyperparameter values, similar to [56]. We will demonstrate that employing the data-driven priors

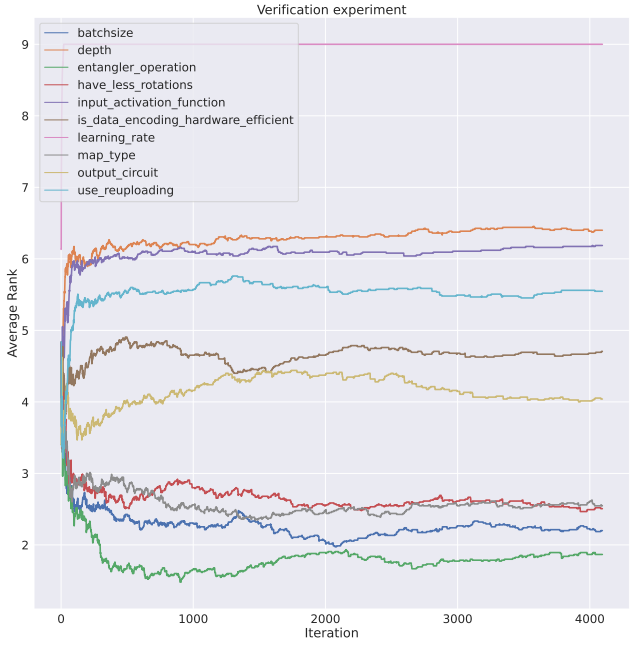


Fig. 6: Verification experiment of the importance of the hyperparameters. A random search procedure is used for up to 4096 iterations, excluding one parameter at a time. A lower curve means the hyperparameter is found to be less important.

improves the results obtained with hyperband with respect to the default uniform sampling.

We will utilize hyperband to verify whether the data-driven priors are more effective than uniform sampling. Generally, in hyperband, a number of hyperparameter configurations are sampled uniformly (where each hyperparameter value is sampled independently) to train machine learning models given a value of a user-defined notion of budget (e.g. the number of epochs). A proportion of the configurations that have performed best are kept for the next iteration where the budget value is increased until one configuration remains. In our case, uniform sampling can be replaced by the data-driven priors which can be sampled to obtain hyperparameter values. In order to get a balanced assessment under which conditions these data-driven priors work well, we also vary the hyperparameters of hyperband itself. While hyperband is arguably robust against ill-specified hyperparameters, we want to see whether the data-driven priors work well across the various options. Table 4 shows the various hyperparameters of hyperband that we considered. We ran an experiment on each combination of these hyperparameters. Note that, to keep the computational cost manageable, we do not run the actual algorithms that hyperband is exploring but use surrogate models instead. The latter are built similarly to the previously considered surrogate models in functional ANOVA (see Sec. 5.2); however, we also incorporate the number of epochs as a feature for this experiment. We do so to use the number of epochs as the notion of budget for hyperband. Configurations are sampled randomly with an increased number of

Table 4: List of hyperparameters considered for hyperband experiments.

Hyperparameter	Values	Description
η	$2, 3, 4$	Elimination rate factor for configurations advancing to the next iteration.
s_max	$3, 4, 5, 6$	Number of unique executions of successive halving.
R	50, 100	Maximum amount of resources (epochs) that can be allocated to a single configuration.
Important Hyperparameters	Adam learning rate, depth, input activation function, use reuploading	Set of important hyperparameters considered to fit a Kernel Density Estimator based on the insights of Section 5.4.
best_N	$10, 20$	Number of best-performing configurations on which a kernel density estimator or multinomial distribution is fitted for important hyperparameters.
Sampling method	Uniform, Data-driven (e.g., kernel density estimator)	The method for generating hyperparameter configurations to run the models on.
Bandwidth Selection method	Silverman, Sheather-Jones	The bandwidth estimator method used to fit the data with kernel density estimator.

epochs for a given hyperband iteration, onto which performances are determined with the surrogate models.

Figure 10 in Appendix B illustrates example kernel density estimators for a quantum neural network's two most important hyperparameters. We see from the figure that: (i) the default values of Adam's learning rate used by the deep learning practitioners have a reasonable default, and (ii) the significance of using a higher depth in parameterized quantum circuits as it makes them more expressive (a desirable property of any machine learning model). We note that in order to evaluate the data-driven priors on a given dataset, we build these priors on all other datasets except this one (leave one dataset out). As such, we have a slightly different set of data-driven priors for each experiment.

For our experiments, we run hyperband with different values for its hyperparameters with 15 random seeds per dataset, using the previously-mentioned surrogate models to obtain performance scores per configuration (to economize on the otherwise extensive runtime). Figure 7 and 8 illustrate our results. For each dataset, the difference in accuracy between the two sampling strategies is depicted: positive values indicate that sampling based on data-driven priors performs better. Each data point represents a given configuration of hyperband with data-driven priors compared to a configuration of hyperband with uniform sampling. The violin plots aggregate over various datasets (5), random seeds (15) and various hyperparameter settings of hyperband (see Table 4).

In general, the results suggest that using data-driven priors from good values of hyperparameters can aid in finding better configurations of quantum neural networks. Indeed, as shown in Figure 7 a), the average improvement is 0.53%,

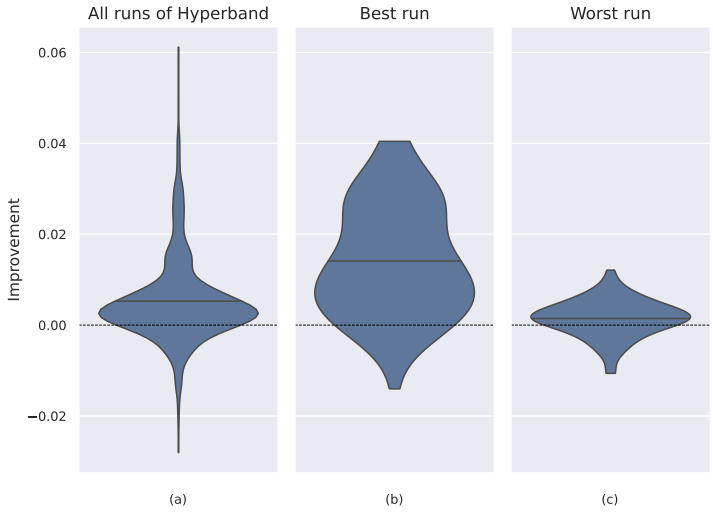


Fig. 7: Relative difference in performance improvement between two instances of hyperband, one sampling based on the learned priors and one using uniform sampling. Positive values indicate superior performance by using data-driven priors and vice versa. We show the distribution of performances over all combinations of hyperband parameters from Table 4, random seeds and datasets (a), and for the data-driven prior hyperband run achieving the largest average performance gain over the uniform priors (averaged over all random seeds and datasets) (b) and for the data-driven prior hyperband run achieving the lowest average performance gain over the uniform priors (averaged over all random seeds and datasets) (c). We note that the data-driven priors are superior to the uniform sampling across all tried configurations of hyperband.

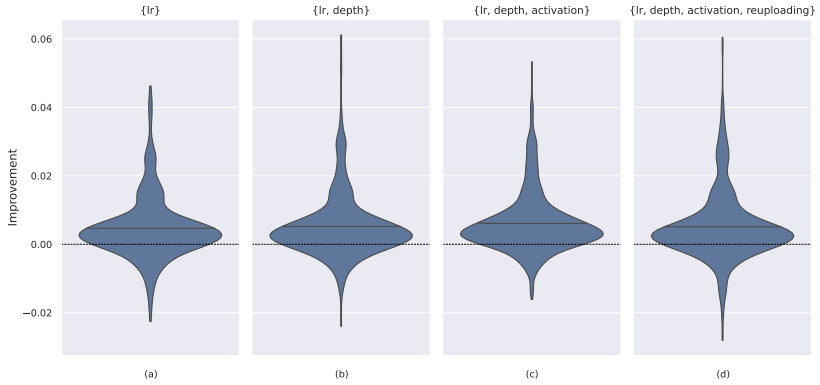


Fig. 8: Relative difference in performance improvement between two instances of hyperband, one sampling based on the data-driven priors and one using uniform sampling. We show the distribution of performances over all combinations of hyperband parameters from Table 4, random seeds, and datasets. The figure from left (a) to right (d) shows a slight increase in average improvement as the number of hyperparameters considered for building data-driven priors grows. Here, lr, depth, activation, and reuploading denote Adam optimizer's learning rate, quantum circuit depth, activation function, and data reuploading.

and the maximum was 6.11% for all hyperband hyperparameters tried. From Figure 7 b), the best run in terms of average performances achieved an average improvement of 1.41%, and the maximum was 4.04%. Moreover, from Figure 7 c), the worst run in terms of average performance achieved an average improvement of 0.14%, and the maximum was 1.21%. Furthermore, the improvement was obtained across all hyperband runs in 77.2% of cases. Finally, Figure 8 shows that the average improvement generally increases as we add a hyperparameter according to its importance ranking when learning data-driven priors. This demonstrates the benefits of balancing between exploration and exploitation using hyperparameter importance when performing hyperparameter optimization.

More simulations with more datasets can be performed as methods enabling the usage of quantum neural networks on datasets with a number of features greater than the number of qubits are being developed [22, 54]. However, we verified empirically that the importance information obtained from functional ANOVA could be helpful for hyperparameter optimization. More sophisticated hyperparameter optimization algorithms can also benefit from such studies.

6 Limitations

We discuss three limitations to better place the work in a context that could guide further research. Firstly, the work heavily leans on the hyperparameter importance definition of functional ANOVA. While this is a well-established measure built on a solid foundation of prior work [69, 26, 56] there are also downsides. The involvement of the marginal has a very desirable property in that it is not conditioned on the value of other hyperparameters, as it averages over all possible values for all other hyperparameters. Following this definition, it considers both very well-performing hyperparameter configurations and all mediocre-performing configurations. Thereby, it gives good information on what hyperparameters are generally important, giving a global picture. However, in some cases, we might be more interested in the hyperparameters that do not seem important on a global level, are responsible for obtaining the final bits of performance. These are usually hard to detect by functional ANOVA. The work of [25] proposes a way to filter out bad-performing configurations, compromising the global overview of functional ANOVA in favour of the aforementioned important details. However, it is essential to note that every definition of hyperparameter importance always comes with a particular definition bias that should be clearly communicated.

Secondly, due to the high amount of required CPU resources, we have decided to perform both the verification experiment and the evaluation of the data-driven priors on learned surrogate models. The surrogate models are learned based on the experimental data we have gathered and evaluated extensively (see Section 5.2). However, like any model, the surrogates are not perfect, and by introducing a surrogate in the experimentation, a bias is induced in favour of economizing on CPU time. Since the experiments of [56] were run on actual models (rather than the surrogate models) and already confirmed the high correlation between the functional ANOVA results and the verification experiment, we believe that using the surrogate models for this paper is a valid approach.

Finally, by means of this study, we have considered a family of quantum neural network architectures, whereas the literature has proposed many more

architectures. While it is likely that some hyperparameter importance results generalize across different architectures (e.g., the learning rate and the depth will likely always be important), it should be noted that only limited conclusions can be transferred across different architectures. Ideally, this study can be repeated over a broad range of quantum neural network architectures, identifying even more general patterns. For instance, various hyperparameters of the models can have an influence on the trainability and optimization of the model. It is known that quantum neural networks can suffer from the phenomenon of barren plateaus [41], similar to the vanishing gradients problem inherent in neural networks, resulting in trainability issues. Several hyperparameters such as the circuit architecture [50], input state [34] and the depth [12] are related to barren plateaus. It would be interesting to extend this study by modifying or adding into the hyperparameters more quantum circuit specifications designed to handle barren plateaus [53, 21, 74, 57] and tailored optimization procedures for quantum machine learning [47, 33].

7 Conclusion

In this work, we assess the importance of several hyperparameters related to quantum neural networks for classification using the functional ANOVA framework. Our experiments rely on OpenML datasets matching the current scale of quantum hardware simulations (i.e., datasets with at most 20 features after applying pre-processing operators, hence using 20 qubits). We selected and presented the hyperparameters based on an investigation of quantum computing literature and software. Firstly, hyperparameter optimization highlighted datasets where we observed high variance in the performance across configurations (see Figure 3). In particular, for the 'wilt' dataset, the performances were spread from 25%–100%. This further underlines the importance of hyperparameter optimization for these datasets. There were also datasets where this variance was negligible.

Following [56], we utilized functional ANOVA to attribute the variance in performance to the various (combinations of) hyperparameters. Hyperparameters that attribute to a high variance are considered important, whereas hyperparameters that attribute to only a small variance are considered unimportant. From our results, we distinguished 3 primary levels of importance. On the one hand, Adam's learning rate, depth, and data encoding strategy are found to be very important, as we expected. On the other hand, the less considered hyperparameters, such as the particular choice of the entangling gate and using 3 rotation types in the variational layer, are in the least important group. Hence, our experiment confirmed expected patterns and revealed new insights for the selection of the quantum model. We confirmed these results by cross-checking these results against an extensive experiment, where we ran several hyperparameter optimization processes, each time optimizing all but one hyperparameter. When such a hyperparameter optimization run was still successful even when a given hyperparameter was not optimized, that indicates that the hyperparameter was not important, and vice-versa. There was a high correlation between the hyperparameters that were found to be important by both results. For example, both results rank learning rate, depth, and use reuploading among the most important hyperparameters.

Finally, following [56], we utilize good configurations across datasets to create data-driven priors that can be used in hyperparameter optimization processes

to sample from (opposed to the commonly-used uniform or log-uniform prior). We demonstrated that using prior information of good values of hyperparameters benefits the hyperparameter optimization process by comparing a hyperband optimization process with the data-driven priors against a hyperband optimization process with uniform priors. This experiment was repeated with many different settings for hyperband, ranging in many different hyperparameter configurations of the hyperparameter optimization method. The results indicate that the data-driven priors outperform the uniform priors in 77.2% of the cases.

For future work, we plan further to investigate methods from the field of automated machine learning to be applied to quantum neural networks [8,17,44]. Indeed, our experiments have shown the importance of hyperparameter optimization, which should become standard practice and part of the protocols applied within the community. We further envision functional ANOVA to be employed in future works related to quantum machine learning and understanding how to apply quantum models in practice. For instance, it would be interesting to consider quantum data, for which quantum machine learning models may have an advantage. Plus, extending hyperparameter importance to techniques for scaling to a large number of features with the number of qubits, such as dimensionality reduction or divide-and-conquer techniques, can be left for future work. Finally, this type of study can also be extended to different noisy hardware and towards algorithm/model selection and design. For example, choosing which hardware works best for machine learning tasks becomes possible if we have access to a cluster of different quantum computers. One could also extend our work with meta-learning [8], where a model configuration is selected based on meta-features created from dataset features. Such types of studies already exist for parameterized quantum circuits applied to combinatorial optimization [46,49,61].

Acknowledgements CM, YJP, and VD acknowledge support from TotalEnergies. This work was supported by the Dutch Research Council (NWO/OCW) as part of the Quantum Software Consortium programme (project number 024.003.037). This research is also supported by the project NEASQC, funded by the European Union's Horizon 2020 research and innovation programme (grant agreement No 951821).

Funding This work was supported by the Dutch Research Council (NWO/OCW) as part of the Quantum Software Consortium programme (project number 024.003.037). This research is also supported by the project NEASQC, funded by the European Union's Horizon 2020 research and innovation programme (grant agreement No 951821).

Authors' contributions All authors contributed to the study conception and design of the presented work. CM and YJP performed the numerical experiments. All authors contributed to the critical review, iterative improvement, and approval of the final version of the manuscript.

Availability of data and materials All the datasets used in the study are open source.

Code availability All code associated with this paper is publicly available from `https://github.com/chMoussa/prior_qnn_surrogate_search`.

Declarations

Conflict of interests The authors declare no conflicts of interest.

Ethics approval Not applicable.

Consent for publication Not applicable.

Consent to participate Not applicable.

References

1. ANIS, M.S., et al.: Qiskit: An open-source framework for quantum computing (2021). DOI 10.5281/zenodo.2573505
2. Benedetti, M., Lloyd, E., Sack, S., Fiorentini, M.: Parameterized quantum circuits as machine learning models. Quantum Science and Technology **4**(4), 043001 (2019)
3. Bergholm, V., Izaac, J.A., Schuld, M., Gogolin, C., Killoran, N.: Pennylane: Automatic differentiation of hybrid quantum-classical computations. CoRR **abs/1811.04968** (2018)
4. Biedenkapp, A., Lindauer, M., Eggensperger, K., Hutter, F., Fawcett, C., Hoos, H.: Efficient parameter importance analysis via ablation with surrogates. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31 (2017)
5. Biedenkapp, A., Marben, J., Lindauer, M., Hutter, F.: CAVE: configuration assessment, visualization and evaluation. In: Learning and Intelligent Optimization - 12th International Conference, *Lecture Notes in Computer Science*, vol. 11353, pp. 115–130. Springer (2018)
6. Biedenkapp, A., Marben, J., Lindauer, M., Hutter, F.: Cave: Configuration assessment, visualization and evaluation. In: Learning and Intelligent Optimization: 12th International Conference, LION 12, Kalamata, Greece, June 10–15, 2018, Revised Selected Papers 12, pp. 115–130. Springer (2019)
7. Bischl, B., Casalicchio, G., Feurer, M., Gijsbers, P., Hutter, F., Lang, M., Mantovani, R.G., van Rijn, J.N., Vanschoren, J.: Openml benchmarking suites. In: Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (2021)
8. Brazdil, P., van Rijn, J.N., Soares, C., Vanschoren, J.: Metalearning: Applications to Automated Machine Learning and Data Mining, 2nd edn. Springer (2022)
9. Breiman, L.: Random forests. Machine Learning **45**(1), 5–32 (2001)
10. Broughton, M., et al.: Tensorflow quantum: A software framework for quantum machine learning. arXiv:2003.02989 (2020)
11. Caro, M.C., Gil-Fuster, E., Meyer, J.J., Eisert, J., Sweke, R.: Encoding-dependent generalization bounds for parametrized quantum circuits. Quantum **5**, 582 (2021)
12. Cerezo, M., Sone, A., Volkoff, T., Cincio, L., Coles, P.J.: Cost function dependent barren plateaus in shallow parametrized quantum circuits. Nature communications **12**(1), 1791 (2021)
13. Du, Y., Huang, T., You, S., Hsieh, M.H., Tao, D.: Quantum circuit architecture search for variational quantum algorithms. npj Quantum Information **8**(1), 62 (2022). DOI 10.1038/s41534-022-00570-y. URL `https://doi.org/10.1038/s41534-022-00570-y`
14. Eggensperger, K., Hutter, F., Hoos, H.H., Leyton-Brown, K.: Efficient benchmarking of hyperparameter optimizers via surrogates. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 1114–1120. AAAI Press (2015)
15. Eggensperger, K., Hutter, F., Hoos, H.H., Leyton-Brown, K.: Efficient benchmarking of hyperparameter optimizers via surrogates. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 1114–1120. AAAI Press (2015)
16. Farhi, E., Goldstone, J., Gutmann, S.: A quantum approximate optimization algorithm. arXiv:1411.4028 (2014)
17. Feurer, M., Eggensperger, K., Falkner, S., Lindauer, M., Hutter, F.: Auto-sklearn 2.0: Hands-free automl via meta-learning. J Machine Learn Res **23**(261), 1–61 (2020)
18. Georgescu, I.M., Ashhab, S., Nori, F.: Quantum simulation. Review of Modern Physics **86**, 153–185 (2014)
19. G'omez, R.B., O'Meara, C., Cortiana, G., Mendl, C.B., Bernab'e-Moreno, J.: Towards autoqml: A cloud-based automated circuit architecture search framework. 2022 IEEE 19th International Conference on Software Architecture Companion (ICSA-C) pp. 129–136 (2022)
20. Google: Cirq: A python framework for creating, editing, and invoking noisy intermediate scale quantum circuits (2018). URL `https://github.com/quantumlib/Cirq`
21. Grant, E., Wossnig, L., Ostaszewski, M., Benedetti, M.: An initialization strategy for addressing barren plateaus in parametrized quantum circuits. Quantum **3**, 214 (2019). DOI 10.22331/q-2019-12-09-214. URL `https://quantum-journal.org/papers/q-2019-12-09-214/`
22. Haug, T., Self, C.N., Kim, M.S.: Quantum machine learning of large datasets using randomized measurements. Machine Learning: Science and Technology **4**(1), 015005 (2023). DOI 10.1088/2632-2153/acb0b4
23. Havlíček, V., Córcoles, A.D., Temme, K., Harrow, A.W., Kandala, A., Chow, J.M., Gambetta, J.M.: Supervised learning with quantum-enhanced feature spaces. Nature **567**(7747), 209–212 (2019)

24. Heimann, D., Hohenfeld, H., Wiebe, F., Kirchner, F.: Quantum deep reinforcement learning for robot navigation tasks. CoRR **abs/2202.12180** (2022)
25. Hooker, G.: Generalized functional anova diagnostics for high-dimensional functions of dependent variables. Journal of Computational and Graphical Statistics **16**(3), 709–732 (2007)
26. Hutter, F., Hoos, H., Leyton-Brown, K.: An efficient approach for assessing hyperparameter importance. In: Proceedings of the 31th International Conference on Machine Learning, ICML 2014, *JMLR Workshop and Conference Proceedings*, vol. 32, pp. 1130–1144 (2014)
27. Hutter, F., Hoos, H.H., Leyton-Brown, K.: Identifying key algorithm parameters and instance features using forward selection. In: Learning and Intelligent Optimization: 7th International Conference, LION 7, Catania, Italy, January 7-11, 2013, Revised Selected Papers 7, pp. 364–381. Springer (2013)
28. Hutter, F., Kotthoff, L., Vanschoren, J. (eds.): Automated Machine Learning - Methods, Systems, Challenges. Springer (2019)
29. Jerbi, S., Fiderer, L.J., Poulsen Nautrup, H., Kübler, J.M., Briegel, H.J., Dunjko, V.: Quantum machine learning beyond kernel methods. Nature Communications **14**(1), 517 (2023)
30. Jerbi, S., Gyurik, C., Marshall, S., Briegel, H.J., Dunjko, V.: Parametrized quantum policies for reinforcement learning. In: Advances in Neural Information Processing Systems 34, pp. 28362–28375 (2021)
31. Kandala, A., Mezzacapo, A., Temme, K., Takita, M., Brink, M., Chow, J.M., Gambetta, J.M.: Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. Nature **549**(7671), 242–246 (2017)
32. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations (2015)
33. Kulshrestha, A., Safro, I.: Beinit: Avoiding barren plateaus in variational quantum algorithms. In: 2022 IEEE International Conference on Quantum Computing and Engineering (QCE), pp. 197–203 (2022). DOI 10.1109/QCE53715.2022.00039
34. Larocca, M., Czarnik, P., Sharma, K., Muraleedharan, G., Coles, P.J., Cerezo, M.: Diagnosing barren plateaus with tools from quantum optimal control. Quantum **6**, 824 (2022)
35. Larraanaga, P., Lozano, J.A.: Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation. Kluwer Academic Publishers, Norwell, MA, USA (2001)
36. Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., Talwalkar, A.: Hyperband: A novel bandit-based approach to hyperparameter optimization. J. Mach. Learn. Res. **18**(1), 6765–6816 (2017)
37. Liu, J.G., Wang, L.: Differentiable learning of quantum circuit born machines. Physical Review A **98**, 062324 (2018)
38. Liu, Y., Arunachalam, S., Temme, K.: A rigorous and robust quantum speed-up in supervised machine learning. Nature Physics **17**(9), 1013–1017 (2021)
39. Marshall, S.C., Gyurik, C., Dunjko, V.: High dimensional quantum machine learning with small quantum computers. CoRR **abs/2203.13739** (2022)
40. Mathur, N., Landman, J., Li, Y.Y., Strahm, M., Kazdaghli, S., Prakash, A., Kerenidis, I.: Medical image classification via quantum neural networks (2021)
41. McClean, J.R., Boixo, S., Smelyanskiy, V.N., Babbush, R., Neven, H.: Barren plateaus in quantum neural network training landscapes. Nature Communications **9**(1), 1–6 (2018). DOI 10.1038/s41467-018-07090-4. URL `https://www.nature.com/articles/s41467-018-07090-4`
42. Mensa, S., Sahin, E., Tacchino, F., Barkoutsos, P.K., Tavernelli, I.: Quantum machine learning framework for virtual screening in drug discovery: a prospective quantum advantage. CoRR **abs/2204.04017** (2022)
43. Mitarai, K., Negoro, M., Kitagawa, M., Fujii, K.: Quantum circuit learning. Physical Review A **98**, 032309 (2018)
44. Mohr, F., van Rijn, J.N.: Learning curves for decision making in supervised machine learning - A survey. CoRR **abs/2201.12150** (2022)
45. Moll, N., et al: Quantum optimization using variational algorithms on near-term quantum devices. Quantum Science and Technology **3**(3), 030503 (2018)
46. Moussa, C., Calandra, H., Dunjko, V.: To quantum or not to quantum: towards algorithm selection in near-term quantum optimization. Quantum Science and Technology **5**(4), 044009 (2020)

47. Moussa, C., Gordon, M.H., Baczyk, M., Cerezo, M., Cincio, L., Coles, P.J.: Resource frugal optimizer for quantum machine learning. arXiv:2211.04965 (2022). URL `https://arxiv.org/abs/2211.04965`
48. Moussa, C., van Rijn, J.N., Bäck, T., Dunjko, V.: Hyperparameter importance of quantum neural networks across small datasets. In: P. Pascal, D. Ienco (eds.) Discovery Science, pp. 32–46. Springer Nature Switzerland, Cham (2022)
49. Moussa, C., Wang, H., Bäck, T., Dunjko, V.: Unsupervised strategies for identifying optimal parameters in quantum approximate optimization algorithm. EPJ Quantum Technology **9**(1) (2022)
50. Napp, J.: Quantifying the barren plateau phenomenon for a model of unstructured variational ans\"{a} tze. arXiv preprint arXiv:2203.06174 (2022)
51. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information: 10th Anniversary Edition. Cambridge University Press (2011)
52. Pérez-Salinas, A., Cervera-Lierta, A., Gil-Fuster, E., Latorre, J.I.: Data re-uploading for a universal quantum classifier. Quantum **4**, 226 (2020)
53. Pesah, A., Cerezo, M., Wang, S., Volkoff, T., Sornborger, A.T., Coles, P.J.: Absence of barren plateaus in quantum convolutional neural networks. Physical Review X **11**(4), 041011 (2021). DOI 10.1103/PhysRevX.11.041011. URL `https://journals.aps.org/prx/abstract/10.1103/PhysRevX.11.041011`
54. Peters, E., Caldeira, J., Ho, A., Leichenauer, S., Mohseni, M., Neven, H., Spentzouris, P., Strain, D., Perdue, G.N.: Machine learning of high dimensional data on a noisy quantum processor. npj Quantum Information **7**(1), 161 (2021)
55. Preskill, J.: Quantum Computing in the NISQ era and beyond. Quantum **2**, 79 (2018)
56. van Rijn, J.N., Hutter, F.: Hyperparameter importance across datasets. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, pp. 2367–2376. ACM (2018)
57. Sack, S.H., Medina, R.A., Michailidis, A.A., Kueng, R., Serbyn, M.: Avoiding barren plateaus using classical shadows. PRX Quantum **3**(2), 020365 (2022)
58. Sagingalieva, A.B., Kurkin, A., Melnikov, A.A., Kuhmistrov, D., Perelshtein, M.R., Melnikov, A.A., Skolik, A., Dollen, D.V.: Hyperparameter optimization of hybrid quantum neural networks for car classification. ArXiv **abs/2205.04878** (2022)
59. Sajjan, M., Li, J., Selvarajan, R., Sureshbabu, S.H., Kale, S.S., Gupta, R., Singh, V., Kais, S.: Quantum machine learning for chemistry and physics. Chemical Society Reviews **51**(15), 6475–6573 (2022)
60. Saltelli, A., Sobol, I.: Sensitivity analysis for nonlinear mathematical models: Numerical experience. Matematicheskoe Modelirovanie **7** (1995)
61. Sauvage, F., Sim, S., Kunitsa, A.A., Simon, W.A., Mauri, M., Perdomo-Ortiz, A.: Flip: A flexible initializer for arbitrarily-sized parametrized quantum circuits. CoRR **abs/2103.08572** (2021)
62. Schetakis, N., Aghamalyan, D., Boguslavsky, M., Griffin, P.: Binary classifiers for noisy datasets: a comparative study of existing quantum machine learning frameworks and some new approaches. CoRR **abs/2111.03372** (2021)
63. Schuld, M., Killoran, N.: Is quantum advantage the right goal for quantum machine learning? Prx Quantum **3**(3), 030101 (2022)
64. Schuld, M., Sweke, R., Meyer, J.J.: Effect of data encoding on the expressive power of variational quantum-machine-learning models. Physical Review A **103**, 032430 (2021)
65. Sharma, A., van Rijn, J.N., Hutter, F., Müller, A.: Hyperparameter importance for image classification by residual neural networks. In: Discovery Science - 22nd International Conference, *Lecture Notes in Computer Science*, vol. 11828, pp. 112–126. Springer (2019)
66. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. Siam Review **41**, 303–332 (1999)
67. Sim, S., Johnson, P.D., Aspuru-Guzik, A.: Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms. Advanced Quantum Technologies **2**(12), 1900070 (2019)
68. Skolik, A., Jerbi, S., Dunjko, V.: Quantum agents in the gym: a variational quantum algorithm for deep q-learning. Quantum **6**, 720 (2022)
69. Sobol, I.M.: Sensitivity estimates for nonlinear mathematical models. Mathematical Modelling and Computational Experiments **1**(4), 407–414 (1993)
70. Sweke, R., Seifert, J., Hangleiter, D., Eisert, J.: On the quantum versus classical learnability of discrete distributions. Quantum **5**, 417 (2021)

71. Wang, H., Gu, J., Ding, Y., Li, Z., Chong, F.T., Pan, D.Z., Han, S.: Quantumnat: quantum noise-aware training with noise injection, quantization and normalization. In: Proceedings of the 59th ACM/IEEE Design Automation Conference, pp. 1–6 (2022)
72. Wang, H., Li, Z., Gu, J., Ding, Y., Pan, D.Z., Han, S.: Qoc: quantum on-chip training with parameter shift and gradient pruning. In: Proceedings of the 59th ACM/IEEE Design Automation Conference, pp. 655–660 (2022)
73. Wossnig, L.: Quantum machine learning for classical data. CoRR **abs/2105.03684** (2021)
74. Zhang, K., Liu, L., Hsieh, M.H., Tao, D.: Escaping from the barren plateau via gaussian initializations in deep variational quantum circuits. Advances in Neural Information Processing Systems **35**, 18612–18627 (2022)
75. Zhang, S.X., Hsieh, C.Y., Zhang, S., Yao, H.: Differentiable quantum architecture search. Quantum Science and Technology **7**(4), 045023 (2022)
76. Zoufal, C., Lucchi, A., Woerner, S.: Quantum generative adversarial networks for learning and loading random distributions. npj Quantum Information **5**(1), 103 (2019)

A Marginals of hyperparameters

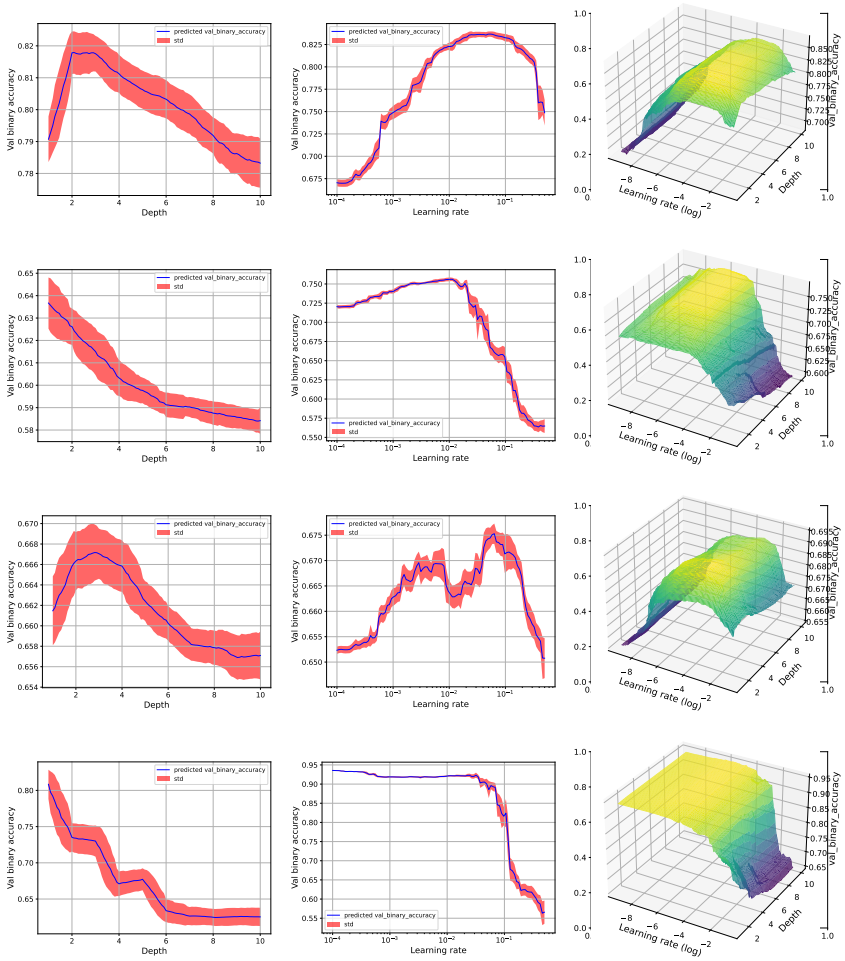


Fig. 9: Marginals of hyperparameters for a quantum neural network with validation accuracy as performance metric on four datasets (from top to below, the order is breast-w, phoneme, diabetes, wilt). The hyperparameters correspond to the number of layers, also known as depth (left column), the learning rate used during training (middle column), and their combination (right columns).

B KDE priors of important hyperparameters

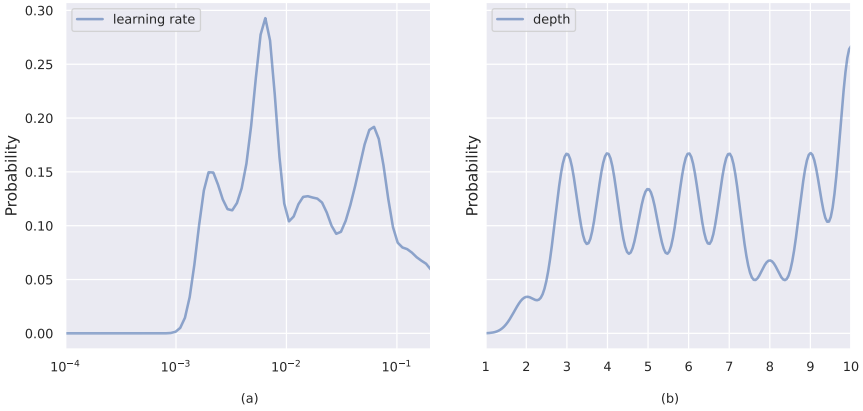


Fig. 10: Priors obtained for the top-2 important hyperparameters (learning rate (a) and depth (b)) for a quantum neural network. The x-axis denotes the values, while the y-axis denotes the probability of the value being sampled.