The power of quantum neural networks

Amira Abbas[1,2], David Sutter[1], Christa Zoufal[1,3], Aurelien Lucchi[3], Alessio Figalli[3], and Stefan Woerner[1]

[1] *IBM Quantum, IBM Research – Zurich*
[2] *University of KwaZulu-Natal, Durban*
[3] *ETH Zurich*

Abstract

It is unknown if near-term quantum computers are advantageous for machine learning tasks. In this work, we address this question by trying to understand how powerful and trainable quantum machine learning models are, relative to popular classical neural networks. We propose the *effective dimension*—a measure that captures these qualities—and prove that it can be used to assess any statistical model's ability to generalize on new data. Crucially, the effective dimension is a data-dependent measure that also depends on the Fisher information, which allows us to gauge the ability of a model to train. We demonstrate numerically that a class of quantum neural networks is able to achieve a significantly better effective dimension than comparable feedforward networks and train faster, suggesting an advantage for quantum machine learning, which we verify on real quantum hardware.

The power of a model lies in its ability to fit a variety of functions [1]. In machine learning, power is often referred to as a model's capacity to express different relationships between variables [2]. Deep neural networks have proven to be extremely powerful models, capable of capturing intricate relationships by learning from data [3]. Quantum neural networks serve as a newer class of machine learning models that are deployed on quantum computers and use quantum effects such as superposition, entanglement, and interference, to do computation. Some proposals for quantum neural networks include [4–11] and hint at potential advantages, such as speed-ups in training and faster processing. Whilst there has been much development in the growing field of quantum machine learning, a systematic study of the trade-offs between quantum and classical models has yet to be conducted [12]. In particular, the question of whether quantum neural networks are more powerful than classical neural networks, is still open.

A common way to quantify the power of a model is by its complexity [13]. In statistical learning theory, the *Vapnik-Chervonenkis* (VC) *dimension* is an established complexity measure, where error bounds on how well a model generalizes (i.e., performs on unseen data), can be derived [14]. Although the VC dimension has attractive properties in theory, computing it in practice is notoriously difficult. Further, using the VC dimension to bound generalization error requires several unrealistic assumptions, including that the model has access to infinite data [15, 16]. The measure also scales with the number of parameters in the model and ignores the distribution of data. Since modern deep neural networks are heavily overparameterized, generalization bounds based on the VC dimension, and other measures alike, are typically vacuous [17, 18].

In [19], the authors analysed the expressive power of parameterized quantum circuits using *memory capacity*, and found that quantum neural networks had limited advantages over classical neural networks. Memory capacity is, however, closely related to the VC dimension and is thus, subject to similar criticisms. In [20], a quantum neural network is presented which exhibits a higher expressibility than certain classical models, captured by the types of probability distributions it can generate. Another result from [21] is based on strong heuristics and provides systematic examples of possible advantages for quantum neural networks.

We turn our attention to measures that are easy to estimate in practice and importantly incorporate the distribution of data. In particular, measures such as the *effective dimension* have been motivated from an information-theoretic standpoint and depend on the *Fisher information*; a

quantity that describes the geometry of a model's parameter space and is essential in both statistics and machine learning [22–24]. We argue that the effective dimension is a robust capacity measure through proof of a generalization error bound and supporting numerical analyses, and subsequently use this measure to study the power of a popular class of neural networks in both classical and quantum regimes.

Despite a lack of quantitative statements on the power of quantum neural networks, another issue is rooted in the trainability of these models. A nice connection between expressibility and trainability for certain classes of quantum neural networks is outlined in [25, 26]. Often, quantum neural networks suffer from a *barren plateau* phenomenon, wherein the loss landscape is perilously flat, and consequently, parameter optimization is extremely difficult [27]. As shown in [28], barren plateaus may be noise-induced, where certain noise models are assumed on the hardware. In other words, the effect of hardware noise can make it very difficult to train a quantum model. Additionally, barren plateaus can be circuit-induced, which relates to the design of a model and random parameter initialization. Methods to avoid the latter have been explored in [29–32], but noise-induced barren plateaus remain problematic.

A particular attempt to understand the loss landscape of quantum models uses the Hessian [33], that quantifies the curvature of a model's loss function at a point in its parameter space [34]. Properties of the Hessian, such as its spectrum, provide useful diagnostic information about the trainability of a model [35]. It was discovered that the entries of the Hessian vanish exponentially in models suffering from a barren plateau [36]. For certain loss functions, the Fisher information matrix coincides with the Hessian of the loss function [37]. Consequently, we can examine the trainability of quantum and classical neural networks by analysing the Fisher information matrix, which is incorporated by the effective dimension. In this way, we may explicitly relate the effective dimension to model trainability [38].

We find that a class of quantum neural networks is able to achieve a significantly higher capacity and faster training ability numerically, than comparable classical feedforward neural networks. A higher capacity is captured by a higher effective dimension, whilst faster training implies that a model will reach a lower training error than another comparable model for a fixed number of training iterations. More generally, trainability is assessed by leveraging the information-theoretic properties of the Fisher information, which we connect to the barren plateau phenomenon. Our experiments reveal that how you encode data in a quantum neural network influences the likelihood of your model encountering a barren plateau. A quantum neural network with a data encoding strategy that is easy to simulate classically, seems more likely to encounter a barren plateau, whilst a harder encoding strategy shows resilience to the phenomenon. Noise, however, remains problematic by inhibiting training in general.

Results

Quantum neural networks

Quantum neural networks are a subclass of variational quantum algorithms which comprise of quantum circuits that contain parameterized gate operations [39]. Information (usually in the form of classical data) is first encoded into a quantum state via a state preparation routine called a quantum feature map [40]. The choice of feature map is geared toward enhancing the performance of the quantum neural network and is typically neither optimized nor trained, though this idea was discussed in [41]. Once data is encoded into a quantum state, a model, called a variational model, is applied. The variational model contains parameterized gate operations which are optimized for a particular task, analogous to classical machine learning techniques [5–7, 42]. The final output of the quantum neural network is extracted from measurements made to the quantum circuit after the variational model is applied. These measurements are often converted to labels or predictions through classical post-processing before being passed to a loss function, where the idea is to choose parameters for the variational model that minimize the loss function.

The quantum models we use can be summarised in Figure 1 with details of the structure and implementation in the Methods section. We create two model variants, one which we call a quantum neural network and the other, an easy quantum model.

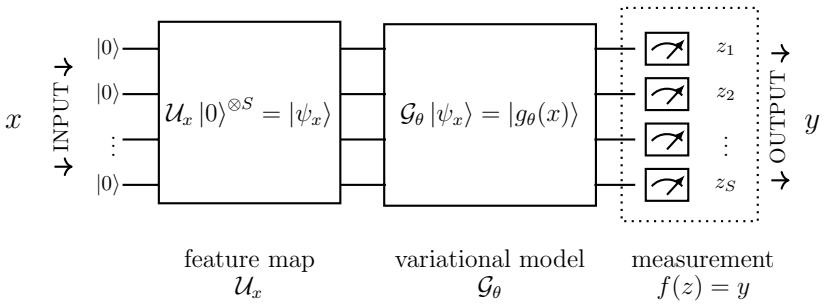


Figure 1: **Overview of the quantum neural network** used in this study. The input $x \in \mathbb{R}^{s_{\text{in}}}$ is encoded into an S-qubit Hilbert space by applying the feature map $|\psi_x\rangle := \mathcal{U}_x |0\rangle^{\otimes S}$. This state is then evolved via a variational form $|g_\theta(x)\rangle := \mathcal{G}_\theta |\psi_x\rangle$, where the parameters $\theta \in \Theta$ are chosen to minimize a certain loss function. Finally a measurement is performed whose outcome $z = (z_1, \ldots, z_S)$ is post-processed to extract the output of the model $y := f(z)$.

The Fisher information

A way to assess the information gained by a particular parameterization of a statistical model is epitomized by the Fisher information. By defining a neural network as a statistical model, we can describe the joint relationship between data pairs (x, y) as $p(x, y; \theta) = p(y|x; \theta)p(x)$ for all $x \in \mathcal{X} \subset \mathbb{R}^{s_{\text{in}}}$, $y \in \mathcal{Y} \subset \mathbb{R}^{s_{\text{out}}}$ and $\theta \in \Theta \subset [-1, 1]^d$. This is achieved by applying an appropriate post-processing function in both classical and quantum networks. In the classical network, we apply a softmax function to the last layer. In the quantum network, we obtain probabilities based on the parity of the output bit strings. The input distribution, $p(x)$ is a prior distribution and the conditional distribution, $p(y|x; \theta)$ describes the input-output relation of the model for a fixed $\theta \in \Theta$. The full parameter space Θ forms a Riemannian space which gives rise to a Riemannian metric, namely, the Fisher information matrix

$$F(\theta) = \mathbb{E}_{(x,y)\sim p}\Big[\frac{\partial}{\partial\theta} \log p(x, y; \theta) \frac{\partial}{\partial\theta} \log p(x, y; \theta)^\mathsf{T}\Big] \in \mathbb{R}^{d\times d},$$

that can be approximated by the empirical Fisher information matrix

$$\tilde{F}_k(\theta) = \frac{1}{k} \sum_{j=1}^{k} \frac{\partial}{\partial\theta} \log p(x_j, y_j; \theta) \frac{\partial}{\partial\theta} \log p(x_j, y_j; \theta)^\mathsf{T}, \tag{1}$$

where $(x_j, y_j)_{j=1}^k$ are i.i.d. drawn from the distribution $p(x, y; \theta)$ [37]. By definition, the Fisher information matrix is positive semidefinite and hence, its eigenvalues are non-negative, real numbers.

The Fisher information conveniently helps capture the sensitivity of a neural network's output relative to movements in the parameter space [43]. In [44], the authors leverage geometric invariances associated with the Fisher information, to produce the *Fisher-Rao norm*, a robust norm-based capacity measure defined as the quadratic form $\|\theta\|_{\text{fr}}^2 := \theta^\mathsf{T} F(\theta)\theta$ for a vectorized parameter set, θ. Notably, the Fisher-Rao norm acts as an umbrella for several other existing norm-based measures [45–47] and has demonstrated desirable properties both theoretically, and empirically.

The effective dimension

The effective dimension is a complexity measure motivated by information geometry, with useful qualities. The goal of the effective dimension is to estimate the size that a model occupies in model space—the space of all possible functions for a particular model class, where the Fisher information matrix serves as the metric. Whilst there are many ways to define the effective dimension, a useful definition is presented in [22] which is designed to be operationally meaningful in settings where data is limited. More precisely, the number of data observations determines a natural scale or

resolution used to observe model space. This is beneficial in practice where data is oftentimes scarce and can help in understanding how data availability influences the accurate capture of model complexity.

The effective dimension is motivated by the theory of minimum description length which is a model selection principle favoring models with the shortest description of the given data. Based on this principle, it can be shown that the complexity at size n of a model is given by

$$\frac{d}{2}\log\frac{n}{2\pi}+\log\left(\int_\Theta\sqrt{\det F(\theta)}\mathrm{d}\theta\right)+o(1)\,,$$

where $o(1)$ vanishes as $n\to\infty$ [48]. The first term containing d is usually interpreted as the dimension of the model, whilst the second term is known as the geometric complexity. Information geometric manipulations allow us to combine both terms into a single expression, referred to as the *effective dimension* [22].

Definition 1. The *effective dimension* of a statistical model $\mathcal{M}_\Theta := \{p(\cdot,\cdot;\theta) : \theta\in\Theta\}$ with respect to $\gamma\in(0,1]$, a d-dimensional parameter space $\Theta\subset\mathbb{R}^d$ and $n\in\mathbb{N}$, $n>1$ data samples is defined as

$$d_{\gamma,n}(\mathcal{M}_\Theta) := 2\frac{\log\left(\frac{1}{V_\Theta}\int_\Theta\sqrt{\det\left(\mathrm{id}_d+\frac{\gamma n}{2\pi\log n}\hat{F}(\theta)\right)}\,\mathrm{d}\theta\right)}{\log\left(\frac{\gamma n}{2\pi\log n}\right)}\,, \tag{2}$$

where $V_\Theta := \int_\Theta \mathrm{d}\theta\in\mathbb{R}_+$ is the volume of the parameter space. The matrix $\hat{F}(\theta)\in\mathbb{R}^{d\times d}$ is the normalized Fisher information matrix defined as

$$\hat{F}_{ij}(\theta) := d\frac{V_\Theta}{\int_\Theta\mathrm{tr}(F(\theta))\mathrm{d}\theta}F_{ij}(\theta)\,.$$

Remark 2 (Properties of the effective dimension)**.** In the limit $n\to\infty$, the effective dimension converges to the maximal rank $\bar{r} := \max_{\theta\in\Theta} r_\theta$, where $r_\theta\leq d$ denotes the rank of the Fisher information matrix $F(\theta)$. The proof of this result can be seen in Section 2.1 in the Supplementary Information, but it is worthwhile to note that the effective dimension does not necessarily increase monotonically with n, as explained in Section 2.2 in the Supplementary Information. The geometric operational meaning of the effective dimension only holds if n is sufficiently large. We conduct experiments over a wide range of n and ensure that conclusions are drawn from results where the choice of n is sufficient.

Another noteworthy point is that the effective dimension is easy to estimate. To see this, recall that we need (i) to estimate the Fisher information matrix $F(\theta)$ and (ii) calculate the integral over the parameter space Θ given in (2). Both of these steps can be done via Monte Carlo integration which, in practice, does not depend on the model's dimension.

There are also two minor differences between (2) and the effective dimension from [22]: the presence of the constant $\gamma\in(0,1]$, and the $\log n$ term. These modifications are helpful in proving a generalization bound such that the effective dimension may be interpreted as a bounded capacity measure, serving as a useful tool to analyse the power of statistical models. We demonstrate this in the Methods section.

The Fisher information spectrum

Classically, the Fisher information spectrum reveals a lot about the optimization landscape of a model. The magnitude of the eigenvalues illustrates the curvature of a model for a particular parameterization. If there is a large concentration of eigenvalues near zero, the optimization landscape will be predominantly flat and parameters become difficult to train with gradient-based methods [38]. On the quantum side, we show in Section 4 of the Supplementary Information that if a model is in a barren plateau, the Fisher information spectrum will be concentrated around zero and training also becomes unfeasible. We can, thus, make connections to capacity via the spectrum of the Fisher information matrix by using the effective dimension. Looking closely at (2),

we see that the effective dimension converges to its maximum the fastest if the Fisher information spectrum is evenly distributed, on average.

We analyse the Fisher information spectra for the quantum neural network, the easy quantum model and all possible configurations of the fully connected feedforward neural network—where all models share a specified triple $(d, s_{\text{in}}, s_{\text{out}})$. To be robust, we sample 100 sets of parameters uniformly on $\Theta = [-1, 1]^d$ and compute the Fisher information matrix 100 times using a standard Gaussian prior for the data. The resulting average distributions of the eigenvalues of these 100 Fisher information matrices are plotted in the top row of Figure 2 for $d = 40$, $s_{\text{in}} = 4$ and $s_{\text{out}} = 2$. A sensitivity analysis is included in Section 3.1 of the Supplementary Information to verify that 100 parameter samples are reasonable for the models we consider. In higher dimensions, this number will need to increase. The bottom row of Figure 2 contains the distribution for eigenvalues less than 1.

The classical model depicted in Figure 2 is the one with the highest average rank of Fisher information matrices. The majority of eigenvalues are negligible (of the order 10^{-14}), with a few very large values. This behavior is observed across all classical configurations that we consider and is consistent with results from literature, where the Fisher information matrix of non-linear feedforward neural networks is known to be highly degenerate, with a few large eigenvalues [38]. The concentration around zero becomes more evident in bottom row of the plot, depicting the eigenvalue distribution of just the eigenvalues less than 1.

The easy quantum model also has most of its eigenvalues close to zero, and whilst there are some large eigenvalues, their magnitudes are not as extreme as the classical model.

The quantum neural network, on the other hand, has a distribution of eigenvalues that is more uniform, with no outlying values. This can be seen from the range of the eigenvalues on the y-axis in Figure 2. This distribution remains more or less constant as the number of qubits increase, even in the presence of hardware noise (see Section 3.2 of the Supplementary Information) and has implications for capacity and trainability which we examine next.

Capacity analysis

In Figure 3a, we plot the normalized effective dimension for all three model types. The normalization ensures that the effective dimension lies between 0 and 1 by simply dividing by d. The convergence speed of the effective dimension to its maximum is slowed down by smaller eigenvalues and uneven Fisher information spectra. Since the classical models contain highly degenerate Fisher matrices, the effective dimension converges the slowest, followed by the easy quantum model. The quantum neural network has non-degenerate Fisher information matrices and more even spectra, hence, it consistently achieves the highest effective dimension over all ranges of finite data considered. Intuitively, we would expect the additional effects of quantum operations such as entanglement and superposition—if used effectively—to generate models with higher capacity. Thus, the quantum neural network with a strong feature map is expected to deliver the highest capacity, but recall that in the limit $n \to \infty$, all models will converge to an effective dimension equal to the maximum rank of the Fisher information matrix (see Remark 2).

To support these observations, we calculate the capacity of each model using a different measure, the Fisher-Rao norm [44]. The average Fisher-Rao norm after training each model 100 times is roughly 250% higher in the quantum neural network than in the classical neural network, with the easy quantum model in between (see Section 3.3 of the Supplementary Information).

Trainability

The observed Fisher information spectrum of the feedforward model is known to have undesirable optimization properties, where the outlying eigenvalues slow down training and loss convergence [35]. These large eigenvalues become even more pronounced in bigger models, as seen in Supplementary Figure 5. Upon examining the easy quantum model over an increasing system size, the average Fisher spectrum becomes more concentrated around zero. This is characteristic of models encountering a barren plateau, presenting another unfavorable scenario for optimization. The quantum neural network, however, maintains its more even distribution of eigenvalues as the number of qubits and trainable parameters increase. Additionally, a large amount of the eigenvalues are not near zero. This highlights the importance of a feature map in a quantum model.

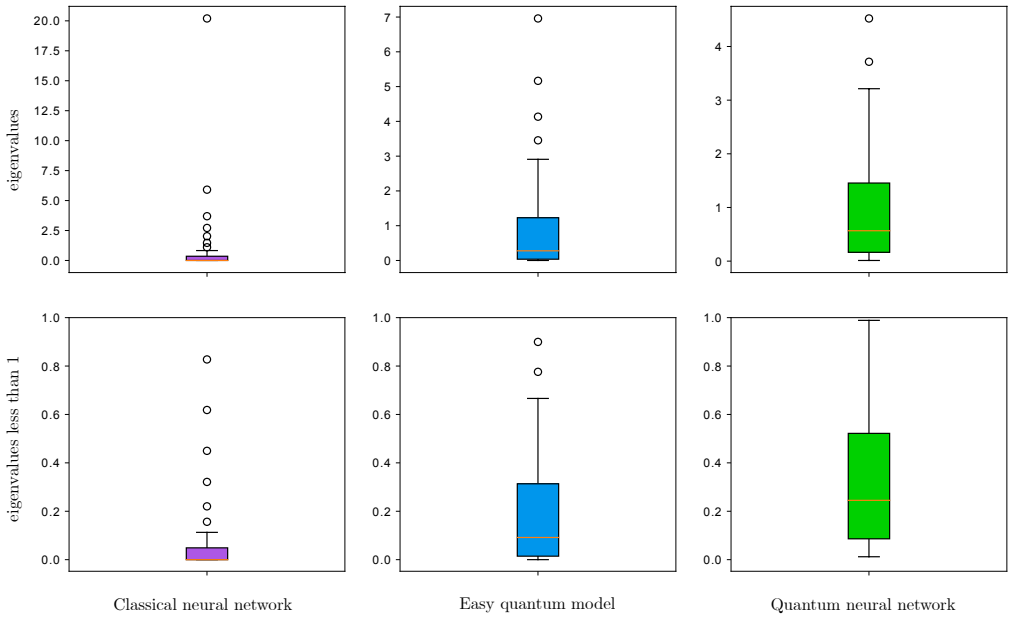
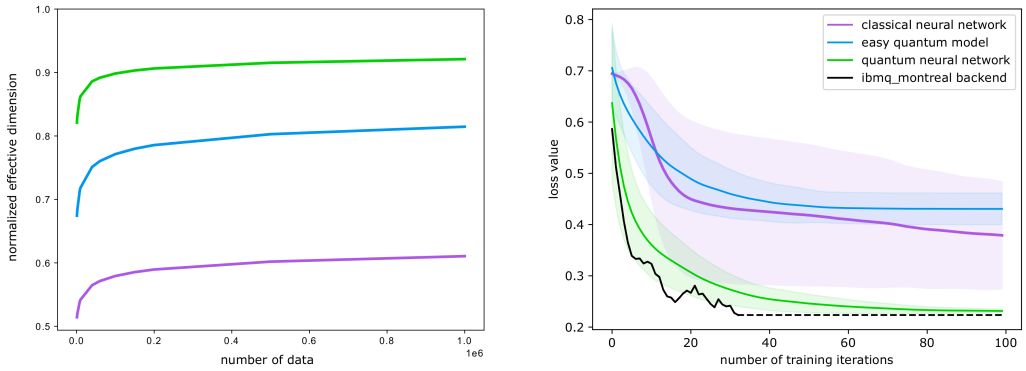


Figure 2: **Average Fisher information spectrum distribution.** Here, box plots are used to reveal the average distribution of eigenvalues of the Fisher information matrix for the classical feedforward neural network and the quantum neural network with two different feature maps. The dots in the box plots represent outlier values relative to the length of the whiskers. The lower whisker is at the lowest datum above Q1 - 1.5*(Q3-Q1), and the upper whisker at the highest datum below Q3 + 1.5*(Q3-Q1), where Q1 and Q3 are the first and third quartiles. This is a standard method to compute these plots. The easy quantum model has a classically simulable data encoding strategy, whilst the quantum neural network's encoding scheme is conjectured to be difficult. In each model, we compute the Fisher information matrix 100 times using parameters sampled uniformly at random and plot the resulting average distribution of the eigenvalues. We fix the number of trainable parameters $d = 40$, input size $s_{\text{in}} = 4$ and output size $s_{\text{out}} = 2$. The top row contains the average distribution of all eigenvalues for each model, whilst the bottom row contains the average distribution of eigenvalues less than 1 for each model.

The harder data encoding strategy used in the quantum neural network seems to structurally change the optimization landscape and remove the flatness, usually associated with suboptimal optimization conditions, such as barren plateaus.

We confirm the training statements for all three models with an experiment illustrated in Figure 3b. Using a cross-entropy loss function, optimized with `ADAM` for a fixed number of training iterations = 100 and an initial learning rate = 0.1, the quantum neural network trains to a lower loss, faster than the other two models over an average of 100 trials. To support the promising training performance of the quantum neural network, we also train it once on real hardware using the `ibmq_montreal` 27-`qubit` device. We reduce the number of CNOT-gates by only considering linear entanglement instead of all-to-all entanglement in the feature map and variational circuit. This is to cope with hardware limitations and could be the reason the hardware training performs even better than the simulated results, as too much entanglement has been shown to have negative effects on model trainability [50]. The full details of the experiment are contained in Section 3.4 of the Supplementary Information. We find that the quantum neural network tangibly demonstrates faster training, however, the addition of hardware noise may still make training difficult, regardless of the optimization landscape (see Section 3.2 of the Supplementary Information).



a) b)

Figure 3: **Normalized effective dimension and training loss** Figure 3a: Normalized effective dimension plotted for the quantum neural network in green, the easy quantum model in blue and the classical feedforward neural network in purple. We fix the input size $s_{\text{in}} = 4$, the output size $s_{\text{out}} = 2$ and number of trainable parameters $d = 40$. Figure 3b: Using the first two classes of the `Iris` dataset [49], we train all three models using $d = 8$ trainable parameters with full batch size. The `ADAM` optimizer with an initial learning rate of 0.1 is selected. For a fixed number of training iterations = 100, we train all models over 100 trials and plot the average training loss along with ± 1 standard deviation. We further verify the performance of the quantum neural network on real quantum hardware and train the model using the `ibmq_montreal 27-qubit` device. We plot the hardware results till they stabilize, at roughly 33 training iterations. The actual hardware implementation contains less CNOT-gates by using linear connectivity for the feature map and variational circuit instead of all-to-all connectivity in order to cope with limited resources, leading to the lower loss values.

Discussion

In stark contrast to classical models, understanding the capacity of quantum neural networks is not well explored. Moreover, classical neural networks are known to produce highly degenerate Fisher information matrices, which can significantly slow down training. For quantum neural networks, no such analysis has been done.

This work attempts to address this gap but leaves room for further research. The feature map in a quantum model plays a large role in determining both its capacity and trainability via the effective dimension and Fisher information spectrum. A deeper investigation needs to be conducted on why the particular higher order feature map used in this study produces a desirable model landscape that induces both a high capacity, and faster training ability. Different variational circuits could also influence the model's landscape and the effects of non-unitary operations, induced through intermediate measurements for example, should be investigated. The Fisher information spectra of certain quantum models seem robust against hardware noise, but trainability remains problematic and the possibility of noise-induced barren plateaus needs examination. Finally, understanding generalization performance on multiple datasets and larger models with complexities that we would be interested in practice, might prove insightful.

Overall, we have shown that quantum neural networks can possess a desirable Fisher information spectrum that enables them to train faster and express more functions than comparable classical and quantum models—a promising reveal for quantum machine learning, which we hope leads to further studies on the power of quantum models.

Methods

Quantum models used in this study

The quantum models used in this study first encode classical data $x \in \mathbb{R}^{s_{\text{in}}}$ into an S-qubit Hilbert space using a feature map, $\mathcal{U}_x$. For the quantum neural network, we use a feature map originally proposed in [51] and in the easy quantum model, we swap out this feature map for one that is easy to simulate classically. Supplementary Figure 1 contains a circuit representation of the feature map from [51] which we refer to as the hard feature map. Here, the number of qubits in the model is chosen to equal the number of feature values of the data, i.e. $S := s_{\text{in}}$. That way, we can associate the same index for each qubit, with each feature value of a data point. For example, if we have data that has 3 feature values, i.e. $\mathbf{x} = (x_1, x_2, x_3)^\mathsf{T}$, we will have a 3 qubit model with qubits $= (q_1, q_2, q_3)$.

The operations in the hard feature map first apply Hadamard gates to each of the qubits, followed by a layer of RZ-gates, whereby the angle of the $Z-$rotation on qubit i depends on the i^{th} feature of the data point $\mathbf{x}$, normalized between $[-1, 1]$. Then, RZZ-gates are applied to every pair of qubits. This time, the value of the controlled $Z-$rotations depends on a product of feature values. For example, if the RZZ-gate is controlled by qubit i and targets qubit j, then the angle of the controlled rotation applied to qubit j is dependent on the product of feature values $x_i x_j$. The RZZ-gates are implemented using a decomposition into two CNOT-gates and one RZ-gate. Thereafter, the RZ and RZZ-gates are repeated once. The classically simulable feature map employed in the easy quantum model, is simply the first sets of Hadamard and RZ-gates with no entanglement between any qubits, as done in the beginning of Supplementary Figure 1. These operations are not repeated.

After the feature map circuit is applied, we apply another set of operations that depend on traiable parameters. We call this the variational circuit, $\mathcal{G}_\theta$. Supplementary Figure 2 depicts the variational form deployed in both the easy quantum model and the quantum neural network. The circuit consists of S qubits, to which parameterized RY-gates are applied to every qubit. Thereafter, CNOT-gates are applied between each pair of qubits in the circuit. Lastly, another set of parameterized RY-gates are applied to every qubit. This circuit has, by definition, $2S$ parameters. If the depth is increased, the entangling layers and second set of parameterized RY-gates are repeated. The number of trainable parameters d can be calculated as $d = (D + 1)S$, where S is equal to the input size of the data s_{in} due to the choice of both feature maps used in this study and D is called the depth of the circuit (i.e. how many times the entanglement and second set of RY operations are repeated).

We lastly measure all qubits in the σ_z basis and classically compute the parity of the output bit strings. For simplicity, we consider binary classification, where the probability of observing class 0 corresponds to the probability of seeing even parity bit strings and similarly, for class 1 with odd parity bit strings.

The reason for the choice of these models' architecture is two-fold: the hard feature map is motivated in [51] to serve as a useful data embedding strategy that is believed to be difficult to simulate classically as the depth and width increase and the easy feature map allows us to benchmark this; and the variational design aims to create more expressive models for quantum algorithms [52]. We benchmark the quantum models against a class of classical models that forms part of the foundation of deep learning, namely feedforward neural networks. We consider all possible topologies with full connectivity for a fixed number of trainable parameters. Networks with and without biases and different activation functions are explored.

Generalization error bounds for the effective dimension

Suppose we are given a hypothesis class, $\mathcal{H}$, of functions mapping from $\mathcal{X}$ to $\mathcal{Y}$ and a training set $\mathcal{S}_n = \{(x_1, y_1), \ldots, (x_n, y_n)\} \in (\mathcal{X} \times \mathcal{Y})^n$, where the pairs (x_i, y_i) are drawn i.i.d. from some unknown joint distribution p. Furthermore, let $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ be a loss function. The challenge is to find a particular hypothesis $h \in \mathcal{H}$ with the smallest possible *expected risk*, defined as $R(h) := \mathbb{E}_{(x,y)\sim p}[L(h(x), y)]$. Since we only have access to a training set $\mathcal{S}_n$, a good strategy to find the best hypothesis $h \in \mathcal{H}$ is to minimize the so called *empirical risk*, defined as $R_n(h) := \frac{1}{n}\sum_{i=1}^{n} L(h(x_i), y_i)$. The difference between the expected and the empirical risk is the

generalization error–an important quantity in machine learning that dictates whether a hypothesis $h \in \mathcal{H}$ learned on a training set will perform well on unseen data, drawn from the unknown joint distribution p [17]. Therefore, an upper bound on the quantity

$$\sup_{h \in \mathcal{H}} |R(h) - R_n(h)|\,, \tag{3}$$

which vanishes as n grows large, is of considerable interest. Capacity measures help quantify the expressiveness and power of $\mathcal{H}$. Thus, the generalization error in (3) is typically bounded by an expression that depends on a capacity measure, such as the VC dimension [3] or the Fisher-Rao norm [44]. Theorem 3 provides a bound based on the effective dimension, which we use to study the power of neural networks from hereon.

In this manuscript, we consider neural networks as models described by stochastic maps, parameterized by some $\theta \in \Theta$. As a result, the variables h and $\mathcal{H}$ are replaced by θ and Θ, respectively. The corresponding loss functions are mappings $L : \mathrm{P}(\mathcal{Y}) \times \mathrm{P}(\mathcal{Y}) \to \mathbb{R}$, where $\mathrm{P}(\mathcal{Y})$ denotes the set of distributions on $\mathcal{Y}$. We assume the following regularity assumption on the model $\mathcal{M}_\Theta := \{p(\cdot, \cdot; \theta) : \theta \in \Theta\}$:

$$\Theta \ni \theta \mapsto p(\cdot, \cdot; \theta) \quad \text{is } M_1\text{-Lipschitz continuous w.r.t. the supremum norm}\,. \tag{4}$$

Theorem 3 (Generalization bound for the effective dimension)**.** *Let $\Theta = [-1, 1]^d$ and consider a statistical model $\mathcal{M}_\Theta := \{p(\cdot, \cdot; \theta) : \theta \in \Theta\}$ satisfying (4) such that the normalized Fisher information matrix $\hat{F}(\theta)$ has full rank for all $\theta \in \Theta$, and $\|\nabla_\theta \log \hat{F}(\theta)\| \leq \Lambda$ for some $\Lambda \geq 0$ and all $\theta \in \Theta$. Let $d_{\gamma,n}$ denote the effective dimension of $\mathcal{M}_\Theta$ as defined in (2). Furthermore, let $L : \mathrm{P}(\mathcal{Y}) \times \mathrm{P}(\mathcal{Y}) \to [-B/2, B/2]$ for $B > 0$ be a loss function that is α-Hölder continuous with constant M_2 in the first argument w.r.t. the total variation distance for some $\alpha \in (0, 1]$. Then there exists a constant $c_{d,\Lambda}$ such that for $\gamma \in (0, 1]$ and all $n \in \mathbb{N}$, we have*

$$\mathbb{P}\left(\sup_{\theta \in \Theta} |R(\theta) - R_n(\theta)| \geq 4M\sqrt{\frac{2\pi \log n}{\gamma n}}\right) \leq c_{d,\Lambda} \left(\frac{\gamma n^{1/\alpha}}{2\pi \log n^{1/\alpha}}\right)^{\frac{d_{\gamma,n^{1/\alpha}}}{2}} \exp\left(-\frac{16M^2\pi \log n}{B^2\gamma}\right), \tag{5}$$

where $M = M_1^\alpha M_2$.

The proof is given in Section 5.1 of the Supplementary Information. Note that the choice of the norm to bound the gradient of the Fisher information matrix is irrelevant due to the presence of the dimensional constant $c_{d,\Lambda}$. In the special case where the Fisher information matrix does not depend on θ, we have $\Lambda = 0$ and (5) holds for $c_{d,0} = 2\sqrt{d}$. This may occur in scenarios where a neural network is already trained, i.e., the parameters $\theta \in \Theta$ are fixed. If we choose $\gamma \in (0, 1]$ to be sufficiently small, we can ensure that the right-hand side of (5) vanishes in the limit $n \to \infty$. This is explained in Section 5 in the Supplementary Information. To verify the effective dimension's ability to capture generalization behavior, we conduct a numerical analysis similar to work presented in [53]. We find that the effective dimension for a model trained on confusion sets with increasing label corruption, accurately captures generalization behavior. The details can be found in Section 5.2 of the Supplementary Information.

The continuity assumptions of Theorem 3 are satisfied for a large class of classical and quantum statistical models [54,55], as well as many popular loss functions. The full rank assumption on the Fisher information matrix, however, often does not hold in classical models. Non-linear feedforward neural networks, which we consider in this study, have particularly degenerate Fisher information matrices [38]. Thus, we further extend the generalization bound to account for a broad range of models that may not have a full rank Fisher information matrix.

Remark 4 (Relaxing the rank constraint in Theorem 3)**.** The generalization bound in (5) can be modified to hold for a statistical model without a full rank Fisher information matrix. By partitioning the parameter space Θ, we discretize the statistical model and prove a generalization bound for the discretized version of $\mathcal{M}_\Theta := \{p(\cdot, \cdot; \theta) : \theta \in \Theta\}$ denoted by $\mathcal{M}_\Theta^{(\kappa)} := \{p^{(\kappa)}(\cdot, \cdot; \theta) : \theta \in \Theta\}$, where $\kappa \in \mathbb{N}$ is a discretization parameter. By choosing κ carefully, we can control the discretization error. We then proceed similarly as in the proof of Theorem 3, i.e., first connecting the generalization error to the covering number and then relating the covering number to the effective dimension. This is explained in detail, along with the proof, in Section 5.3 of the Supplementary Information.

Training the quantum neural network on real hardware

The hardware experiment is conducted on the `ibmq_montreal 27-qubit` device. We use 4 qubits with linear connectivity to train the quantum neural network on the first two classes of the `Iris` dataset. We deploy the same training specifications as in Section 3.3 of the Supplementary Information and randomly initialize the parameters. Once the training loss stabilizes, i.e. the change in the loss from one iteration to the next is small, we stop the hardware training. This occurs after roughly 33 training steps. The results are contained in Figure 3b and the real hardware shows remarkable performance relative to all other models. Due to limited hardware availability, this experiment is only run once and an analysis of the hardware noise and the spread of the training loss for differently sampled initial parameters would make these results more robust.

We plot the circuit that is implemented on the quantum device in Supplementary Figure 8. As in the quantum neural network discussed in Section 1 of the Supplementary Information, the circuit contains parameterized RZ and RZZ rotations that depend on the data, as well as parameterized RY-gates with 8 trainable parameters. Note the different entanglement structure presented here as opposed to the circuits in Supplementary Figures 1 and 2. This is to reduce the number of CNOT-gates required, in order to incorporate current hardware constraints and could be the reason the actual hardware implementation trains so well as too much entanglement has been shown to have a negative effect on model trainability [50]. The full circuit repeats the feature map encoding once before the variational form is applied.

Correspondence and requests for materials should be addressed to S.W.

Data Availability The data for the graphs and analyses in this study was generated using Python. Source data for Figures 2 and 3 are available with this manuscript. All other data can be accessed through the following repository https://doi.org/10.5281/zenodo.4732830 [56].

Code Availability All code to generate the data, figures and analyses in this study is publicly available with detailed information on the implementation via the following repository https://doi.org/10.5281/zenodo.4732830 [56].

Acknowledgements The authors thank Maria Schuld for the insightful discussions on data embedding in quantum models. The authors also thank Travis L. Scholten for constructive feedback on the manuscript. CZ acknowledges support from the National Centre of Competence in Research *Quantum Science and Technology* (QSIT).

Author contributions The main ideas were developed by all authors. A.A. provided numerical simulations. D.S. and A.F. proved the technical claims. All authors contributed to the write-up.

Competing interests The authors declare no competing interests.

References

[1] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[2] P. Baldi and R. Vershynin. The capacity of feedforward neural networks. *Neural networks*, 116:288–311, 2019. DOI: 10.1016/j.neunet.2019.04.009.

[3] G. K. Dziugaite and D. M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence*, 2017. Available online: https://arxiv.org/abs/1703.11008.

[4] M. Schuld. *Supervised learning with quantum computers*. Springer, 2018. DOI: 10.1007/978-3-319-96424-9.

[5] C. Zoufal, A. Lucchi, and S. Woerner. Quantum generative adversarial networks for learning and loading random distributions. *npj Quantum Information*, 5(1):1–9, 2019. DOI: 10.1038/s41534-019-0223-2.

[6] J. Romero, J. P. Olson, and A. Aspuru-Guzik. Quantum autoencoders for efficient compression of quantum data. *Quantum Science and Technology*, 2(4):045001, 2017. DOI: 10.1088/2058-9565/aa8072.

[7] V. Dunjko and H. J. Briegel. Machine learning & artificial intelligence in the quantum domain: a review of recent progress. *Reports on Progress in Physics*, 81(7):074001, 2018. DOI: 10.1088/1361-6633/aab406.

[8] C. Ciliberto, M. Herbster, A. D. Ialongo, M. Pontil, A. Rocchetto, S. Severini, and L. Wossnig. Quantum machine learning: a classical perspective. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474(2209):20170551, 2018. DOI: 10.1098/rspa.2017.0551.

[9] N. Killoran, T. R. Bromley, J. M. Arrazola, M. Schuld, N. Quesada, and S. Lloyd. Continuous-variable quantum neural networks. *Phys. Rev. Research*, 1:033063, 2019. DOI: 10.1103/PhysRevResearch.1.033063.

[10] M. Schuld, I. Sinayskiy, and F. Petruccione. The quest for a quantum neural network. *Quantum Information Processing*, 13(11):2567–2586, 2014. DOI: 10.1007/s11128-014-0809-8.

[11] E. Farhi and H. Neven. Classification with quantum neural networks on near term processors. *Quantum Review Letters*, 1(2), 2020. DOI: 10.37686/qrl.v1i2.80.

[12] S. Aaronson. Read the fine print. *Nature Physics*, 11(4):291–293, 2015. DOI: 10.1038/nphys3272.

[13] V. Vapnik. *The Nature of Statistical Learning Theory*, volume 8, pages 1–15. 2000. DOI: 10.1007/978-1-4757-3264-1_1.

[14] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971. DOI: 10.1137/1116025.

[15] E. D. Sontag. VC dimension of neural networks. *NATO ASI Series F Computer and Systems Sciences*, 168:69–96, 1998.

[16] V. Vapnik, E. Levin, and Y. L. Cun. Measuring the VC-dimension of a learning machine. *Neural computation*, 6(5):851–876, 1994. DOI: 10.1162/neco.1994.6.5.851.

[17] B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro. Exploring generalization in deep learning. In *Advances in neural information processing systems*, pages 5947–5956, 2017. DOI: 10.5555/3295222.3295344.

[18] S. Arora, R. Ge, B. Neyshabur, and Y. Zhang. Stronger generalization bounds for deep nets via a compression approach. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 254–263. PMLR, 2018. Available online: http://proceedings.mlr.press/v80/arora18b.html.

[19] L. G. Wright and P. L. McMahon. The capacity of quantum neural networks. In *Conference on Lasers and Electro-Optics*, page JM4G.5. Optical Society of America, 2020. Available online: http://www.osapublishing.org/abstract.cfm?URI=CLEO_QELS-2020-JM4G.5.

[20] Y. Du, M.-H. Hsieh, T. Liu, and D. Tao. Expressive power of parametrized quantum circuits. *Phys. Rev. Research*, 2:033125, 2020. DOI: 10.1103/PhysRevResearch.2.033125.

[21] H.-Y. Huang, M. Broughton, M. Mohseni, R. Babbush, S. Boixo, H. Neven, and J. R. McClean. Power of data in quantum machine learning, 2020. Available online: https://arxiv.org/abs/2011.01938.

[22] O. Berezniuk, A. Figalli, R. Ghigliazza, and K. Musaelian. A scale-dependent notion of effective dimension, 2020. Available online: https://arxiv.org/abs/2001.10872.

[23] J. J. Rissanen. Fisher information and stochastic complexity. *IEEE Transactions on Information Theory*, 42(1):40–47, 1996. DOI: 10.1109/18.481776.

[24] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley Interscience, 2006. DOI: 10.1002/047174882X.

[25] K. Nakaji and N. Yamamoto. Expressibility of the alternating layered ansatz for quantum computation. *Quantum*, 5:434, 2021. DOI: 10.22331/q-2021-04-19-434.

[26] Z. Holmes, K. Sharma, M. Cerezo, and P. J. Coles. Connecting ansatz expressibility to gradient magnitudes and barren plateaus, 2021. Available online: https://arxiv.org/abs/2101.02138.

[27] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven. Barren plateaus in quantum neural network training landscapes. *Nature communications*, 9(1):1–6, 2018. DOI: 10.1038/s41467-018-07090-4.

[28] S. Wang, E. Fontana, M. Cerezo, K. Sharma, A. Sone, L. Cincio, and P. J. Coles. Noise-induced barren plateaus in variational quantum algorithms, 2020. Available online: https://arxiv.org/abs/2007.14384.

[29] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles. Cost-function-dependent barren plateaus in shallow quantum neural networks, 2020. Available online: https://arxiv.org/abs/2001.00550.

[30] G. Verdon, M. Broughton, J. R. McClean, K. J. Sung, R. Babbush, Z. Jiang, H. Neven, and M. Mohseni. Learning to learn with quantum neural networks via classical neural networks, 2019. Available online: https://arxiv.org/abs/1907.05415.

[31] T. Volkoff and P. J. Coles. Large gradients via correlation in random parameterized quantum circuits, 2020. Available online: https://arxiv.org/abs/2005.12200.

[32] A. Skolik, J. R. McClean, M. Mohseni, P. van der Smagt, and M. Leib. Layerwise learning for quantum neural networks. *Quantum Machine Intelligence*, 3(1):5, 2021. DOI: 10.1007/s42484-020-00036-4.

[33] P. Huembeli and A. Dauphin. Characterizing the loss landscape of variational quantum circuits. *Quantum Science and Technology*, 6(2), 2021. DOI: 10.1088/2058-9565/abdbc9.

[34] C. Bishop. Exact calculation of the Hessian matrix for the multilayer perceptron, 1992. DOI: 10.1162/neco.1992.4.4.494.

[35] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. *Efficient BackProp*, pages 9–48. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. DOI: 10.1007/978-3-642-35289-8_3.

[36] M. Cerezo and P. J. Coles. Impact of barren plateaus on the Hessian and higher order derivatives, 2020. Available online: https://arxiv.org/abs/2008.07454.

[37] F. Kunstner, P. Hennig, and L. Balles. Limitations of the empirical Fisher approximation for natural gradient descent. In *Advances in Neural Information Processing Systems 32*, pages 4156–4167. 2019. http://papers.nips.cc/paper/limitations-of-fisher-approximation.

[38] R. Karakida, S. Akaho, and S.-I. Amari. Universal statistics of Fisher information in deep neural networks: Mean field approach. volume 89 of *Proceedings of Machine Learning Research*, pages 1032–1041. PMLR, 2019. Available online: http://proceedings.mlr.press/v89/karakida19a.html.

[39] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe. Circuit-centric quantum classifiers. *Phys. Rev. A*, 101(3):032308, 2020. DOI: 10.1103/PhysRevA.101.032308.

[40] M. Schuld, R. Sweke, and J. J. Meyer. Effect of data encoding on the expressive power of variational quantum-machine-learning models. *Phys. Rev. A*, 103:032430, 2021. DOI: 10.1103/PhysRevA.103.032430.

[41] S. Lloyd, M. Schuld, A. Ijaz, J. Izaac, and N. Killoran. Quantum embeddings for machine learning, 2020. Available online: https://arxiv.org/abs/2001.03622.

[42] I. Cong, S. Choi, and M. D. Lukin. Quantum convolutional neural networks. *Nature Physics*, 15(12):1273–1278, 2019. DOI: 10.1038/s41567-019-0648-8.

[43] S.-I. Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998. DOI: 10.1162/089976698300017746.

[44] T. Liang, T. Poggio, A. Rakhlin, and J. Stokes. Fisher-Rao metric, geometry, and complexity of neural networks. volume 89 of *Proceedings of Machine Learning Research*, pages 888–896. PMLR, 2019. Available online: http://proceedings.mlr.press/v89/liang19a.html.

[45] B. Neyshabur, R. R. Salakhutdinov, and N. Srebro. Path-sgd: Path-normalized optimization in deep neural networks. In *Advances in Neural Information Processing Systems*, pages 2422–2430, 2015. DOI: 10.5555/2969442.2969510.

[46] B. Neyshabur, R. Tomioka, and N. Srebro. Norm-based capacity control in neural networks. volume 40 of *Proceedings of Machine Learning Research*, pages 1376–1401, Paris, France, 2015. PMLR. Available online: http://proceedings.mlr.press/v40/Neyshabur15.html.

[47] P. L. Bartlett, D. J. Foster, and M. J. Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems 30*, pages 6240–6249. Curran Associates, Inc., 2017. http://papers.nips.cc/paper/7204-spectrally-normalized.

[48] J. J. Rissanen. Fisher information and stochastic complexity. *IEEE Transactions on Information Theory*, 42(1):40–47, 1996. DOI: 10.1109/18.481776.

[49] D. Dua and C. Graff. UCI machine learning repository, 2017. Available online: http://archive.ics.uci.edu/ml.

[50] C. O. Marrero, M. Kieferová, and N. Wiebe. Entanglement induced barren plateaus, 2020. Available online: https://arxiv.org/abs/2010.15968.

[51] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta. Supervised learning with quantum-enhanced feature spaces. *Nature*, 567(7747):209–212, 2019. DOI: 10.1038/s41586-019-0980-2.

[52] S. Sim, P. D. Johnson, and A. Aspuru-Guzik. Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms. *Advanced Quantum Technologies*, 2(12):1900070, 2019. DOI: 10.1002/qute.201900070.

[53] Z. Jia and H. Su. Information-theoretic local minima characterization and regularization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 4773–4783. PMLR, 2020. Available online: http://proceedings.mlr.press/v119/jia20a.html.

[54] A. Virmaux and K. Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. In *Advances in Neural Information Processing Systems 31*, pages 3835–3844. 2018. http://papers.nips.cc/paper/lipschitz-regularity-of-deep-neural-networks.

[55] R. Sweke, F. Wilde, J. J. Meyer, M. Schuld, P. K. Fährmann, B. Meynard-Piganeau, and J. Eisert. Stochastic gradient descent for hybrid quantum-classical optimization. *Quantum*, 4:314, 2020. DOI: 10.22331/q-2020-08-31-314.

[56] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner. Code for manuscript "The power of quantum neural networks", 2020. DOI: 10.5281/zenodo.4732830.

Supplementary Information: The power of quantum neural networks

Amira Abbas[1,2], David Sutter[1], Christa Zoufal[1,3], Aurelien Lucchi[3], Alessio Figalli[3], and Stefan Woerner[1]

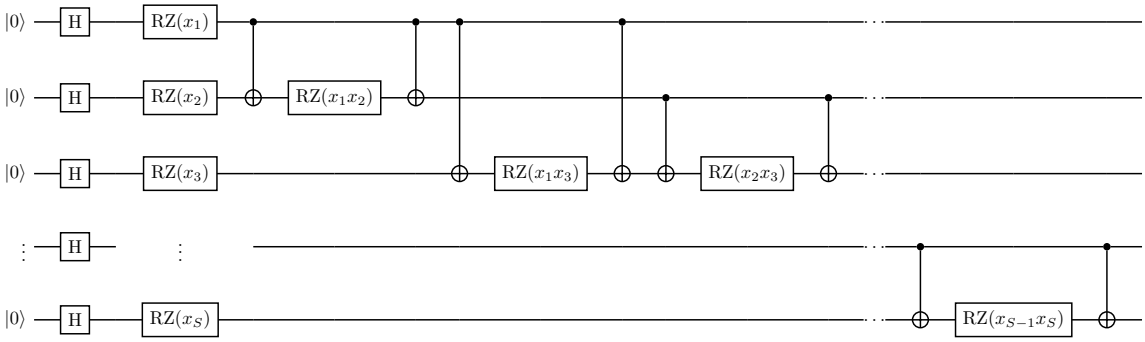
[1] *IBM Quantum, IBM Research – Zurich*
[2] *University of KwaZulu-Natal, Durban*
[3] *ETH Zurich*

1 Details of the quantum models

The quantum models considered in this study are of the form given in Figure 1 of the main manuscript. In the following, we depict the circuits used for the feature map and variational form of the quantum neural network.

1.1 The feature map

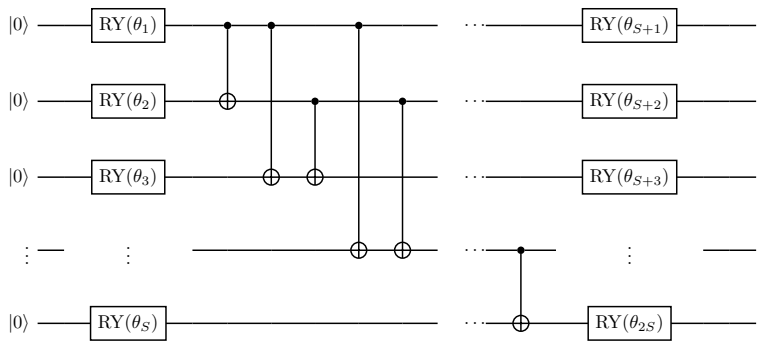
Supplementary Figure 1 contains a circuit representation of the hard feature map used in the quantum neural network. The Z-rotations are about angles which depend on the feature values of the data, x_i. A more detailed explanation is contained in the Methods section of the main manuscript.



Supplementary Figure 1: **Feature map** from [1], used in the quantum neural network. First, Hadamard gates are applied to each qubit. Then, normalized feature values of the data are encoded using RZ-gates. This is followed by CNOT-gates and higher order data encoding between every pair of qubits, and every pair of features in the data. The feature map is repeated to create a depth of 2. The easy quantum model, introduced in the results section, applies only the first sets of Hadamard and RZ-gates.

1.2 The variational form

Supplementary Figure 2 contains a circuit diagram of the variational form used in the quantum neural network, as well as the easy quantum model. There are layers of parameterized Y-rotations and all-to-all qubit entanglement. More details can be found in the Methods section of the main manuscript.



Supplementary Figure 2: **Variational circuit** used in both quantum models is plotted in this figure. The circuit contains parameterized RY-gates, followed by CNOT-gates and another set of parameterized RY-gates.

2 Properties of the effective dimension

2.1 Effective dimension converges to maximal rank of Fisher information matrix

The effective dimension converges to the maximal rank of the Fisher information matrix denoted by $\bar{r} := \max_{\theta\in\Theta} r_\theta$ in the limit $n \to \infty$. Since the Fisher information matrix is positive semidefinite, it can be unitarily diagonalized. By definition of the effective dimension, we see that, without loss of generality, $F(\theta)$ can be diagonal, i.e. $F(\theta) = \mathrm{diag}(\lambda_1(\theta), \ldots, \lambda_{r_\theta}(\theta), 0 \ldots, 0)$. Furthermore we define the normalization constant

$$\beta := d\frac{V_\Theta}{\int_\Theta \mathrm{tr}\, F(\theta)\mathrm{d}\theta}\,,$$

such that $\hat{F}(\theta) = \beta F(\theta)$. Let $\kappa_n := \frac{\gamma n}{2\pi \log n}$ and consider n to be sufficiently large such that $\kappa_n \geq 1$. By definition of the effective dimension we find

$$\begin{aligned}
d_{\gamma,n} &= 2\log\left(\frac{1}{V_\Theta}\int_\Theta \sqrt{\det(\mathrm{id}_d + \kappa_n \hat{F}(\theta))}\mathrm{d}\theta\right) / \log(\kappa_n) \\
&= 2\log\left(\frac{1}{V_\Theta}\int_\Theta \sqrt{\big(1+\kappa_n\beta\lambda_1(\theta)\big)\ldots\big(1+\kappa_n\beta\lambda_{r_\theta}(\theta)\big)}\mathrm{d}\theta\right) / \log(\kappa_n) \\
&\leq 2\log\left(\frac{1}{V_\Theta}\int_\Theta \sqrt{\kappa_n^{r_\theta}\big(1+\beta\lambda_1(\theta)\big)\ldots\big(1+\beta\lambda_{r_\theta}(\theta)\big)}\mathrm{d}\theta\right) / \log(\kappa_n) \\
&\leq 2\log\left(\frac{\kappa_n^{\bar{r}/2}}{V_\Theta}\int_\Theta \sqrt{\big(1+\beta\lambda_1(\theta)\big)\ldots\big(1+\beta\lambda_{\bar{r}}(\theta)\big)}\mathrm{d}\theta\right) / \log(\kappa_n)\,,
\end{aligned}$$

where the final step uses that the Fisher information matrix is positive definite. Taking the limit $n \to \infty$ gives

$$\lim_{n\to\infty} d_{\gamma,n} \leq \bar{r} + \lim_{n\to\infty} 2\log\left(\frac{1}{V_\Theta}\int_\Theta \sqrt{\big(1+\beta\lambda_1(\theta)\big)\ldots\big(1+\beta\lambda_{\bar{r}}(\theta)\big)}\mathrm{d}\theta\right) / \log(\kappa_n) = \bar{r}\,.$$

To see the other direction, let $\mathcal{A} := \{\theta \in \Theta : r_\theta = \bar{r}\}$ and denote its volume by $|\mathcal{A}|$. By definition of the effective dimension we obtain

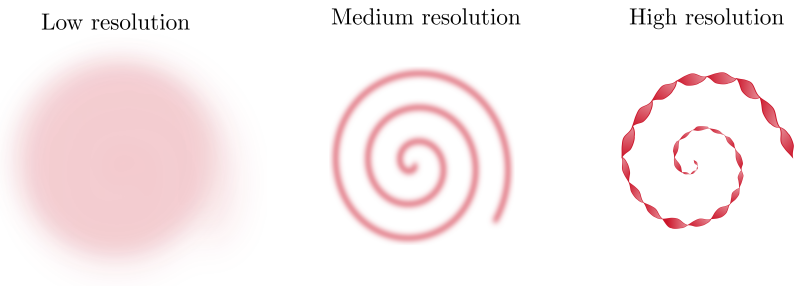
$$\begin{aligned}
\lim_{n\to\infty} d_{\gamma,n} &\geq \lim_{n\to\infty} 2\log\left(\frac{1}{V_\Theta}\int_\mathcal{A} \sqrt{\det(\mathrm{id}_d + \kappa_n \hat{F}(\theta))}\mathrm{d}\theta\right) / \log(\kappa_n) \\
&= \lim_{n\to\infty} 2\log(|\mathcal{A}|/V_\Theta) / \log(\kappa_n) + \lim_{n\to\infty} 2\log\left(\frac{1}{|\mathcal{A}|}\int_\mathcal{A} \sqrt{\det(\mathrm{id}_d + \kappa_n \hat{F}(\theta))}\mathrm{d}\theta\right) / \log(\kappa_n) \\
&\geq \lim_{n\to\infty} 2\log\left(\frac{1}{|\mathcal{A}|}\int_\mathcal{A} \sqrt{\det(\kappa_n \hat{F}(\theta))}\mathrm{d}\theta\right) / \log(\kappa_n)
\end{aligned}$$

$$= \bar{r} + \lim_{n\to\infty} 2\log\left(\frac{1}{|\mathcal{A}|}\int_{\mathcal{A}}\sqrt{\det(\hat{F}(\theta))}\mathrm{d}\theta\right)/\log(\kappa_n)$$
$$= \bar{r}\,.$$

This proves the other direction and concludes the argument.

2.2 A geometric depiction of the effective dimension

The effective dimension defined in the main document does not necessarily increase monotonically with the number of data, n. Recall that the effective dimension attempts to capture the size of a model, whilst n determines the resolution at which the model can be observed. Supplementary Figure 3 contains an intuitive example of a case where the effective dimension is not monotone in n. We can interpret a model as a geometric object. When n is small, the resolution at which we are able to see this object is very low. In this unclear, low resolution setting, the model can appear to be a 2-dimensional disk as depicted in Supplementary Figure 3. Increasing n, increases the resolution and the model can then look 1-dimensional, as seen by the spiralling line in the medium resolution regime. Going to very high resolution, and thus, very high n, reveals that the model is a 2-dimensional structure. In this example, the effective dimension will be high for small n, where the model is considered 2-dimensional, lower for slightly higher n where the model seems 1-dimensional, and high again as the number of data becomes sufficient to accurately quantify the true model size. Similar examples can be constructed in higher dimensions by taking the same object and allowing it to spiral inside the unit ball of the ambient space $\mathbb{R}^d$. Then, the effective dimension will be d for small n, it will go down to a value close to 1, and finally converge to 2 as $n \to \infty$. In all experiments conducted in this study, we examine the effective dimension over a wide range of n, to ensure it is sufficient in accurately estimating the size of a model.



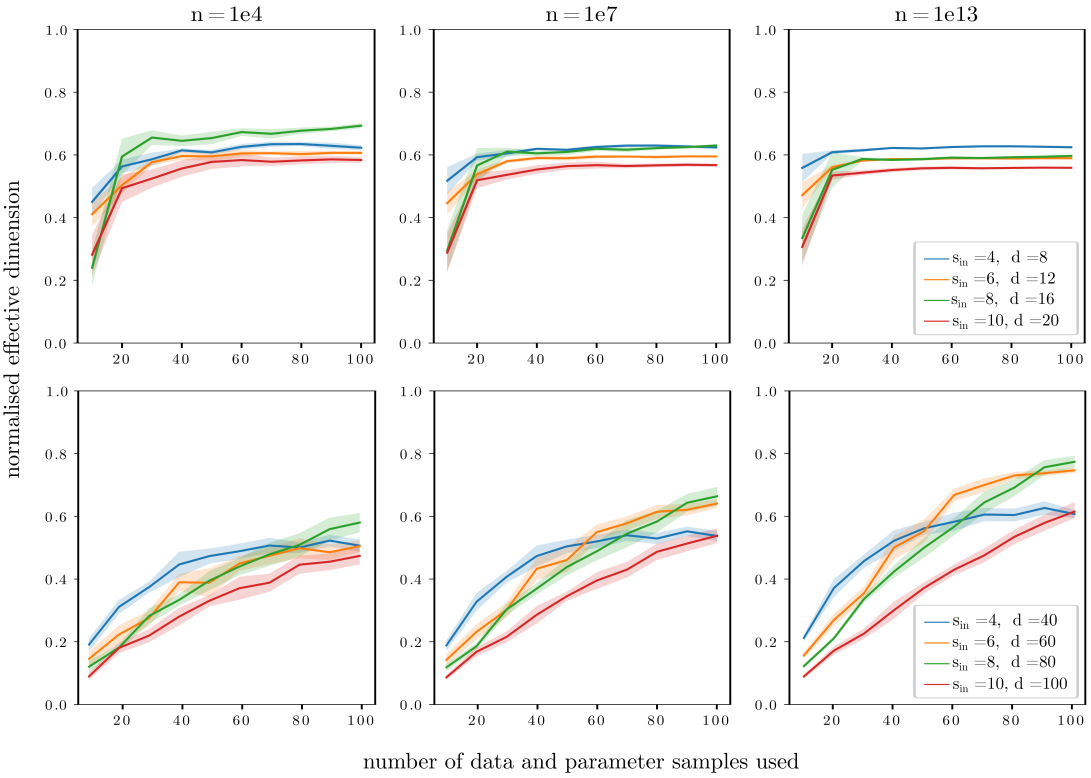
Supplementary Figure 3: **Geometric picture of a model at different resolution scales.** In the low resolution scale, the model can appear as a 2-dimensional disk and the effective dimension attempts to quantify the size of this disk. As we enhance the resolution by increasing the number of data used in the effective dimension, the medium scale reveals a 1-dimensional line, spiralling. Adding sufficient data and moving to high resolution allows the effective dimension to accurately capture the model's true size, which in this case is actually a 2-dimensional object. Thus, the effective dimension does not necessarily increase monotonically with the number of data used.

3 Numerical experiments

3.1 Sensitivity analysis for the effective dimension

We use Monte Carlo sampling to estimate the effective dimension. The capacity results will, thus, be sensitive to the number of data samples used in estimating the Fisher information matrix for a given θ, and to the number of θ samples then used to calculate the effective dimension. We plot the normalized effective dimension with n fixed, in Supplementary Figure 4 over an increasing number of data and parameter samples using the classical feedforward model. For networks with less trainable parameters, d, the results stabilize with as little as 40 data and parameter samples. When higher dimensions are considered, the standard deviation around the results increases, but

100 data and parameter samples are still reasonable given that we consider a maximum of $d = 100$. For higher d, it is likely that more samples will be needed.



Supplementary Figure 4: **Sensitivity analysis** of the normalized effective dimension to different numbers of data and parameter samples, used in calculating the empirical Fisher information matrix, and subsequently, the effective dimension.

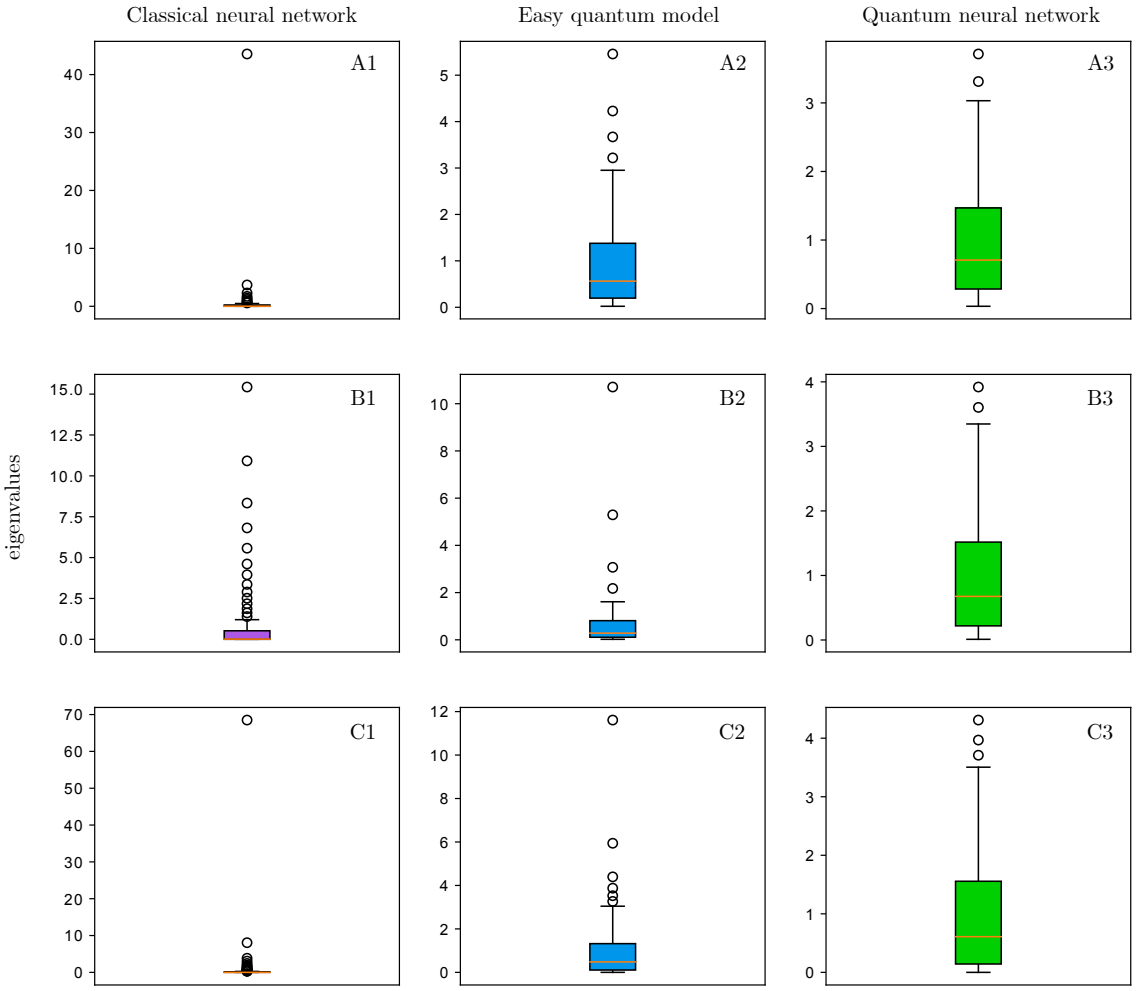
3.2 The Fisher information spectra for varying model size

Supplementary Figure 5 plots the average distribution of the Fisher information eigenvalues for all model types in box plots, over increasing input size, s_{in}, and hence, increasing number of parameters, d. The dots in the box plots represent outlier values relative to the length of the whiskers. The lower whisker is at the lowest datum above Q1 - 1.5*(Q3-Q1), and the upper whisker at the highest datum below Q3 + 1.5*(Q3-Q1), where Q1 and Q3 are the first and third quartiles. This is a standard method to compute these plots.

These average distributions are generated using 100 Fisher information matrices with parameters, θ, drawn uniformly at random on $\Theta = [-1, 1]^d$. Row A contains the distribution of eigenvalues for models with $s_{\text{in}} = 6$, row B for $s_{\text{in}} = 8$ and row C for $s_{\text{in}} = 10$. In all scenarios, the classical model has a majority of its eigenvalues near or equal to zero, with a few very large eigenvalues. The easy quantum model has a somewhat uniform spectrum for a smaller input size, but this deteriorates as the input size (also equal to the number of qubits in this particular model) increases. The quantum neural network, however, maintains a more uniform spectrum over increasing s_{in} and d, showing promise in avoiding unfavorable qualities, such as barren plateaus.

3.2.1 The Fisher information spectra with hardware noise

Similar to Supplementary Figure 5, Supplementary Figure 6 contains the average distribution of the Fisher information eigenvalues where the outputs of both quantum models are attained from a a shot-based simulation of the `ibmq_montreal 27-qubit` chip which includes a model for the underlying physical noise. Row A contains models with $s_{\text{in}} = 4$, row B with $s_{\text{in}} = 6$, row C

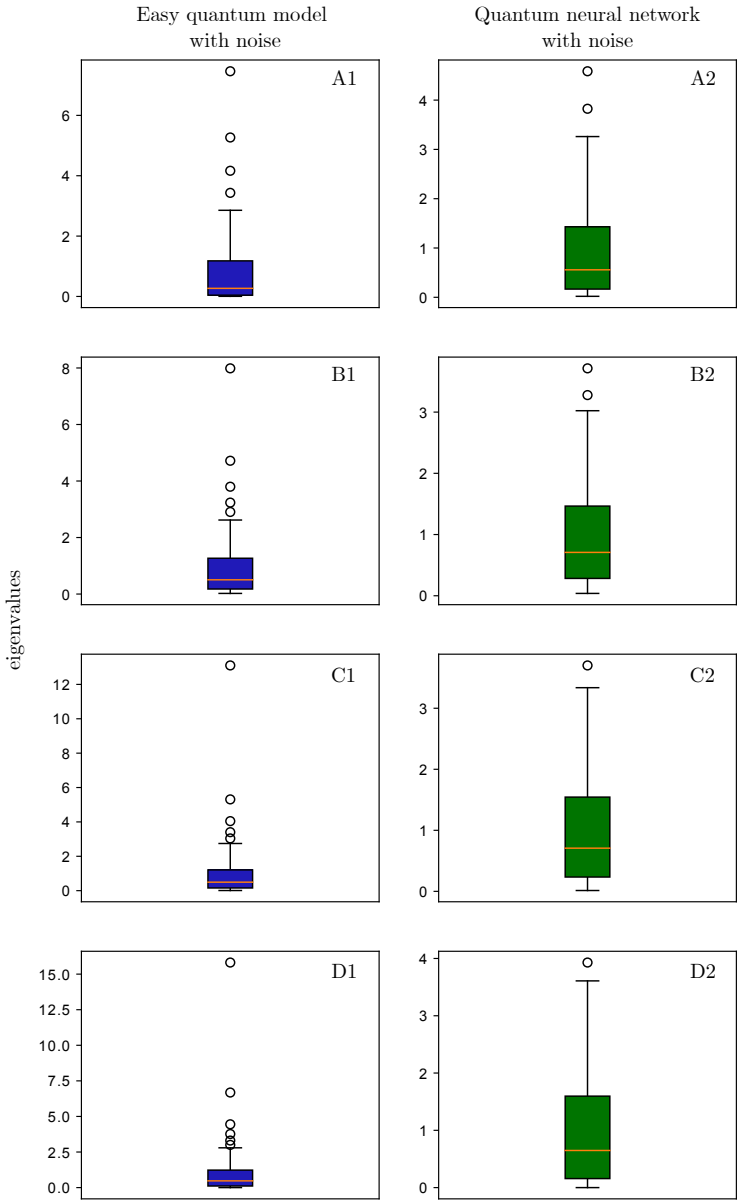


Supplementary Figure 5: **Average Fisher information spectrum** depicted as a boxplot plot for all three model types, over increasing input size, s_{in}. Row A contains models with $s_{\text{in}} = 6$ and $d = 60$, row B has $s_{\text{in}} = 8$ and $d = 80$ and row C has $s_{\text{in}} = 10$ and $d = 100$. In all cases, $s_{\text{out}} = 2$.

with $s_{\text{in}} = 8$ and row D with $s_{\text{in}} = 10$. The effect of hardware noise does not seem to change the eigenvalue distributions significantly. The easy quantum model continues to display uneven spectra as the model's size increases, whereas the quantum neural network's spectra remains stable.

3.3 Training the models using a simulator

To test the trainability of all three model types, we conduct a simple experiment using the `Iris` dataset. In each model, we use an input size of $s_{\text{in}} = 4$, output size $s_{\text{out}} = 2$ and $d = 8$ trainable parameters. We train the models for 100 training iterations, using 100 data points from the first two classes of the dataset. Standard hyperparameter choices are made, using an initial learning rate = 0.1 and the `ADAM` optimizer. Each model is trained 100 times, with initial parameters θ sampled uniformly on $\Theta = [-1, 1]^d$ each trial. We choose $\Theta = [-1, 1]^d$ as the sample space for the initial parameters, as well as for the parameter sample space in the effective dimension. Another convention is to use $[-2\pi, 2\pi]^d$ as the parameter space for initialization of the quantum model, however, we stick with $[-1, 1]^d$ to be consistent and align with classical neural network literature. We note that for the effective dimension, using either parameter space does affect the observed results. The average training loss and average Fisher-Rao norm after 100 training iterations, is captured in Supplementary Table 1. The quantum neural network notably has the highest Fisher-Rao norm and lowest training loss on average.



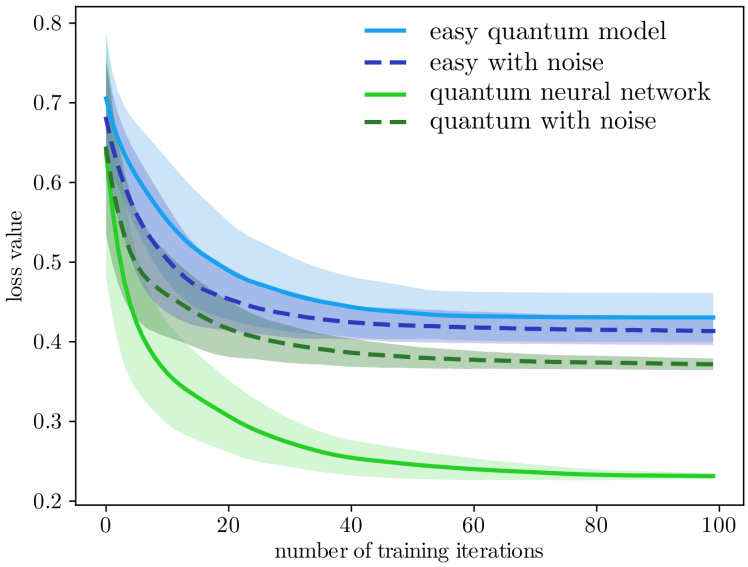
Supplementary Figure 6: **Average Fisher information spectrum** for the easy quantum model and the quantum neural network with hardware noise. Row A contains models with $s_{\text{in}} = 4$ and $d = 40$, row B with $s_{\text{in}} = 6$ and $d = 60$, row C with $s_{\text{in}} = 8$ and $d = 80$ and row D with $s_{\text{in}} = 10$ and $d = 100$.

Model	Training loss	Fisher-Rao norm
Classical neural network	37.90%	46.45
Easy quantum model	43.05%	104.89
Quantum neural network	23.14%	117.84

Supplementary Table 1: **Average training loss** and **average Fisher-Rao norm** for all three models, using 100 different trials with 100 training iterations.

3.3.1 Training with hardware noise

Training quantum models can be drastically effected by the impact of hardware noise. In order to explore this further, we investigate the training performance under noisy hardware conditions for the quantum neural network and the easy quantum model using a shot-based simulation of the `ibmq_montreal 27-qubit` device and plot the results in Supplementary Figure 7. As one would expect, the introduction of noise slows down the loss convergence for the quantum neural network. Interestingly, the easy quantum model's training performance improves when noise is added. The quantum neural network with noise still outperforms the easy quantum model with/without noise. But in both noisy situations, the models struggle to train using the `ADAM` optimizer.



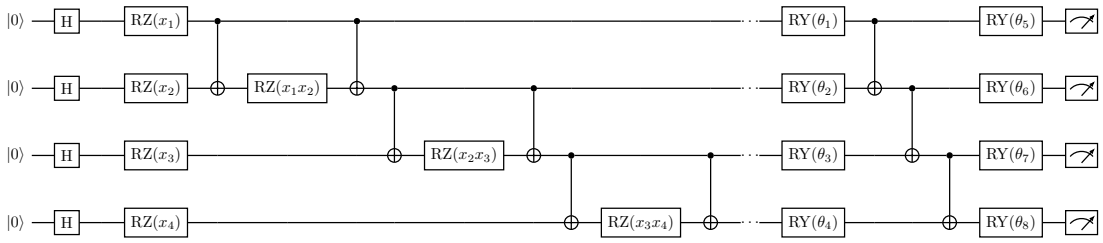
Supplementary Figure 7: **Training with hardware noise** using a cross entropy loss function and the `ADAM` optimizer. As expected, the quantum neural network's performance is negatively impacted by hardware noise. On the other hand, noise seems to improve the training performance of the easy quantum model, but overall, the quantum neural network (even with noise) performs better than the easy quantum model with and without noise.

3.4 Training the quantum neural network on real hardware

The hardware experiment is conducted on the `ibmq_montreal 27-qubit` device. We use 4 qubits with linear connectivity to train the quantum neural network on the first two classes of the `Iris` dataset and plot the circuit in Supplementary Figure 8.

4 The Fisher spectrum and the barren plateau phenomenon

The Fisher information spectrum for fully connected feedforward neural networks reveals that the parameter space is flat in most dimensions, and strongly distorted in a few others [2]. These distortions are captured by a few very large eigenvalues, whilst the flatness corresponds to eigenvalues being close to zero. This behavior has also been reported for the Hessian matrix, which coincides with the Fisher information matrix under certain conditions, e.g., under the use of certain loss functions [3–5]. These types of spectra are known to slow down a model's training and may render optimization suboptimal [6]. In the quantum realm, the negative effect of barren plateaus on training quantum neural networks has been linked to the Hessian matrix [7]. It was found that the entries of the Hessian vanish exponentially with the size of the system in models that are in



Supplementary Figure 8: **Circuit implemented on quantum hardware.** First, Hadamard gates are applied. Then the data is encoded using RZ-gates applied to each qubit whereby the Z-rotations depend on the feature values of the data. Thereafter, CNOT entangling layers with RZ-gates encoding products of feature values are applied. The data encoding gates, along with the CNOT-gates are repeated to create a depth 2 feature map. Lastly, parameterized RY-gates are applied to each qubit followed by linear entanglement and a final layer of parameterized RY-gates. The circuit has a total of 8 trainable parameters.

a barren plateau. This implies that the loss landscape becomes increasingly flat as the size of the model increases, making optimization more difficult.

The Fisher information can also be connected to barren plateaus. Assuming a log-likelihood loss function, without loss of generality, we can formulate the empirical risk over the full training set as

$$R_n(\theta) = -\frac{1}{n}\log\Big(\prod_{i=1}^{n} p(y_i|x_i;\theta)\Big) = -\frac{1}{n}\sum_{i=1}^{n}\log p(y_i|x_i;\theta),$$

where $p(y_i|x_i;\theta)$ is the conditional distribution for a data pair (x_i, y_i). From Bayes rule, note that the derivative of the empirical risk function is then equal to the derivative of the log of the joint distribution summed over all data pairs, i.e.,

$$\frac{\partial}{\partial\theta}R_n(\theta) = -\frac{\partial}{\partial\theta}\frac{1}{n}\sum_{i=1}^{n}\log p(y_i|x_i;\theta) = -\frac{\partial}{\partial\theta}\frac{1}{n}\sum_{i=1}^{n}\log p(x_i,y_i;\theta) = -\frac{1}{n}\sum_{i=1}^{n}\frac{\partial}{\partial\theta}\log p(x_i,y_i;\theta)\,,$$

since the prior distribution $p(\cdot)$ does not depend on θ. From [8], we know that we are in a barren plateau if, for parameters θ uniformly sampled from Θ, each element of the gradient of the loss function with respect to θ vanishes exponentially in the number of qubits, S. In mathematical terms this means

$$\left|\mathbb{E}_\theta\Big[\frac{\partial}{\partial\theta_j}R_n(\theta)\Big]\right| = \left|\mathbb{E}_\theta\Big[\frac{1}{n}\sum_{i=1}^{n}\frac{\partial}{\partial\theta_j}\log p(x_i,y_i;\theta)\Big]\right| = \left|\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_\theta\Big[\frac{\partial}{\partial\theta_j}\log p(x_i,y_i;\theta)\Big]\right| \leq \omega_S\,,$$

for all $j = 1, \dots d$ and for some nonnegative constant ω_S that goes to zero exponentially fast with increasing S. The barren plateau result also tells us that $\mathrm{Var}_\theta[\frac{\partial}{\partial\theta_j}R_n(\theta)] \leq \omega_S$ for models in a barren plateau. By definition of the empirical Fisher information (see main document), the entries of the Fisher matrix can be written as

$$F(\theta)_{jk} = \frac{\partial}{\partial\theta_j}R_n(\theta)\frac{\partial}{\partial\theta_k}R_n(\theta)\,,$$

for $j, k = 1, \dots, d$. Hence we can write

$$\mathbb{E}_\theta[F(\theta)_{jj}] = \mathbb{E}_\theta\left[\left(\frac{\partial}{\partial\theta_j}R_n(\theta)\right)^2\right] = \mathrm{Var}_\theta\left[\frac{\partial}{\partial\theta_j}R_n(\theta)\right] + \left(\mathbb{E}_\theta\left[\frac{\partial}{\partial\theta_j}R_n(\theta)\right]\right)^2 \leq \omega_S + \omega_S^2\,,$$

which implies $\mathrm{tr}(\mathbb{E}_\theta[F(\theta)]) \leq d(\omega_S + \omega_S^2)$. Due to the positive semidefinite nature of the Fisher information matrix and by definition of the Hilbert-Schimdt norm, all matrix entries will approach zero if a model is in a barren plateau, and natural gradient optimization techniques become unfeasible. We can conclude that a model suffering from a barren plateau will have a Fisher information

spectrum with an increasing concentration of eigenvalues approaching zero as the number of qubits in the model increase. Conversely, a model with a Fisher information spectrum that is not concentrated around zero is unlikely to experience a barren plateau.

5 Generalization properties of the effective dimension

5.1 Proof of generalisation bound

Here we provide a proof for the generalisation bound stated as a theorem in the main document. Given a positive definite matrix $A \geq 0$, and a function $g : \mathbb{R}^+ \to \mathbb{R}^+$, we define $g(A)$ as the matrix obtained by taking the image of the eigenvalues of A under the map g. In other words, $A = U^\dagger \mathrm{diag}(\mu_1, \dots, \mu_d)U$ implies $g(A) = U^\dagger \mathrm{diag}(g(\mu_1), \dots, g(\mu_d))U$. To prove the assertion of the theorem, we start with a lemma that relates the effective dimension to the covering number.

Lemma 1. *Let $\Theta = [-1, 1]^d$, and let $\mathcal{N}(\varepsilon)$ denote the number of boxes of side length ε required to cover the parameter set Θ, the length being measured with respect to the metric $\hat{F}_{ij}(\theta)$. Under the assumption of the theorem (see main document), there exists a dimensional constant $c_d < \infty$ such that for $\gamma \in (0, 1]$ and for all $n \in \mathbb{N}$, we have*

$$\mathcal{N}\left(\sqrt{\frac{2\pi \log n}{\gamma n}}\right) \leq c_d \left(\frac{\gamma n}{2\pi \log n}\right)^{d_{\gamma,n}/2} .$$

Proof. The result follows from the arguments presented in [9]. More precisely, thanks to the bound $\|\nabla_\theta \log \hat{F}(\theta)\| \leq \Lambda$, which holds by assumption, it follows that

$$\|\hat{F}(\theta) - \hat{F}(0)\| \leq c_d \Lambda \|\hat{F}(0)\| \qquad \text{and} \qquad \|\hat{F}(\theta) - \hat{F}(0)\| \leq c_d \Lambda \|\hat{F}(\theta)\| \qquad \forall\, \theta \in \Theta\,. \tag{1}$$

In the following, we set $\varepsilon := \sqrt{2\pi \log n/(\gamma n)}$. Note that, if $\mathcal{B}_\varepsilon(\bar{\theta}_k)$ is a box centered at $\bar{\theta}_k \in \Theta$ and of length ε (the length being measured with respect to the metric $\hat{F}_{ij}$), then this box contains $(1 + c_d\Lambda)^{-1/2}\mathcal{B}_\varepsilon(\bar{\theta}_k)$, where

$$\mathcal{B}_\varepsilon(\bar{\theta}_k) := \{\theta \in \Theta \,:\, \langle \hat{F}(0) \cdot (\theta - \bar{\theta}_k), \theta - \bar{\theta}_k\rangle \leq \varepsilon^2\}\,.$$

Up to a rotation, we can diagonalize the Fisher information matrix as $\hat{F}(0) = \mathrm{diag}(s_1^2, \dots, s_d^2)$. Then, we see that n the number of boxes of the form $(1 + c_d\Lambda)^{-1/2}\mathcal{B}_\varepsilon(\bar{\theta}_k)$ needed to cover Θ is given by

$$\begin{aligned}
\hat{c}_d(1 + c_d\Lambda)^{d/2} \prod_{i=1}^{d} \lceil \varepsilon^{-1} s_i \rceil &\leq \hat{c}_d(1 + c_d\Lambda)^{d/2} \sqrt{\prod_{i=1}^{d} \left(1 + \varepsilon^{-2} s_i^2\right)} \\
&= \hat{c}_d(1 + c_d\Lambda)^{d/2} \sqrt{\det\left(\mathrm{id}_d + \frac{\gamma n}{2\pi \log n} \hat{F}(0)\right)} \\
&\leq \hat{c}_d(1 + c_d\Lambda)^{d/2} \sqrt{\det\left(\mathrm{id}_d + \frac{\gamma n}{2\pi \log n} (1 + c_d\Lambda)\hat{F}(\theta)\right)} \\
&\leq \hat{c}_d(1 + c_d\Lambda)^{d} \sqrt{\det\left(\mathrm{id}_d + \frac{\gamma n}{2\pi \log n} \hat{F}(\theta)\right)},
\end{aligned}$$

where the second inequality follows from (1) and the fact that the determinant is operator monotone on the set of positive definite matrices, i.e., $0 \leq A \leq B$ implies $\det(A) \leq \det(B)$ [10, Exercise 12 in Section 82]. Here $\hat{c}_d$ depends on the orientation of $[-1, 1]^d$ with respect to the boxes $\mathcal{B}_\varepsilon(\bar{\theta}_k)$. In particular $\hat{c}_d \leq 2\sqrt{d}$ (the length of the diagonal of $[-1, 1]^d$), and if the boxes $\mathcal{B}_\varepsilon(\bar{\theta}_k)$ are aligned along the canonical axes, then $\hat{c}_d = 2$.

Since the number of boxes of size ε (with respect to the metric $\hat{F}_{ij}$) needed to cover Θ is bounded by the number of boxes of the form $(1 + c_d\Lambda)^{-1/2}\mathcal{B}_\varepsilon(\bar{\theta}_k)$, averaging the bound above with

respect to $\theta \in \Theta$ we proved that

$$\mathcal{N}(\varepsilon) \leq \hat{c}_d(1 + c_d\Lambda)^d \frac{1}{V_\Theta} \int_\Theta \sqrt{\det\left(\mathrm{id}_d + \frac{\gamma n}{2\pi \log n} \hat{F}(\theta)\right)} \mathrm{d}\theta \,,$$

which implies the inequality in the statement of Lemma 1 by recalling the definition of the effective dimension and $\varepsilon := \sqrt{2\pi \log n/(\gamma n)}$. □

Lemma 2. *Let $\varepsilon \in (0, 1)$. Under the assumption of the theorem (see main document), we have*

$$\mathbb{P}\left(\sup_{\theta\in\Theta} |R(\theta) - R_n(\theta)| \geq \varepsilon\right) \leq 2\mathcal{N}\left(\left(\frac{\varepsilon}{4M}\right)^{1/\alpha}\right) \exp\left(-\frac{n\varepsilon^2}{2B^2}\right) ,$$

where $\mathcal{N}(\varepsilon)$ denotes the number of balls of side length ε, with respect to $\hat{F}$, required to cover the parameter set Θ.

Proof. The proof is a slight generalization of a result found in [11, Chapter 3]. Let $S(\theta) := R(\theta) - R_n(\theta)$. Then

$$|S(\theta_1) - S(\theta_2)| \leq |R(\theta_1) - R(\theta_2)| + |R_n(\theta_1) - R_n(\theta_2)| \leq 2M\|\theta_1 - \theta_2\|_\infty^\alpha \,, \tag{2}$$

where the final step uses the fact that $R(\cdot)$ as well as $R_n(\cdot)$ are α-Hölder continuous with constant M for $M = M_1^\alpha M_2$. To see this recall that by definition of the risk, we find for the observed input and output distributions $r \in \mathrm{P}(\mathcal{X})$ and $q \in \mathrm{P}(\mathcal{Y})$, respectively,

$$\begin{aligned}
|R(\theta_1) - R(\theta_2)| &= \left|\mathbb{E}_{r,q}\Big[L\big(p(y|x;\theta_1)r(x), q(y)\big)\Big] - \mathbb{E}_{r,q}\Big[L\big(p(y|x;\theta_2)r(x), q(y)\big)\Big]\right| \\
&\leq \mathbb{E}_{r,q}\Big[\big|L\big(p(y|x;\theta_1)r(x), q(y)\big) - L\big(p(y|x;\theta_2)r(x), q(y)\big)\big|\Big] \\
&\leq M_2\mathbb{E}_r\big[\,\|p(y|x;\theta_1)r(x) - p(y|x;\theta_2)r(x)\|_1^\alpha\,\big] \\
&\leq M_2\,\|p(y|x;\theta_1) - p(y|x;\theta_2)\|_\infty^\alpha\,\mathbb{E}_r\big[\,\|r(x)\|_1^\alpha\,\big] \\
&= M_2\,\|p(y|x;\theta_1) - p(y|x;\theta_2)\|_\infty^\alpha \\
&\leq M_2 M_1^\alpha\,\|\theta_1 - \theta_2\|_\infty^\alpha \,,
\end{aligned}$$

where the third step uses the continuity assumption of the loss function and the fourth step follows from Hölder's inequality. The final step uses the Lipschitz continuity assumption of the model. Equivalently we see that

$$|R_n(\theta_1) - R_n(\theta_2)| \leq M_2 M_1^\alpha\,\|\theta_1 - \theta_2\|_\infty^\alpha \,.$$

Assume that Θ can be covered by k subsets $B_1, \ldots, B_k$, i.e. $\Theta = B_1 \cup \ldots \cup B_k$. Then, for any $\varepsilon > 0$,

$$\mathbb{P}\left(\sup_{\theta\in\Theta} |S(\theta)| \geq \varepsilon\right) = \mathbb{P}\left(\bigcup_{i=1}^k \sup_{\theta\in B_i} |S(\theta)| \geq \varepsilon\right) \leq \sum_{i=1}^k \mathbb{P}\left(\sup_{\theta\in B_i} |S(\theta)| \geq \varepsilon\right) , \tag{3}$$

where the inequality is due to the union bound. Finally, let $k = \mathcal{N}((\frac{\varepsilon}{4M})^{1/\alpha})$ and let $B_1, \ldots, B_k$ be balls of radius $(\frac{\varepsilon}{4M})^{1/\alpha}$ centered at $\theta_1, \ldots, \theta_k$ covering Θ. Then the following inequality holds for all $i = 1, \ldots, k$,

$$\mathbb{P}\left(\sup_{\theta\in B_i} |S(\theta)| \geq \varepsilon\right) \leq \mathbb{P}\left(|S(\theta_i)| \geq \frac{\varepsilon}{2}\right) . \tag{4}$$

To prove (4), observe that by using (2) we have for any $\theta \in B_i$,

$$|S(\theta) - S(\theta_i)| \leq 2M\|\theta - \theta_i\|_\infty^\alpha \leq \frac{\varepsilon}{2} \,.$$

The last inequality implies that, if $|S(\theta)| \geq \varepsilon$, it must be that $|S(\theta_i)| \geq \frac{\varepsilon}{2}$. This in turns implies (4).

To conclude, we apply Hoeffding's inequality, which yields

$$\mathbb{P}\left(|S(\theta_i)| \geq \frac{\varepsilon}{2}\right) = \mathbb{P}\left(|R(\theta_i) - R_n(\theta_i)| \geq \frac{\varepsilon}{2}\right) \leq 2\exp\left(\frac{-n\varepsilon^2}{2B^2}\right). \tag{5}$$

Combined with (3), we obtain

$$\begin{aligned}\mathbb{P}\left(\sup_{\theta\in\Theta}|S(\theta)| \geq \varepsilon\right) &\leq \sum_{i=1}^{k}\mathbb{P}\left(\sup_{\theta\in B_i}|S(\theta)| \geq \varepsilon\right)\\ &\leq \sum_{i=1}^{k}\mathbb{P}\left(|S(\theta_i)| \geq \frac{\varepsilon}{2}\right)\\ &\leq 2\mathcal{N}\left(\left(\frac{\varepsilon}{4M}\right)^{1/\alpha}\right)\exp\left(\frac{-n\varepsilon^2}{2B^2}\right),\end{aligned}$$

where the second step uses (4). The final step follows from (5) and by recalling that $k = \mathcal{N}((\frac{\varepsilon}{4M})^{1/\alpha})$. □

Having Lemma 1 and Lemma 2 at hand we are ready to prove the assertion of the theorem stated in the main document. Lemma 2 implies for $\varepsilon = 4M\sqrt{2\pi\log n/(\gamma n)}$

$$\begin{aligned}&\mathbb{P}\left(\sup_{\theta\in\Theta}|R(\theta) - R_n(\theta)| \geq 4M\sqrt{2\pi\log n/(\gamma n)}\right)\\ &\qquad\leq 2\,\mathcal{N}\left(\left(\frac{2\pi\log n}{\gamma n}\right)^{\frac{1}{2\alpha}}\right)\exp\left(-\frac{16M^2\pi\log n}{B^2\gamma}\right)\\ &\qquad\leq 2\,\mathcal{N}\left(\left(\frac{2\pi\log n^{1/\alpha}}{\gamma n^{1/\alpha}}\right)^{\frac{1}{2}}\right)\exp\left(-\frac{16M^2\pi\log n}{B^2\gamma}\right)\\ &\qquad\leq 4c_d\left(\frac{\gamma n^{1/\alpha}}{2\pi\log n^{1/\alpha}}\right)^{\frac{d_{\gamma,n^{1/\alpha}}}{2}}\exp\left(-\frac{16M^2\pi\log n}{B^2\gamma}\right),\end{aligned} \tag{6}$$

where the penultimate step uses

$$\left(\frac{2\pi\log n}{\gamma n}\right)^{\frac{1}{2\alpha}} \geq \left(\frac{2\pi\log n^{1/\alpha}}{\gamma n^{1/\alpha}}\right)^{\frac{1}{2}},$$

for all $\lambda \in (0,1]$ and $\alpha \in (0,1]$. The final step in (6) uses Lemma 1. □

Remark 3 (Choice of γ parameter)**.** As mentioned in the main text the parameter $\gamma \in (0,1]$ needs to be chosen sufficiently small to ensure that the right-hand side of the generalisation bound vanishes in the limit $n \to \infty$. A simple calculation reveals that this occurs if γ scales at most as $\gamma \sim 32\pi\alpha M^2/(dB^2)$. To see this, we use the fact that $d_{\gamma,n} \leq d + \tau/|\log n|$ for some constant $\tau > 0$.

Remark 4 (Improved scaling for relative entropy loss function)**.** The relative entropy is commonly used as a loss function. Note that the relative entropy is log-Lipschitz in the first argument which is better than Hölder continuous. Recall that the function $f(t) = t\log(t)$ is log-Lipschitz with constant 1, i.e., $|f(t) - f(s)| \leq |t-s|\log(|t-s|)$ for $|t-s| \leq 1/\mathrm{e}$. As a result we can improve the bound from Lemma 2 to

$$\mathbb{P}\left(\sup_{\theta\in\Theta}|R(\theta) - R_n(\theta)| \geq \varepsilon\right) \leq 2\,\mathcal{N}\left(\frac{\varepsilon/(4M)}{|\log(\varepsilon/4)|}\right)\exp\left(-\frac{n\varepsilon^2}{2B^2}\right),$$

by following the proof given above and utilizing the log-Lipschitz property of the relative entropy in its first argument and the fact that the inverse of $t|\log(t)|$ behaves like $s/|\log(s)|$ near the origin. More precisely we can choose $k = \mathcal{N}(\frac{\varepsilon/(4M)}{|\log(\varepsilon/4)|})$ in the proof above.

Remark 5 (Boundedness assumption of loss function)**.** By utilizing a stronger concentration bound than Hoeffding's inequality in (5), one may be able to relax the assumption that the loss function in the generalisation bound has to be bounded.

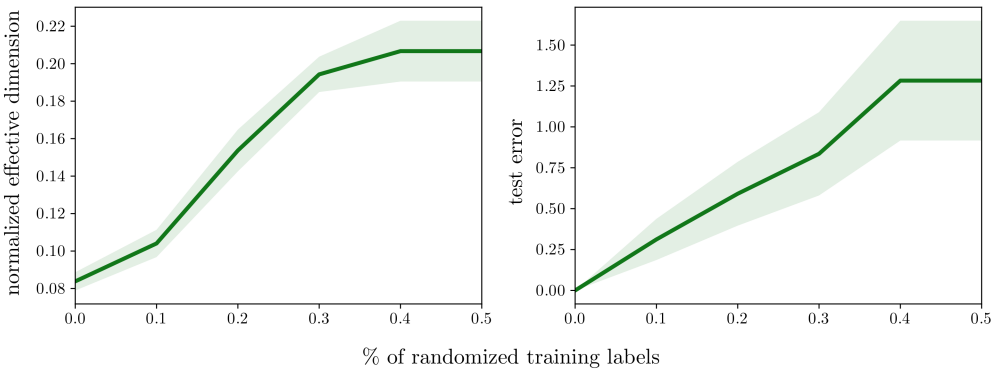
5.2 Generalization ability of the effective dimension

In order to assess the effective dimension's ability to capture generalization behavior, we conduct a numerical experiment originally introduced in [12]. The authors argue that a capacity measure is able to capture generalization behavior across local minima, if it is positively correlated with the test error on a dataset. To demonstrate this, we fix the architecture of a classical neural network model where we use a feedforward network with a single hidden layer, an input size of $s_{\text{in}} = 6$, output size $s_{\text{out}} = 2$ and number of trainable weights $d = 880$, and train it on multiple confusion sets constructed from scikit-learn's `make blobs` dataset [13]. A confusion set is a dataset where the training labels are randomized to some degree.

We first generate 1000 data points and separate them into training and test datasets using an $80\% - 20\%$ split. Then, we create confusion sets by gradually increasing the entropy in the training data through randomizing the labels in increments of 10%. Using a cross entropy loss function, we train the neural network to zero error on the original training dataset (i.e. with no randomization). Then we calculate the effective dimension using the trained parameters, as well as the test error on the test dataset. Next, we repeat this process by training the same model to zero error on each of the confusion sets and calculate the resulting effective dimension and test error.

We plot the normalized effective dimension and test error over the increasing degree of randomization in Supplementary Figure 9. As we would expect, the test error (also referred to as the generalization error) increases as the randomization in the training set increases. This is because the model is being trained on data that is increasingly noisy and eventually starts to overfit the data (i.e. starts to fit the noise). Thus, the model's ability to accurately classify the test data diminishes and it's generalization error increases.

Similarly in that regard, if a proposed capacity measure accurately captures generalization ability, we would expect to see an increasing capacity as the percentage of randomized labels in the confusion set increases. Intuitively, this is because the model requires more expressive power to fit these random labels which leads to overfitting and a higher generalization error. The effective dimension displays exactly this behavior, increasing as the entropy in the training data increases and in line with the generalization error. This experiment nicely motivates that the effective dimension can be used to capture a model's generalization ability.



Supplementary Figure 9: **Generalization behavior** of the effective dimension. On the left, we plot the normalized effective dimension for the same network trained on confusion sets with increasing randomization, averaged over 10 different training runs, with one standard deviation above and below the mean. The effective dimension correctly increases as the data becomes "more random" and is thus, able to accurately capture a model's generalization behavior. The test error, also referred to as generalization error, is plotted on the right. As the noise in the data increases, we see the generalization error increasing as the model starts to overfit (i.e. fit the noise).

5.3 Removing the rank constraint via discretization

The aim of this section is to find a suitable generalization of the results in Section 5.1 when the Fisher information matrix does not satisfy the bound $||\nabla_\theta \log \hat{F}|| \leq \Lambda$. Indeed, this is a rather strong bound as it forces $\hat{F}$ to have constant rank, so it is desirable to find a variant of Lemmas 1 and 2 that do not require such an assumption.

Our approach to this general problem is based on the idea that, in practical applications, the Fisher matrix is evaluated at finitely many points, so it makes sense to approximate a statistical model with a discretized one where the corresponding Fisher information matrix is piecewise constant.

Let $\Theta = [-1,1]^d$ and consider a statistical model $\mathcal{M}_\Theta := \{p(\cdot,\cdot;\theta) : \theta \in \Theta\}$ with a Fisher information matrix denoted by $F(\theta)$ for $\theta \in \Theta$. Given an integer $\kappa > 1$, we consider a discretized version of the statistical model. More precisely, we split Θ into κ^d disjoint cubes $\{G_i\}_{i=1}^{\kappa^d}$ of size $2/\kappa$. Then, given one of these small cubes G_i, we consider its center x_i and we split G_i into 2^d disjoint simplices, where each simplex is generated by x_i and one of the faces of ∂G_i. We denote the set of all these simplices by $\{\Theta_\ell\}_{\ell=1}^m$, where $m = 2^d\kappa^d$. Note that $\{\Theta_\ell\}_{\ell=1}^m$ is a regular triangulation of Θ.

Now, let $\mathcal{M}_\Theta^{(\kappa)} := \{p^{(\kappa)}(\cdot,\cdot;\theta) : \theta \in \Theta\}$ be a discretized version of $\mathcal{M}_\Theta$ such that $p^{(\kappa)}$ is affine on each simplex Θ_ℓ. For this, it suffices to define $p^{(\kappa)}(\cdot,\cdot;\theta) = p(\cdot,\cdot;\theta)$ whenever θ coincides with one of the vertices of Θ_ℓ for some ℓ, and then one extends $p^{(\kappa)}$ inside each simplex Θ_ℓ as an affine function. Note that, with this definition, the Fisher information matrix of the discretized model $F^{(\kappa)}(\theta)$ is constant inside each simplex Θ_ℓ. We note that, by construction, $\theta \mapsto p^{(\kappa)}(\cdot,\cdot;\theta)$ is still M_1-Lipschitz continuous. Indeed, recall that we defined $p^{(\kappa)} = p$ on the vertices of the simplices and then we extended $p^{(\kappa)}$ as an affine function inside each simplex. With this construction, the Lipschitz constant of $p^{(\kappa)}$ is bounded by the Lipschitz constant of p (since the affine extension does not increase the Lipschitz constant).

The risk function with respect to the discretized model is denoted by $R^{(\kappa)}$.

Theorem 6 (Generalization bound for effective dimension without rank constraint)**.** *Let $\Theta = [-1,1]^d$ and consider a statistical model $\mathcal{M}_\Theta := \{p(\cdot,\cdot;\theta) : \theta \in \Theta\}$ satisfying Equation (4) from the main document. For $\kappa \in \mathbb{N}$, let $\mathcal{M}_\Theta^{(\kappa)} := \{p^{(\kappa)}(\cdot,\cdot;\theta) : \theta \in \Theta\}$ be the discretized form as described above. Let $d_{\gamma,n}^{(\kappa)}$ denote the effective dimension of $\mathcal{M}_\Theta^{(\kappa)}$ as defined in the main document. Furthermore, let $L : \mathrm{P}(\mathcal{Y}) \times \mathrm{P}(\mathcal{Y}) \to [-B/2, B/2]$ for $B > 0$ be a loss function that is α-Hölder continuous with constant M_2 in the first argument w.r.t. the total variation distance for some $\alpha \in (0,1]$. Then, there exists a dimensional constant c_d such that for $\gamma \in (0,1]$ and for all $n \in \mathbb{N}$, we have*

$$\mathbb{P}\left(\sup_{\theta\in\Theta} |R^{(\kappa)}(\theta) - R_n^{(\kappa)}(\theta)| \geq 4M\sqrt{\frac{2\pi\log n}{\gamma n}}\right) \leq c_d \left(\frac{\gamma n^{1/\alpha}}{2\pi \log n^{1/\alpha}}\right)^{\frac{d_{\gamma,n}^{(\kappa)}}{2}} \exp\left(-\frac{16M^2\pi\log n}{B^2\gamma}\right), \quad (7)$$

where $M = M_1^\alpha M_2$.

To prove the statement of the theorem we need a preparatory lemma that is the discretized version of Lemma 1.

Lemma 7. *Let $\Theta = [-1,1]^d$, and let $\mathcal{N}^{(\kappa)}(\varepsilon)$ denote the number of boxes of side length ε required to cover the parameter set Θ, the length being measured with respect to the metric $\hat{F}_{ij}^{(\kappa)}(\theta)$. Under the assumption of Theorem 6, there exists a dimensional constant $c_d < \infty$ such that for $\gamma \in (0,1]$ and for all $n \in \mathbb{N}$, we have*

$$\mathcal{N}^{(\kappa)}\left(\sqrt{\frac{2\pi\log n}{\gamma n}}\right) \leq c_d \left(\frac{\gamma n}{2\pi\log n}\right)^{d_{\gamma,n}^{(\kappa)}/2}.$$

Proof. Recall that we work in the discretized model $\mathcal{M}_\Theta^{(\kappa)}$, so our metric $\hat{F}_{ij}^{(\kappa)}(\theta)$ is constant on each element Θ_ℓ of the partition. So, we fix ℓ, and we count first the number of boxes of side length ε required to cover Θ_ℓ.

Up to a rotation, we can diagonalize the Fisher information matrix $\hat{F}^{(\kappa)}|_{\Theta_\ell}$ as $\mathrm{diag}(s_1^2, \dots, s_d^2)$. Note that Θ_ℓ has Euclidean diameter bounded by $2\kappa^{-1}$ and volume κ^{-d}. Also, if $\mathcal{B}_\varepsilon(\bar{\theta}_\ell)$ is a ball centered at $\bar{\theta}_\ell \in \Theta_\ell$ and of length ε, then

$$\mathcal{B}_\varepsilon(\bar{\theta}_\ell) \cap \Theta_\ell := \left\{ \theta \in \Theta_\ell \,:\, \sum_{i=1}^d s_i^2 [(\theta - \bar{\theta}_\ell) \cdot e_i]^2 \le \varepsilon^2 \right\}.$$

Then, the number of balls of size ε needed to cover Θ_ℓ is bounded by

$$\begin{aligned}
\hat{c}_d \prod_{i=1}^d \left\lceil 2\kappa^{-1}\varepsilon^{-1} s_i \right\rceil &\le \hat{c}_d 2^d \kappa^{-d} \prod_{i=1}^d \left\lceil \varepsilon^{-1} s_i \right\rceil \\
&\le \hat{c}_d 2^d \kappa^{-d} \sqrt{\prod_{i=1}^d \left(1 + \varepsilon^{-2} s_i^2\right)} \\
&= \hat{c}_d 2^d \int_{\Theta_\ell} \sqrt{\det\left(\mathrm{id}_d + \varepsilon^{-2} \hat{F}^\kappa(\theta)\right)} \mathrm{d}\theta \,,
\end{aligned}$$

where $\hat{c}_d$ is a positive dimensional constant, and the last equality follows from the fact that the volume of Θ_ℓ is equal to κ^{-d} and that $\hat{F}^{(\kappa)}$ is constant on Θ_ℓ.

Summing this bound over $\ell = 1, \dots, m$, we conclude that (note that $V_\Theta = 2^d$)

$$\mathcal{N}^{(\kappa)}(\varepsilon) \le \hat{c}_d 2^d \sum_{\ell=1}^m \int_{\Theta_\ell} \sqrt{\det\left(\mathrm{id}_d + \varepsilon^{-2} \hat{F}^{(\kappa)}(\theta)\right)} \mathrm{d}\theta = \hat{c}_d 4^d \frac{1}{V_\Theta} \int_\Theta \sqrt{\det\left(\mathrm{id}_d + \varepsilon^{-2} \hat{F}^{(\kappa)}(\theta)\right)} \mathrm{d}\theta \,.$$

Applying this bound with $\varepsilon = \sqrt{2\pi \log n/(\gamma n)}$ and recalling the definition of the effective dimension, the result follows. □

Proof of Theorem 6. We start be noting that Lemma 2 remains valid for the discretized setting and under the assumption of Theorem 6, where Lemma 2 does not require the full rank assumption of the Fisher information matrix, i.e.,

$$\mathbb{P}\left(\sup_{\theta \in \Theta} |R^{(\kappa)}(\theta) - R_n^{(\kappa)}(\theta)| \ge \varepsilon \right) \le 2\mathcal{N}^{(\kappa)} \left(\left(\frac{\varepsilon}{4M} \right)^{1/\alpha} \right) \exp\left(-\frac{n\varepsilon^2}{4B^2} \right), \tag{8}$$

where $\mathcal{N}^{(\kappa)}(\varepsilon)$ denotes the number of balls of side length ε, with respect to $\hat{F}^{(\kappa)}$, required to cover the parameter set Θ. This can be seen by going through the proof of Lemma 2. Hence, by Lemma 7, we find for $\varepsilon = 4M\sqrt{2\pi \log n/(\gamma n)}$

$$\begin{aligned}
&\mathbb{P}\left(\sup_{\theta \in \Theta} |R^{(\kappa)}(\theta) - R_n^{(\kappa)}(\theta)| \ge 4M\sqrt{2\pi \log n/(\gamma n)} \right) \\
&\qquad \le 4c_d \left(\frac{\gamma n^{1/\alpha}}{2\pi \log n^{1/\alpha}} \right)^{\frac{d^{(\kappa)}_{\gamma, n^{1/\alpha}}}{2}} \exp\left(-\frac{16M^2 \pi \log n}{B^2 \gamma} \right).
\end{aligned} \tag{9}$$

□

Remark 8 (How to choose the discretization parameter κ)**.** In this remark we discuss conditions such that the generalization bound of Theorem 6 for the discretized model $\mathcal{M}_\Theta^{(k)}$ is a good approximation to a generalization bound of the original model $\mathcal{M}_\Theta$. Assume that the model $\mathcal{M}_\Theta$ satisfies an additional regularity assumption of the form $\|\nabla_\theta \hat{F}(\theta)\| \le \Lambda$ for some $\Lambda \ge 0$ and for all $\theta \in \Theta$, then choosing the discretization parameter $\kappa \gg \Lambda$ ensures that that $\mathcal{M}_\Theta \approx \mathcal{M}_\Theta^{(\kappa)}$ and $F(\theta) \approx F^{(\kappa)}(\theta)$. Furthermore, $\sqrt{n} \gg \kappa$ is required to ensure that the balls used to cover each simplex of the triangulation are smaller than the size of each simplex.

References

[1] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta. Supervised learning with quantum-enhanced feature spaces. *Nature*, 567(7747):209–212, 2019. DOI: 10.1038/s41586-019-0980-2.

[2] R. Karakida, S. Akaho, and S.-I. Amari. Universal statistics of Fisher information in deep neural networks: Mean field approach. volume 89 of *Proceedings of Machine Learning Research*, pages 1032–1041. PMLR, 2019. Available online: http://proceedings.mlr.press/v89/karakida19a.html.

[3] J. Pennington and P. Worah. The spectrum of the Fisher information matrix of a single-hidden-layer neural network. In *Advances in Neural Information Processing Systems 31*, pages 5410–5419. Curran Associates, Inc., 2018. http://papers.nips.cc/paper/7786-the-spectrum-of-the-fisher.

[4] F. Kunstner, P. Hennig, and L. Balles. Limitations of the empirical Fisher approximation for natural gradient descent. In *Advances in Neural Information Processing Systems 32*, pages 4156–4167. 2019. http://papers.nips.cc/paper/limitations-of-fisher-approximation.

[5] Z. Liao, T. Drummond, I. Reid, and G. Carneiro. Approximate fisher information matrix to characterise the training of deep neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP:1–1, 2018. DOI: 10.1109/TPAMI.2018.2876413.

[6] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. *Efficient BackProp*, pages 9–48. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. DOI: 10.1007/978-3-642-35289-8_3.

[7] M. Cerezo and P. J. Coles. Impact of barren plateaus on the Hessian and higher order derivatives, 2020. Available online: https://arxiv.org/abs/2008.07454.

[8] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven. Barren plateaus in quantum neural network training landscapes. *Nature communications*, 9(1):1–6, 2018. DOI: 10.1038/s41467-018-07090-4.

[9] O. Berezniuk, A. Figalli, R. Ghigliazza, and K. Musaelian. A scale-dependent notion of effective dimension, 2020. Available online: https://arxiv.org/abs/2001.10872.

[10] P. Halmos. *Finite-Dimensional Vector Spaces*. Springer-Verlag New York, 1958. DOI: 10.1007/978-1-4612-6387-6.

[11] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of machine learning*. MIT press, 2018. Available online: https://cs.nyu.edu/~mohri/mlbook/.

[12] Z. Jia and H. Su. Information-theoretic local minima characterization and regularization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 4773–4783. PMLR, 2020. Available online: http://proceedings.mlr.press/v119/jia20a.html.

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. DOI: 10.5555/1953048.2078195.