Learning to rank quantum circuits for hardware-optimized performance enhancement

Gavin S. Hartnett, Aaron Barbosa, Pranav S. Mundada, Michael Hush, Michael J. Biercuk, and Yuval Baum

Q-CTRL, Sydney, NSW Australia, and Los Angeles, CA USA

We introduce and experimentally test a machine-learning-based method for ranking logically equivalent quantum circuits based on expected performance estimates derived from a training procedure conducted on real hardware. We apply our method to the problem of layout selection, in which abstracted qubits are assigned to physical qubits on a given device. Circuit measurements performed on IBM hardware indicate that the maximum and median fidelities of logically equivalent layouts can differ by an order of magnitude. We introduce a circuit score used for ranking that is parameterized in terms of a physics-based, phenomenological error model whose parameters are fit by training a ranking-loss function over a measured dataset. The dataset consists of quantum circuits exhibiting a diversity of structures and executed on IBM hardware, allowing the model to incorporate the contextual nature of real device noise and errors without the need to perform an exponentially costly tomographic protocol. We perform model training and execution on the 16-qubit *ibmq_guadalupe* device and compare our method to two common approaches: random layout selection and a publicly available baseline called Mapomatic. Our model consistently outperforms both approaches, predicting layouts that exhibit lower noise and higher performance. In particular, we find that our best model leads to a $1.8\times$ reduction in selection error when compared to the baseline approach and a $3.2\times$ reduction when compared to random selection. Beyond delivering a new form of predictive quantum characterization, verification, and validation, our results reveal the specific way in which context-dependent and coherent gate errors appear to dominate the divergence from performance estimates extrapolated from simple proxy measures.

Gavin S. Hartnett: gavin.hartnett@q-ctrl.com

1 Introduction

A wide range of quantum computers have become available through commercial, academic, and other public-sector platforms [1–3]. Both researchers and end-users are faced with a choice of architectures and qubit technologies, requiring a challenging process of selection in which the relative performance advantages and disadvantages of each specific platform must be assessed, synthesized, and used to choose a target hardware system for algorithmic execution. As both the number and size of available quantum devices grow, the question of how to select the best device or sub-system of a device for the execution of a given quantum circuit is becoming increasingly important. Ultimately, this process is an ideal target for automation and abstraction if suitable tools for predicting the expected performance of a target algorithm on a hardware backend can be devised.

The topic of quantum computer benchmarking, or Quantum Characterization, Validation, and Verification (QCVV), aims to provide insights informing this selection process. This field has generally focused on enabling the identification of quantum processors with the best overall performance characteristics based on abstracted and human-interpretable proxy measures describing low-level physical parameters such as qubit coherence or quantum logic gate errors, as derived from measurement protocols such as Randomized or Cross-Entropy Benchmarking [4–7]. While it is intuitive to link low-level hardware characteristics to the device selection process, much subtlety is

masked by the fact that the dominant framework for QCVV fundamentally trades between simplicity/efficiency and predictive power [8–10].

Extrapolating from hardware-level proxies to system-level performance is complicated by the fact that most common scalable QCVV approaches are by design insensitive to the myriad sub-structures, or contexts, that can appear within quantum circuits and which can have important impacts on the performance. This is in part due to the historic use of these metrics in analyses of fault tolerance, in which errors are typically (and speciously) assumed to be statistically independent. For example, randomized benchmarking (RB) is relatively efficient to implement and easy to interpret, yet it masks coherent errors exhibiting correlations in time over many gates in a circuit [10, 11]. Similarly, crosstalk may change both the magnitude and the nature of the error of a specific gate depending on the existence or absence of simultaneous gates on neighboring qubits [11, 12] - contexts that are ignored or averaged over in standard gate-level QCVV protocols. In particular, errors that occur during idling periods are in general poorly captured by most standard methods, both at the single qubit level where T_2 effects are relevant, and at the multi-qubit level where quantum crosstalk can lead to an unwanted unitary evolution between neighboring qubits [13–15]. Furthermore, the overall effect of an error on the measured fidelity of a quantum circuit will be dependent on the structure of the circuit; random circuits will generally be less sensitive to over-rotation gate errors due to self-cancellations of the random excess angles [16], whereas circuits with a repetitive structure, such as Trotter circuits, will be very sensitive to these types of errors [8, 10, 17]. The aggregate effect of these various circumstances - all of which are routinely encountered in the execution of real algorithms on hardware - tends to cause QCVV protocols to only be loosely predictive of algorithmic performance [9, 14].

Here, we focus on the concrete problem of *layout selection* within a single processor: the mapping of virtual or abstract qubits in a compiled circuit to physical device qubits in a manner that maximizes circuit fidelity. Previous work has examined the topic of layout selection via various methods [18–23] (see also [24, 25]), but has not presented a comprehensive ranking procedure based on measurements of real circuits executed on hardware. This omission ignores the essential contextual information that is known to cause divergence between proxy measure predictions and achievable performance.

In this work, we develop a new form of *predictive* QCVV for layout selection for arbitrary target circuits based on a heuristic, machine learning solution. We report direct measurements on an IBM quantum computer revealing that different layouts of individual target circuits exhibit up to a factor of 11× variation in the ratio of the maximum achieved to median Hellinger fidelity, highlighting the importance of this task. We develop a procedure to rank the expected relative performance of logically equivalent layouts for arbitrary target circuits within a quantum processor, drawing from a large body of work in machine learning where this general problem is known as "Learning to Rank" [26–29]. In so doing, this method prioritizes the *relative* performance between logically equivalent permutations of a user-defined target circuit over accurate estimates of absolute performance, and omits the reliance on human-interpretable proxy measures. Our method trains a circuit-score model over a dataset of circuits exhibiting diverse structural characteristics, and parameterizes the target-circuit score in a physics-based phenomenological manner. The model then assigns to each layout of the target circuit a numerical score used to quantify relative expected performance (a higher score corresponds to a higher expected performance). We test the efficacy of the ranking procedure using direct fidelity measurements of the different layouts on hardware, and compare our learning-to-rank method against alternative layout-selection approaches. We also compare five different candidate loss functions and quantitatively demonstrate that the listwise Rank MSE loss function performs best across multiple metrics. For the best model, measurements demonstrate a reduction in selection error – the relative loss in Hellinger fidelity due to sub-optimal layout selection – by 3.2× (1.8×) when compared to a random and a baseline error-informed layout selection strategy, respectively, validating the utility of this approach.

The rest of the article is organized as follows. In Sec. 2 we briefly review the problem of layout selection. In Sec. 3 we introduce our approach, and then perform experiments on real quantum

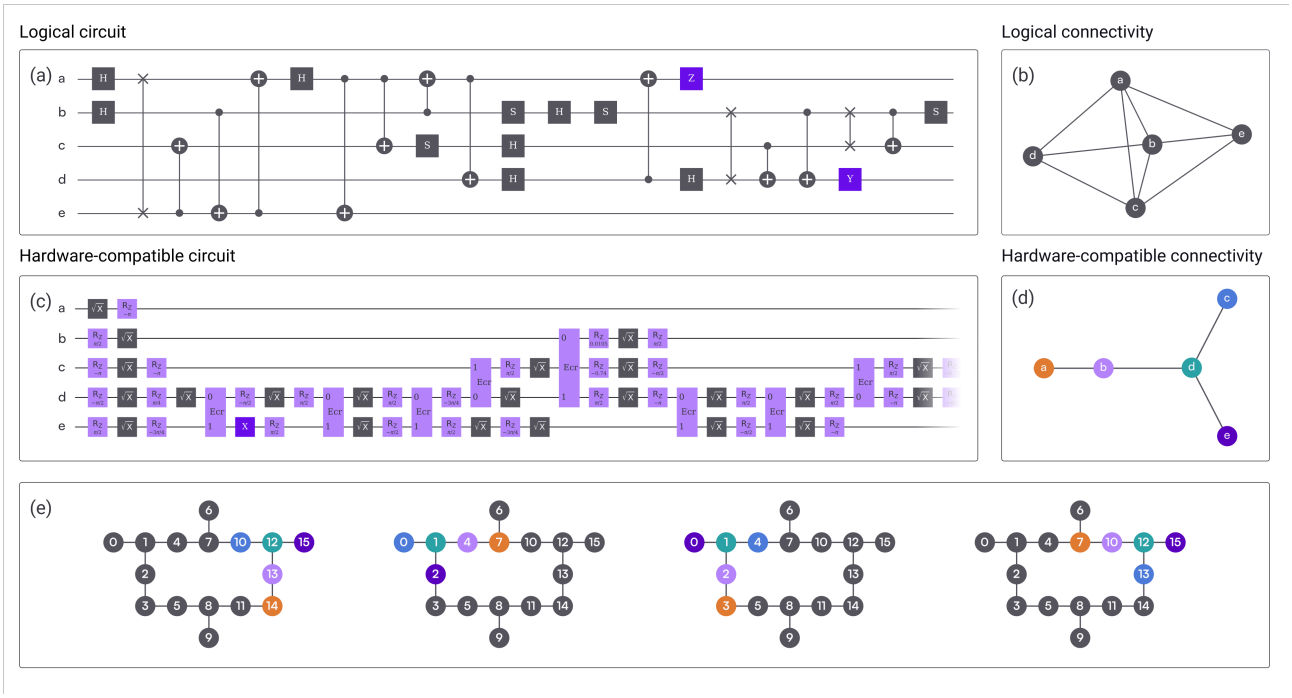

Figure 1: Circuit transpilation and the task of layout selection. (a) A representative initial circuit. Such circuits are unconstrained by hardware considerations, such as matching the device topology or consisting only of native gates. (b) The qubit connectivity graph of the initial circuit. Nodes represent the qubits appearing in (a) and edges represent 2-qubit gates. (c) The hardware-compatible circuit. Hardware transpilation transforms the initial circuit into a unitarily equivalent, hardware-compatible circuit. Shown here is the hardware-transpiled circuit corresponding to the initial circuit on the *ibmq_guadalupe* device (only a portion is shown for brevity). (d) The qubit connectivity graph of the hardware-transpiled circuit. Unlike the initial connectivity graph, the transpiled connectivity graph is a subgraph of the device topology (shown in (e)). The node labels correspond to the qubit labels in (d). (e) The layout is a map between circuit qubits and physical qubits. There are typically many such mappings; shown here is a subset of four layouts. The goal of layout selection is then to choose the best-performing layout for execution on the device.

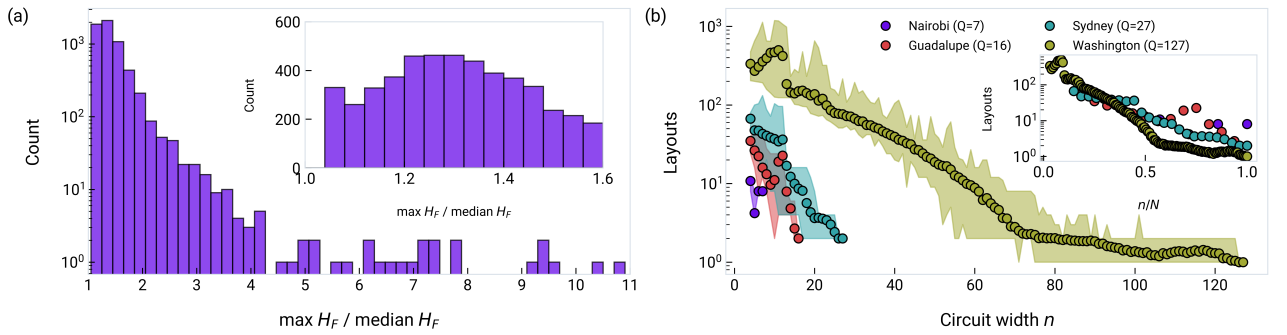

Figure 2: The importance of layout selection. (a) Distribution of the performance quality ratio between the median- and maximum-performing layouts as measured by the Hellinger fidelity, H_F, evaluated over large a data set of more than 6000 circuits (see the main text for more details about the data set). The inset shows a zoomed-in and higher-resolution plot for performance qualities between 0 and 60%. (The first bar in the inset has zero counts, i.e., the difference between the top and median performing layouts exceeded 4% for all circuits.) (b) The number of layouts, averaged over 500 random initial circuits, as a function of the circuit width n. The markers indicate averages, and the shaded region indicates the range. The inset shows the average number of layouts as a function of n/N, the fraction of device qubits used. Note that these results assume bi-directional device graphs.

hardware in Sec. 4, before concluding with a discussion in Sec. 5. Further technical details are included in Appendix A and B.

2 The layout-selection problem

Hardware transpilation takes a generic circuit as input and modifies it so that it consists only of operations native to a given quantum backend. Crucially, the circuit modifications made in this step must preserve the logical operation of the circuit, i.e., the unitary transformation effected by the circuit must remain unchanged (perhaps up to a specified non-zero tolerance). For example, circuit identities involving SWAP gates may be used to adjust the qubit connectivity of a circuit to match a given device topology. Most backends natively support only single- and two-qubit gates, and throughout this work, we will assume this as well (q-qubit gates for $q > 2$ can be decomposed in terms of 1- and 2-qubit gates during the compilation step).

The transpiled circuit and quantum backend device may each be associated with graphs that will be central to the formulation of the layout selection problem. For every two-qubit gate present in the circuit, the circuit graph will contain an edge connecting the participating qubits. Similarly, the edges of the device graph will correspond to the qubit connectivity of the backend. The impact of this transition from abstract logical circuits to hardware-compatible circuits is illustrated in Fig. 1(a)-(d). Generally, the sparse connectivity of the device translates into an increased complexity of the transpiled circuit.

Further, a circuit will "fit" on a device if the circuit graph is subgraph-isomorphic to the device graph. Such subgraph isomorphisms are represented as layouts, mappings from the abstract circuit qubits to specific physical device qubits, as shown in Fig. 1(d)-(e). The set of all layouts for a given transpiled circuit can be efficiently identified using VF2, a well-known subgraph isomorphism algorithm [30], as well as related algorithms [31].

Once the set of allowed layouts for a target circuit has been enumerated, the problem of *layout selection* is to identify a single layout to be used for the actual execution of the circuit on a device, mapping abstract or virtual qubits to physical devices and gate sets. This introduces a mechanism for significant performance variability, as gate errors and coherence times can exhibit order-of-magnitude variation across a single device [11]. In addition, the fidelity and execution time of an individual gate will generally be context-dependent, varying with the selected subset of participating qubits.

Due to the variability among layouts, layout selection is an integral sub-task in quantum circuit compilation that can contribute substantially to the overall performance of the circuit when executed on hardware. This is demonstrated in Fig. 2 (a), which shows experimental measurements of the circuit fidelity achieved for over 6000 initial circuits. For each circuit, all possible layouts were identified and executed on the 16-qubit *ibmq_guadalupe* device. The figure shows the distribution of the ratio between the performance of the best and median layouts, as measured using the Hellinger fidelity. [1] All circuits are executed in an identical manner, and comparisons are made strictly between measured results vs theoretical predictions. Any deviation from the value one on the horizontal axis of Fig. 2 (a) represents a deleterious impact on circuit fidelity as a result of sub-optimal layout selection. For example, a value of 3 indicates that with 50% probability, a randomly selected layout will exhibit 3× worse performance than that achieved by the top-performing layout. For a majority of layouts,f the degradation in performance between the top and median layouts is substantial - for example, about 28% of circuits exhibit a max-to-median ratio that exceeds 1.5×.

Layout selection is further complicated by the combinatorially large number of possible layouts - a number determined by the connectivity graphs of the transpiled circuit and the device. In general, the two key quantities determining the number of layouts are the circuit width, n, and the number of qubits in the device, Q. For $n \ll Q$

[1] The Hellinger fidelity is a measure of the closeness of two probability distributions. For distributions over the set of length n bitstrings, $H_F(p,q) = \left(\sum_{x\in\{0,1\}^n} \sqrt{p(x)q(x)}\right)^2$. $H_F = 1$ corresponds to perfect agreement, and $H_F = 0$ corresponds to p and q having zero overlap. In this work, the two distributions of interest are $\hat{q}$, the empirical bitstring distribution obtained through bitstring counts in N_{shots} measurements of an executed circuit, and p, the ideal, noise-free bitstring distribution. We use the overloaded notation $H_F(C)$ to denote $H_F(p,\hat{q})$.

the number of layouts can be extremely large, and it gradually diminishes as $n \to Q$, given there are fewer accessible ways to "fit" a large circuit on the device. This relationship is demonstrated in Fig. 2 (b), which shows the number of layouts for random circuits of varying widths compiled for the IBM devices *ibm_nairobi*, *ibmq_guadalupe*, *ibmq_sydney*, *ibm_washington*, which have $Q = 7, 16, 27,$ and 127, respectively. For circuits using less than half of the available qubits on the 127-qubit *ibm_washington* device, the number of layouts can reach several hundred.

Given the variability of circuit performance demonstrated in Fig. 2 – with the maximum circuit sometimes exhibiting more than 10× higher performance than the median – it becomes imperative to define a procedure to predict which among this potentially large selection of equivalent circuit layouts is most likely to exhibit the highest performance. To address this, in the following, we present a learning algorithm allowing for approximation of this ordinal ranking that can be run prior to compilation and execution on hardware.

3 Circuit ranking for layout selection

We now present an overview of a practical and efficient machine-learning approach for ranking-based layout selection. We use a score-based approach for ranking circuit performance. The circuit score S is a function that maps scheduled circuits to a real-valued numerical score: higher scores correspond to higher estimated fidelities. In a good ranking procedure, ordering layouts by circuit score should lead to the same ordinal ranking among circuits as a chosen performance metric derived from measurements on hardware. In this work, we take this measured quantity to be the Hellinger fidelity H_F between the ideal bitstring distribution and the empirically observed bitstring distribution.

Therefore, given two circuits C_1, C_2, the following relation should hold:

$$H_F(C_1) \lesssim H_F(C_2) \quad \text{iff} \quad S(C_1) \lesssim S(C_2)\,. \tag{1}$$

Our objective is now to learn score functions that approximately obey Eq. 1.

The circuit score is modeled using a hybrid approach that augments phenomenological physical models of device errors with an experimental-data-driven, machine learning approach. First, we assume that the score may be written as a product of terms

$$S = \prod_a (S_a)^{p_a}\,, \tag{2}$$

with each $S_a \in [0, 1]$. The power parameters account for the fact that the circuit structure affects the relative importance of different noise mechanisms; for example, deep random circuits will be more sensitive to incoherent errors and less to systematic gate calibration errors compared to shallow structured circuits (such as Trotter circuits). Complete definitions of the S_a scores are presented in Appendix A.

These scores are parameterized in terms of physically meaningful quantities that are typically estimated by the hardware provider, such as gate and measurement error rates, T_1 times, and two-qubit interaction couplings. However, standard calibration routines do not provide generalized context-aware information: during calibration specific circuit structures are deliberately employed to "amplify" the physical process being characterized. This intentional bias in characterization contributes to errors when ranking layouts, by missing the link between the context of the circuit to be executed and the isolated performance of the underlying hardware.

We address this shortcoming by promoting these parameters to be learnable, and the circuit score to be a machine learning model. The parameters are fit by minimizing a loss function $\mathcal{L}(\theta)$ over a training dataset, where we introduce θ as the vector of model parameters:

$$\theta_* = \mathrm{argmin}_\theta\, \mathcal{L}(\theta)\,. \tag{3}$$

The loss function provides a continuous and differentiable way to penalize deviations from Eq. 1. We consider multiple loss functions, summarized in Table 1, reflecting the fact that there does not appear to be a clear preferred choice in the Learning-to-Rank literature. These can be grouped into three categories: pointwise, pairwise, and listwise. Pointwise approaches do not involve any comparisons, pairwise approaches involve comparisons between pairs of layouts, and listwise approaches involve comparisons across a set of all possible layouts for a given initial circuit. One important consideration of the pairwise loss, compared to the pointwise MSE loss, is that it

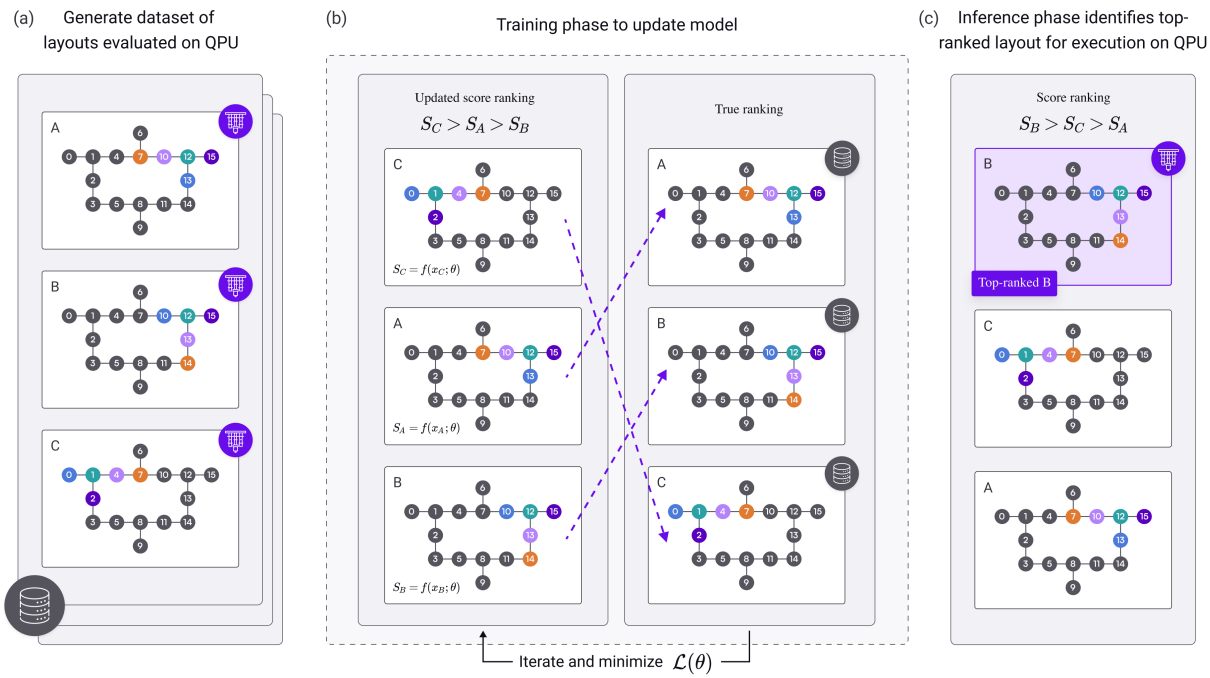

Figure 3: Machine learning pipeline for layout selection. (a) Layout generation. Graphs indicate connectivity of *ibmq_guadalupe* device: graph nodes indicate qubits and colored nodes are guides to the eye indicating different layouts for a notional five-qubit circuit. Only three layouts, A-C, shown for visual clarity. Different layers represent sets of layouts for different initial circuits. Purple QPU icons indicate hardware execution. The black database icon indicates that data from hardware execution form a dataset. (b) Overview of training phase. Ordering of layouts is now set by ranking conducted using current state of model. Purple dashed arrows indicate mismatches between model prediction and true ranking determined by database. The lower arrow indicates an iterative loop to minimize loss function via model parameter updates (see Table 1). (c) Overview of inference phase. The highest-ranking layout is selected for hardware execution, indicated by purple shading. There is no restriction that these layouts be present in the dataset, and in the primary use-case, the goal will be to predict layout rankings when the true ranking is unavailable

facilitates a quadratic expansion of the dataset: an initial dataset of N distinct circuits yields $N(N-1)/2$ distinct pairs of circuits. This expansion can be useful because it allows for many more training examples to be drawn from a given dataset, but it also has drawbacks because the dataset size can become unwieldy as well as depart from being independent and identically distributed (iid) [29]. For these reasons, we did not implement a pairwise loss in our experiments. Details for the various loss functions considered are presented in Appendix B.

For Learning-to-Rank problems, a dataset of objects and their true ranking is required. In the present context, the objects are transpiled circuits differing only in their layout, and the true ranking is provided by the Hellinger fidelity deduced through hardware execution. Such a dataset can be created by first sampling an initial circuit from a statistical ensemble (to be discussed presently), transpiling that circuit into multiple circuits differing only in their layout, executing each transpiled circuit on a quantum device, and finally computing the Hellinger fidelity. This procedure results in a dataset of the form

$$\mathcal{D} = \{\mathcal{C}_i, \mathcal{F}_i\}_{i=1,\ldots,N_B}\,, \tag{4}$$

where $\mathcal{C}_i = \{C_{i,1}, C_{i,2}, \ldots C_{i,L_i}\}$ is a batch of L_i circuits, one for each possible layout, and where $\mathcal{F}_i = \{H_F(C_{i,1}), H_F(C_{i,2}), \ldots, H_F(C_{i,L_i})\}$ is the collection of Hellinger fidelities for each circuit in the batch. Here, N_B counts the total number of layout-batches considered, which is equal to the number of initial, un-transpiled circuits. Note that the Hellinger fidelity compares the empirically measured and ideal bitstring distributions, and therefore its computation presupposes the ability to simulate the circuit.

To ensure that our model leads to an accurate ranking across a range of quantum algorithms,

Table 1: Learning to rank loss functions. Multiple loss functions were considered; each penalize deviations from a perfect ranking in a different manner. The Score MSE loss encourages the score function to agree pointwise with the Hellinger fidelity. The Pearson and Soft Spearman losses encourage the score to be perfectly correlated with Hellinger fidelity across a batch of layouts. The Rank MSE encourages the Hellinger fidelity rank to agree with the score rank across a batch of layouts. Lastly, the Negative Log-Likelihood (NLL) loss maximizes the probability of predicting the top-K layouts, with K a hyper-parameter.

Loss function	Type	Optimization description
Score MSE	pointwise	Minimize difference between score and Hellinger fidelity
Pearson	listwise	Maximize correlation between score and Hellinger fidelity
Soft Spearman	listwise	Maximize correlation between score and Hellinger fidelity
Rank MSE	listwise	Minimize difference of true and predicted rank
NLL	listwise	Maximize model probability of observed top-K layouts

the circuit score is learned by minimizing the loss function averaged over a training dataset consisting of a diversity of circuits. These are chosen to reflect a wide range of structures and depths commonly encountered in real algorithms, making them differentially sensitive to different types of error:

- Bernstein-Vazirani [32] circuits have a shallow, sequential structure, and are relatively more sensitive to idling and SPAM errors.
- Inverse QFT circuits consist of a layer of single-qubit gates, followed by the inverse quantum Fourier transform circuit primitive. The initial layer is such that the ideal output state is a computational basis state. These are deeper circuits (quadratic in n), and are more sensitive to 2-qubit gate errors.
- Clifford-conjugated-Pauli circuits (also referred to as Mirror circuits in [9, 17]) are of the form $C = (C_{\text{Clifford}})^\dagger \left(\otimes_{k=1}^{n} P_k\right) C_{\text{Clifford}}$, where P_k is a single-qubit Pauli gate, and $C_{\text{Clifford}} \in \text{Cl}(n)$ is a circuit implementing a random Clifford element (either chosen uniformly at random using the algorithm of [33] or by uniformly sampling a pre-determined number of Clifford gates). These have a high density with randomized gates and are therefore mainly sensitive to incoherent errors.
- QAOA circuits [34] exhibit arbitrary rotations and are therefore mainly sensitive to over-rotation coherent errors.

In addition to containing a broad range of contexts, these circuits were chosen because they are either efficiently simulable or can be easily constructed to return a desired output, which allows the Hellinger fidelity between the ideal bitstring distribution and the empirically observed distribution to be easily calculated. While not efficiently simulable, QAOA circuits are included as they are currently a very popular algorithm for execution on existing quantum devices.

A summary of the machine learning pipeline for layout selection is presented in Fig. 3. To train the ML model, a dataset of circuits is first generated. This is illustrated in Fig. 3(a), where each layer represents a batch of layouts transpiled to the hardware for a different circuit. Each layout is then executed on the quantum processing unit (QPU) and the Hellinger fidelity is computed for all layouts by comparing the ideal bitstring distribution against the empirical distribution. Together, these batches of executed layouts and their fidelities form a dataset of labeled data. In the training phase, illustrated in Fig. 3(b), the batches of layouts in the dataset are ranked and the model iteratively updated. For each batch, the current state of the model produces a score ranking that is compared against the true ranking derived from the Hellinger fidelity. Mismatches in the two rankings are penalized by the differentiable loss $\mathcal{L}(\theta)$. The model parameters θ are updated through multiple iterations of a gradient-based algorithm. In the inference phase, illustrated in Fig. 3(c), the goal is to use the trained model to predict a single layout for execution on the QPU. The input is a single transpiled circuit and the most up-to-date backend information (such as gate errors and T_1 times). The set of all valid layouts is generated using the subgraph isomorphism capability of the Mapomatic package, and a score S is assigned to each layout by the already trained model. The layouts are ranked according to the circuit score, and the circuit with

the largest score is chosen for execution on the device.

4 Experimental results

We perform circuit ranking using training data and experimental measurements executed on IBM hardware. In parameterizing the circuit score, we therefore consider four common error mechanisms for fixed-frequency superconducting qubit devices: gate error (for both one- and two-qubit gates), measurement error, T_1 error, and ZZ crosstalk [14]. Each error type is separately modeled using a parameterized phenomenological approach based on device parameters resulting in a set of four scores: S_{gate}, S_{msmt}, S_{T_1}, and S_{ZZ}.

A training dataset of different circuit types is generated for the 16-qubit *ibmq_guadalupe* device, and split into training and testing sets according to an 80:20 ratio. These 6020 initial circuits become 132,866 circuits after hardware transpilation, and the VF2 algorithm (as implemented in [35]) is used to calculate all possible layouts. For each circuit, we compute the ideal bitstring distribution and execute the circuit on the quantum computer using $N_{\text{shots}} = 4096$ in order to obtain the Hellinger fidelity. To ensure a clear learning signal for our circuit score, we apply an automated error suppression pipeline from our software suite to each executed circuit [14] to maximize execution fidelity. Due to the large number of layouts considered, this data collection was spaced over a period of 9 months (from Nov 2022 to July 2023). All models are trained using the gradient-based Adam optimizer [36] and the OneCycleLR learning rate scheduler [37].

We benchmark our approach against both random selection and selection via the Mapomatic package [35]. Random selection is the default approach when utilizing a standard compiler. Many informed users employ Mapomatic, a layout selection method that estimates circuit fidelity by aggregating the backend-estimated fidelities of each individual operation (i.e., gate errors, measurement errors, etc). [2] Mapomatic also uses a scoring function to rank layouts; the key points of difference are in the choice of score function and the fact that in the models developed here, the parameters are *learned* by training the model over a dataset containing a diverse range of circuit contexts.

To evaluate different methods, we quantify the quality of a layout selection procedure using a number of evaluative metrics. First, the true rank of the selected layout serves as the natural metric for this learning-to-rank problem. We adopt the convention that the true rank R ranges from 1 (top-ranked) to L (worst-ranked) for a batch of L layouts. As L can vary significantly depending on the algorithm and device, it is also useful to consider a normed rank $r = (R-1)/(L-1)$, with $r \in [0,1]$ (a lower value is better). Second, the Top-1 accuracy is defined as the fraction of circuits for which the model predicts the top-ranked layout. Third, the selection error, defined as $1 - H_F(C_{\text{pred}})/H_F(C_{R=1})$, measures the fraction of the optimal Hellinger fidelity that is lost by choosing a sub-optimal layout. In particular, the top-ranked layout by definition has a selection error of zero.

Fig. 4 presents the key results of our machine-learned circuit score method for layout selection, comparing the various metrics and loss functions introduced above; numerical values are tabulated in Table 2. In all cases, the circuit score outperforms the baseline provided by Mapomatic or random selection. The best-performing model across all metrics were trained on the Rank MSE loss function. Compared to random selection, the Rank MSE models correspond to a 3.6× improvement in the normed rank and a 3.2× improvement with respect to the selection error. Compared to Mapomatic, the Rank MSE models correspond to a 2.5× improvement in the normed rank and a 1.8× improvement with respect to the selection error.

The Rank MSE models, on average, predict layouts that achieve about 89% of the Hellinger fidelity of the optimal layout (i.e., the selection error is about 11%). The median normed rank predicted by these models is about 0.13, and the top layout is selected 22% of the time. In comparison, Mapomatic achieves a selection error of about 19%, an average normed rank of 0.34, and predicts the top layout 15% of the time.

We can also compare layout selection methods

[2] Note that Mapomatic is the name of both the layout selection algorithm as well as the Python package that first identifies all valid layouts and then ranks them according to a simple heuristic score - here, we will use Mapomatic to specifically refer to the score-based layout selection algorithm.

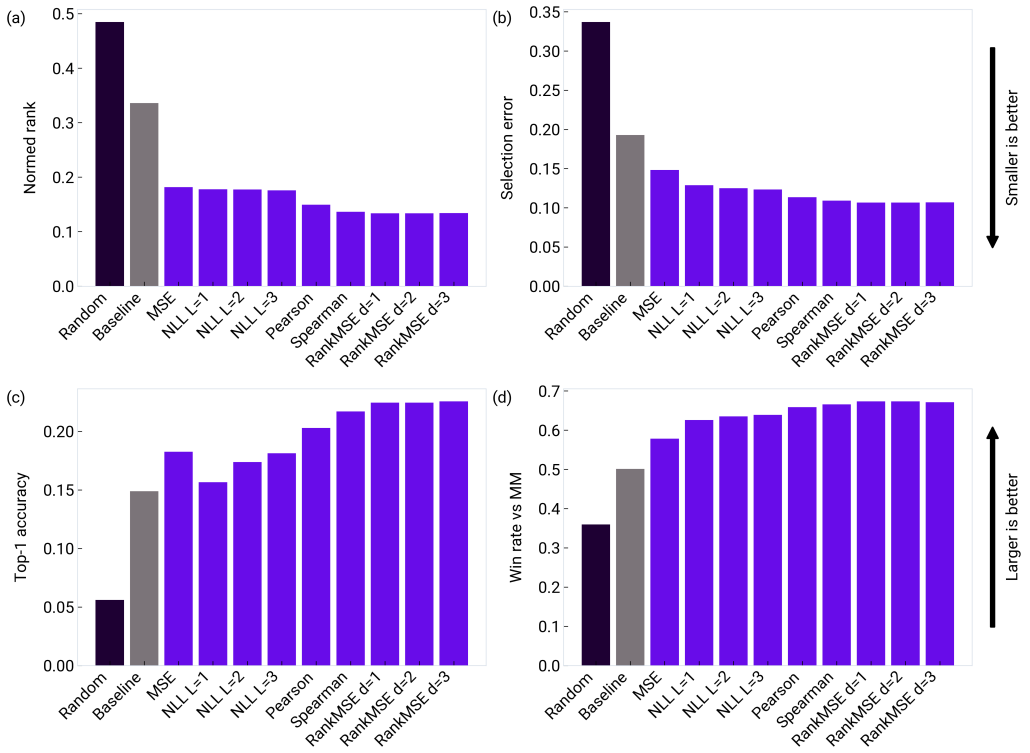

Figure 4: Model performance. The performance of circuit score models trained on the loss functions listed in Table 1, as measured by multiple evaluation metrics averaged over the test set. The win rate is measured against Mapomatic (MM). Note that a lower value is better for the normed rank (a) and selection error (b), whereas a higher value is better for the top-1 accuracy (c) and win rate (d). To account for statistical variations in the optimization and in the training/testing split, for each loss function 10 random seeds are used, and numerical values are averaged over these.

using the win rate

$$\text{win rate} = \frac{\text{wins} + \frac{1}{2} \times \text{ties}}{\text{total games}}, \tag{5}$$

which treats each layout selection problem as a "game", and assigns a win to the method that predicted a better-performing layout. Ties are reserved for the case when both methods predict equally performant layouts. A 50% win rate of a method compared to the baseline signifies an equivalence between the methods, while a win rate above (below) 50% signifies superiority (inferiority) compared to the baseline. [3] The baseline ranking method (Mapomatic) outperforms random selection, as indicated by the 35% win rate listed in Table 2 (equivalently, Mapomatic "wins" against random selection 65% of the time). The learned circuit score models all consistently outperform Mapomatic, as indicated by win rates >50%. In particular, our best models exhibit a win rate of 67%, indicating a substantial improvement over the baseline method.

[3] Given that the empirical Hellinger fidelity will be subjected to shot noise, it would be sensible to define wins and ties within some tolerance. However, averaging over a dataset of many circuits will reduce sensitivity to these fluctuations.

5 Discussion

In this work, we considered the general problem of identifying the best device or sub-system of a device and focused on a particular instantiation of this problem in the form of layout selection for a single device. Layout selection is an oft-overlooked component of quantum compilation and transpilation. Omitting this step by simply selecting a layout randomly is strongly sub-optimal.

We developed a novel score-based method for ranking quantum circuit performance as a form of predictive QCVV. The score function is trained by minimizing a loss function that penalizes deviations from the ideal ranking over a large dataset of circuits containing a wide range of contextual errors that are likely to appear in circuits of in-

Table 2: Model performance. The performance of the circuit score model is measured by multiple evaluative metrics. Median rank r corresponds to the median normed rank, Median rank R corresponds to the median un-normalized rank, Selection Err corresponds to the selection error, and Top-1 Acc corresponds to the Top-1 accuracy. Each row corresponds to a circuit score model trained on a different loss function, averaged over 10 random seeds. Uncertainties correspond to 1 standard error. For comparison, the results for random layout selection and Mapomatic are also shown. The win rate of each method against Mapomatic (MM) is shown in the final column.

Model	Loss	Median rank r	Median rank R	Selection Err (%)	Top-1 Acc (%)	Win Rate (%)
Random	N/A	0.4838(57)	10.00(15)	33.64(30)	5.56(27)	35.86(42)
Mapomatic	N/A	0.3351(53)	7	19.24(11)	14.86(30)	50
Circuit Score	MSE	0.1808(22)	4.50(17)	14.759(92)	18.23(26)	57.71(32)
	$\text{NLL}_{K=1}$	0.17701(53)	4.20(13)	12.818(80)	15.63(32)	62.48(36)
	$\text{NLL}_{K=2}$	0.1764(35)	4.20(13)	12.44(24)	17.34(41)	63.40(52)
	$\text{NLL}_{K=3}$	0.17494(42)	4.10(10)	12.29(11)	18.09(38)	63.80(34)
	Pearson	0.1487(50)	4.0	11.295(62)	20.26(31)	65.73(25)
	Spearman	0.1358(23)	3.90(10)	10.863(68)	21.67(46)	66.47(27)
	$\text{Rank MSE}_{d=1}$	**0.13275(39)**	**3.70(15)**	**10.603(97)**	22.41(41)	**67.19(18)**
	$\text{Rank MSE}_{d=2}$	**0.13275(39)**	**3.70(15)**	**10.603(97)**	22.41(41)	**67.19(18)**
	$\text{Rank MSE}_{d=3}$	0.13304(29)	3.80(13)	10.638(78)	**22.52(46)**	67.03(18)

terest, and which are missed through an exclusive reliance on simplified proxy measures. Once trained, the score function is efficient to compute and introduces no additional overhead. We experimentally verified on IBM hardware that our method provides a 1.8× improvement in selection error when compared to a baseline approach, and 3.2× when compared to random layout selection.

The results we present are generically applicable to ranking problems in the execution of hardware circuits and can be adapted beyond layout selection within a single device. For instance, as an increasing number of hardware providers expose their devices to the public, identifying the best device for a given task is an important capability. Various algorithmic benchmarking routines have been explored to enable comparison between quantum processors, but linking benchmarking datasets to prediction for a user's specific algorithm has not been possible [38, 39]. The learning-to-rank protocol developed here can be directly applied to this problem, potentially even leveraging training data derived from the various benchmarking packages in use.

Data availability

The data supporting the findings of this study are publicly available in Zenodo at https://doi.org/10.5281/zenodo.14037434 [40].

Acknowledgments

The authors are grateful to all other colleagues at Q-CTRL whose technical, product engineering, and design work has supported the results presented in this paper.

A Phenomenological circuit score

The overall circuit score is expressed as a product of circuit scores for four mechanisms, $S = \prod_a (S_a)^{p_a}$, where a runs over gate, measurement, T_1, and ZZ errors. The power parameters p_a are introduced to account for possible systematic under/over-weighting of these different errors. Explicitly,

$$S = \left(S_{\text{gate}}{}^{p_{\text{gate}}}\right)\left(S_{\text{msmt}}{}^{p_{\text{msmt}}}\right)\left(S_{T_1}{}^{p_{T_1}}\right)\left(S_{ZZ}{}^{p_{ZZ}}\right). \tag{6}$$

Gate errors are modeled as independent binary events, with each gate g inducing an error with probability $1 - f(g)$, where $f(g)$ is the fidelity of that particular gate. Note that here the gates are taken to be specified to the qubits upon which they act - for example, an X gate acting on qubit 1 has, in general, a different fidelity than an X gate acting on qubit 2. Thus, the gate score for circuit

C, as an estimate of the total gate fidelity, is modeled as:

$$S_{\text{gate}}(C) = \prod_g f(g)^{n(C;g)}\,, \tag{7}$$

where the product is over all native gates for the device in question, and $f(g)$, $n(C;g)$ are the fidelity and count of gate g, respectively. The gate fidelities are device constants that can be obtained directly from the backend, typically through Randomized or Cross-Entropy Benchmarking [4–7], which provides a single fidelity measure that is insensitive to the circuit context. However, in this work, we treat $f(g)$ and $f(\text{msmt}_i)$ as learnable parameters initialized to the estimated values obtained from the backend.

The measurement error is modeled similarly,

$$S_{\text{msmt}}(C) = \prod_{i=0}^{n-1} f(\text{msmt}_i)^{n(C;\text{msmt}_i)}\,, \tag{8}$$

where msmt_i represents the measurement operation on qubit i, and $f(\text{msmt}_i)$ corresponds to one minus the mean miss-classification rate. Note that measurement error mitigation has been applied to the executed circuits, effectively reducing this error rate.

Next, we model the effect of T_1 decoherence on idle qubits. According to the Bloch-Redfield decoherence model [41], the time-dependent density matrix of an idle qubit in a general pure state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ evolves as:

$$\rho(t) = \begin{pmatrix} 1 + (|\alpha|^2 - 1)e^{-\Gamma_1 t} & \alpha\beta^* e^{-\Gamma_2 t} \\ \alpha^*\beta e^{-\Gamma_2 t} & |\beta|^2 e^{-\Gamma_1} \end{pmatrix}\,, \tag{9}$$

where $\Gamma_1 = 1/T_1$ is the longitudinal relaxation rate, $\Gamma_2 = \Gamma_1/2 + \Gamma_\phi$ is the transverse relaxation rate, which includes a contribution from the longitudinal rate as well as a pure dephasing term $\Gamma_\phi = 1/T_2$. The fidelity between the time-dependent and initial states is:

$$F(\rho(0), \rho(t)) = |\alpha|^2 + \left(|\alpha|^4 + |\beta|^4 - |\alpha|^2\right) e^{-\Gamma_1 t} + 2|\alpha|^2|\beta|^2 e^{-\Gamma_2 t}\,, \tag{10}$$

This fidelity is clearly state-dependent. It decays exponentially to the minimal value of $|\alpha|^2$, with $|\alpha| = 1$ corresponding to the special case where the initial state is at the North Pole of the Bloch sphere and is therefore unaffected by these decoherence effects.

For the purpose of developing a simple phenomenological error metric applicable to typical quantum circuits, we average over all possible input states using the Haar measure. As our circuit implementation includes dynamical decoupling, the effect of T_2 is highly suppressed. To isolate the effect of T_1 we set $\Gamma_\phi = 0$. The final result is that the fidelity of an average single-qubit state subject to T_1 decoherence for an idle time t is

$$f_{T_1}(t; \Gamma_1, a, b) := \bar{F}(\rho(0), \rho(t)) = (1 - a - b) + a\, e^{-\Gamma_1 t/2} + b\, e^{-\Gamma_1 t}\,, \tag{11}$$

where $a = 1/3$, $b = 1/6$ are constants obtained from the Haar average. In the machine learning model, a and b are promoted to model parameters which are initialized to the Haar-averaged values. Note that the rate parameter Γ_1 will be both device- and qubit-dependent. We use Γ_1 the values provided by the backend during the circuit execution.

To obtain a circuit-wide fidelity metric, we assume that the T_1 errors act independently for each qubit and for each idle period:

$$S_{T_1} = \prod_{i=0}^{n-1} \prod_{t \in \text{idle intervals(i)}} f_{T_1}(t; \Gamma_1^{(i)}, a, b)\,. \tag{12}$$

Here i runs over each qubit in the device, and the notation idle intervals(i) is used to indicate the set of all idle periods for the i-th qubit. Notice that S_{T_1} accounts for decoherent errors during idle periods only. Decoherent errors during gate operations are captured by S_{gate}.

The next error we account for is coherent error due to ZZ crosstalk for connected and mutually idle qubits, and is modeled via S_{ZZ}. In cases where qubits are active, the ZZ interaction with their neighbors is suppressed due to the inherent echo structure of the driving pulses.

The ZZ interaction term for a pair of qubits i, j is $H_{ZZ} = (\omega_{ZZ})_{ij} Z_i Z_j/2$. In terms of the computational basis states for the i, j qubits, the effect of the interaction is to induce an overall phase shift, with the sign of the phase dependent on the parity of the state:

$$|\psi_e\rangle \to e^{-i(\omega_{ZZ})_{ij} t/2} |\psi_e\rangle\,, \qquad |\psi_o\rangle \to e^{i(\omega_{ZZ})_{ij} t/2} |\psi_o\rangle\,, \tag{13}$$

where $|\psi_e\rangle$ represents an arbitrary linear combination of the even-parity states $|00\rangle, |11\rangle$, and $|\psi_o\rangle$ represents an arbitrary linear combination of odd-parity states $|01\rangle$, $|10\rangle$. ZZ coherent errors can be suppressed by applying context-aware dynamical decoupling (DD) [14]. While a perfect timing of DD sequence can eliminate the error completely, a sub-optimal timing leads to a mismatch duration for which the effect of ZZ crosstalk is not canceled.

As with the T_1 error, the ZZ error is state-dependent, and therefore we perform a Haar average to obtain a general expression that can be used to estimate the ZZ fidelity for arbitrary idle blocks within a given circuit:

$$f_{ZZ}(\Delta t; (\omega_{ZZ})_{ij}, c) = 1 - c \, \sin^2\left(\frac{(\omega_{ZZ})_{ij} \Delta t}{2}\right)\,, \tag{14}$$

In the above expression, Δt represent the ZZ mismatch duration in a given mutual idling period. In the absence of DD, the mismatch duration is identical to the full idling duration while in the presence of an optimal DD the mismatch duration would go to zero.

Haar-averaging corresponds to setting $c = 2/3$; however, as with the a, b constants in the T_1 score, c will be promoted to a learnable parameter in the machine learning model. As before, to obtain a circuit-wide fidelity metric, we assume that the errors act independently for each pair of connected qubits and for each separate mutual idle period:

$$S_{ZZ} = \prod_{(i,j)\in G} \prod_{\Delta t \in \text{ idle intervals}(i,j)} f_{ZZ}(\Delta t; (\omega_{ZZ})_{ij}, c)\,. \tag{15}$$

Here, G is the device coupling graph (undirected), and the outer product is over pairs of connected qubits. The inner product is over periods where both qubits in question are idle.

The final subset of parameters to consider are the powers p_a, which are introduced to give the model additional flexibility. There is a redundancy associated with these, as the ranking induced by S is invariant under rescaling by a positive constant, i.e.

$$(p_{\text{gate}}, p_{\text{msmt}}, p_{T_1}, p_{ZZ}) \to \lambda\,(p_{\text{gate}}, p_{\text{msmt}}, p_{T_1}, p_{ZZ})\,, \tag{16}$$

with $\lambda > 0$. This motivates the identification of two sets of powers if they differ by an overall positive scale. This may be accomplished by parameterizing the powers in a manner that allows the overall scale to be fixed; thereby removing the redundancy. A natural choice is to introduce spherical coordinates over the $\mathbb{R}^4$ spanned by the power parameters p_a and to fix the radius to be 1, which has the effect of restricting the parameters to the unit S^3. For example, using Hopf coordinates yields

$$p_{\text{gate}} = \cos\xi_1 \sin\eta\,, \tag{17a}$$

$$p_{\text{msmt}} = \sin\xi_1 \sin\eta\,, \tag{17b}$$

$$p_{T_1} = \cos\xi_2 \cos\eta\,, \tag{17c}$$

$$p_{ZZ} = \sin\xi_2 \cos\eta\,. \tag{17d}$$

Technically, we are only interested in the region of S^3 where all powers are positive, which restricts the angular ranges to $\xi_{1,2} \in [0, \pi/2]$, $\eta \in [0, \pi/2]$ (the full sphere may be parameterized using $\xi_i \in [0, 2\pi]$).

With the exception of the power p_a and the Haar-averaged constants a, b, c, the parameters appearing in the score function are related to device physics and can be estimated via calibration routines. In our

machine learning approach, these are promoted to be learnable parameters that are fit by minimizing a loss function over a dataset, Eq. 19. However, an important point is that the device-specific parameters naturally vary with time (hence the need to routinely re-estimate them). To account for the fact that the dataset consists of circuit data executed during a period lasting several months, we implemented a parameterization of the circuit score that differs slightly from the above presentation. Parameters that we expect to vary over time, such $f(g)$ (the gate fidelity for gate g), were re-parameterized as $f(g) = f_t(g)^{\lambda(g)}$, where $f_t(g)$ is the time-dependent value as reported by the backend, and where λ is the parameter fit in the training step.

Lastly, for reference, the Mapomatic score function is:

$$S_{\text{Mapomatic}}(C) = \prod_g f(g)^{n(C;g)} \prod_{i=0}^{n-1} f(\text{msmt}_i)^{n(C;\text{msmt}_i)}\,, \tag{18}$$

where the first product runs over all native gates g and the second product runs over the qubits. Here $f(o)$, $n(C;o)$ are the fidelity and count of the operation o, respectively, and $f(\text{msmt}_i)$ corresponds to one minus the mean miss-classification rate for qubit i. In this expression, both the gate and measurement fidelities are automatically estimated by IBM several times a day using common QCVV methods and obtained directly from the backend.

B Loss functions

Given a parametric circuit score S, the learning task is to perform an optimization of the model parameters θ so that the ranking induced by the score closely approximates the empirical ranking provided by the Hellinger fidelity. This is an example of a Learning to Rank problem, which has been well-studied in the machine learning literature [26]. The basic strategy is to introduce a loss function $\mathcal{L}(\theta)$ that quantifies the extent to which the ranking induced by the circuit score S differs from the "true" ranking, which here we take to be given by the Hellinger fidelity H_F. Then, this loss function is averaged over a training dataset $\mathcal{D}_{\text{train}}$, and then minimized with respect to the model parameters:

$$\theta_* = \text{argmin}_\theta\, \mathcal{L}(\theta)\,. \tag{19}$$

Here, θ schematically refers to the full vector of model parameters. We considered multiple loss functions for training the circuit score (summarized in Table 1). These can be categorized according to the nature of the comparisons being made. Pointwise losses do not involve any comparisons among different layouts (i.e., the loss is computed on a per-layout basis), pairwise losses involve comparisons between two layouts, and listwise losses involve comparisons across the set of all layouts for a given initial circuit.

The sole pointwise loss function considered here is the Mean Squared Error (MSE) between the empirical Hellinger fidelity and the circuit score:

$$\mathcal{L}_{\text{Score MSE}}(\theta) = \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{C\in\mathcal{D}_{\text{train}}} \left(S(C) - H_F(C)\right)^2\,. \tag{20}$$

This loss encourages the circuit score to match the fidelity. While straightforward, this score MSE loss might be too strong of an objective, given that the goal is for the score to merely reproduce the ordinal ranking induced by the Hellinger fidelity, as opposed to the precise numerical value for each circuit. In particular, any monotonic transformation of the circuit score will leave the ranking unchanged. This motivates loss functions that compare two or more layouts.

A common pairwise approach is to use the binary cross entropy (BCE) loss:

$$\mathcal{L}_{\text{Pairwise BCE}}(\theta) = -\frac{1}{2} \sum_{\substack{C_1,C_2\in\mathcal{D}_{\text{train}} \\ :C_1\neq C_2}} [[H_F(C_i) < H_F(C_j)]] \ln P_{ij}\,. \tag{21}$$

Here $[[x]]$ is the Iverson bracket (evaluating to 1 if x is true and 0 otherwise), and P_{ij} is the probability under the model that the circuit j has a higher Hellinger fidelity than circuit i. This is commonly modeled as the logistic sigmoid of the score difference [28],

$$P_{ij} = \frac{1}{1 + \exp\left(-(S(C_i) - S(C_j))\right)} \,. \tag{22}$$

As discussed in the main text, the pairwise loss necessitates a quadratic expansion of the dataset: an initial dataset of N distinct circuits yields $N(N-1)/2$ distinct pairs of circuits. This expansion can be useful because it allows for many more training examples to be drawn from a given dataset, but it also has drawbacks because the dataset size can become unwieldy as well as depart from being independent and identically distributed (iid) [29]. For these reasons, we did not implement a pairwise loss in our experiments.

The listwise approach to Learning to Rank assigns a scalar loss to an entire set of circuits. This is naturally suited to layout selection given that the problem is to correctly predict the best performing circuit layout out of the set of all possible layouts. In these approaches, the dataset is divided into N_B batches. Each batch consists of the set of layouts derived from a single initial circuit. A listwise approach will then assign a loss value to each batch characterizing the extent to which the rankings induced by the Hellinger fidelity and circuit score differ.

We considered a number of listwise loss functions. The Pearson correlation loss is motivated by the observation that an ideal circuit score will be perfectly correlated with the Hellinger fidelity, and therefore the negative Pearson correlation may be used as a loss function:

$$\mathcal{L}_{\text{Pearson}}(\theta) = -\frac{1}{N_B} \sum_{i=1}^{N_B} r_{H_F,S} \,. \tag{23}$$

Here $r_{x,y} \in [-1, 1]$ is the Pearson correlation coefficient for two sequences x, y, with $r_{x,y} = 1$ corresponding to perfect correlation (sequence indices have been suppressed). Note that the Pearson correlation coefficient is a biased estimator of the true correlation $\rho(X, Y)$.

The closely related Spearman correlation loss is motivated by the observation that, although two monotonically related circuit scores produce the same ranking, the Pearson correlation loss is not invariant under monotonic transformations of the circuit score. The Spearman correlation loss addresses this by replacing the Pearson correlation coefficient with the Spearman rank correlation coefficient $r^{(s)}_{x,y} = r_{R(x),R(y)}$, which is just the Pearson correlation of the ranks of each variable $R(x)$, $R(y)$. This loss is manifestly invariant under monotonic transformations of the circuit score due to the invariance of the rank function. An important technical obstacle to using the Spearman loss is the fact that the rank function is not differentiable, which is a necessary condition for gradient-based optimization algorithms. We surmount this obstacle by computing a "soft" Spearman rank correlation coefficient using the method of [42], which introduced a relaxed definition of the rank function based on projections onto the permutahedron (the convex hull of permutations). This prescription allows the Spearman correlation loss to be optimized using first-order approaches such as gradient descent.

Next, the rank MSE loss function is motivated by the fact that the Spearman rank correlation coefficient between two length-L sequences x and y can be written in terms of the squared difference of ranks:

$$r^{(s)}_{x,y} = 1 - \frac{6 \sum_{i=1}^{L} \left(R(x_i) - R(y_i)\right)^2}{L(L^2 - 1)} \,. \tag{24}$$

Consequently, minimizing the Spearman loss is equivalent to minimizing the MSE of the rank. In many ranking problems, including layout selection, errors among the top-ranked items should be more costly than errors among low-ranked items, which suggests a simple modification of the above:

$$\mathcal{L}_{\text{Rank MSE}}(\theta; d) = \frac{1}{N_B} \sum_{i=1}^{N_B} \sum_{\ell=1}^{L_i} \frac{\left(R(H_F(C_\ell)) - R(S(C_\ell)\right)^2}{R(H_F(C_\ell))^d} \,, \tag{25}$$

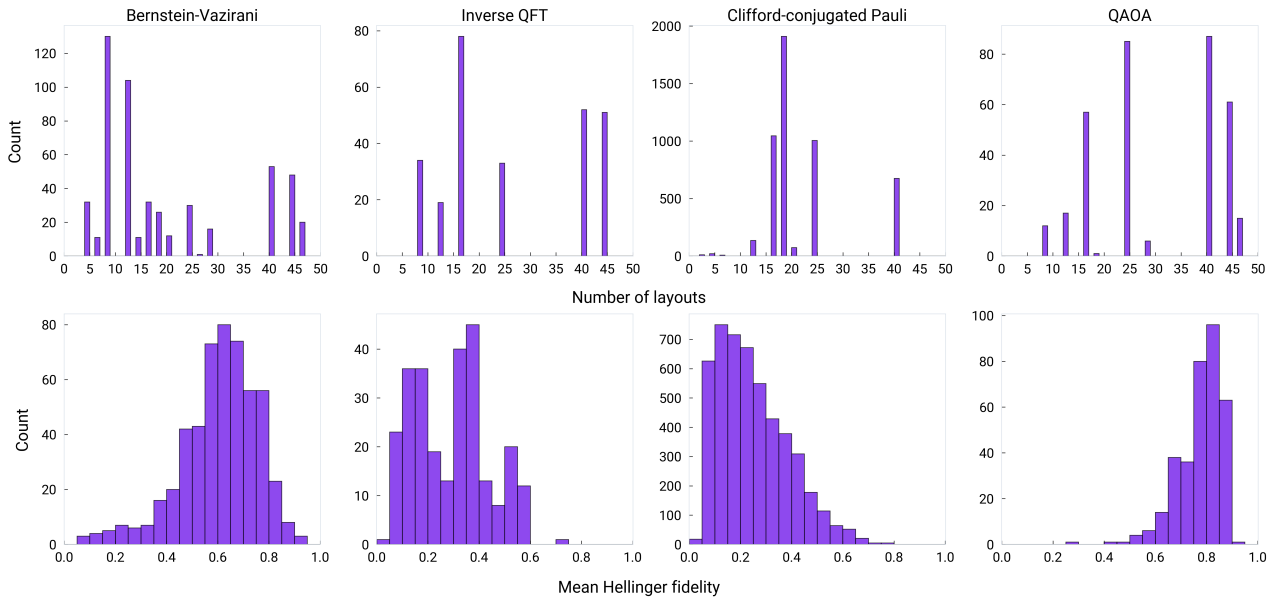

Figure 5: Dataset statistics. (*Top*): The distribution of layout sizes (the number of ways that a given initial circuit can "fit" onto the device after hardware transpilation), according to algorithm. The number of layouts per circuit ranges from 2 to 46. (*Bottom*): The distribution of empirical Hellinger fidelities, also according to algorithm. Note that the details of each algorithm, such as the circuit width n or the random graph used in the QAOA circuit were chosen to produce circuits whose Hellinger fidelities were not extreme (i.e., near 0 or near 1).

where d is a hyper-parameter used to control the extent to which top-rank errors should be over-penalized relative to low-rank errors. The rank MSE loss with $d = 0$ is equivalent to the Spearman loss in that the two loss functions are the same up to linear transformation.

Lastly, another way to bias the loss function towards the top-ranked layouts is to maximize the probability of observing the empirically-observed top-K layouts across all batches, $\prod_{i=1}^{N_B} \Pr(\text{top } K)_i$. This is equivalent to minimizing the negative log-likelihood:

$$\mathcal{L}_{\text{NLL}}(\theta; K) = -\frac{1}{N_B} \sum_{i=1}^{N_B} \ln \Pr(\text{top } K)_i \,. \tag{26}$$

An expression for $\Pr(\text{top } K)_i$ is required to implement this loss. We adopt the Plackett-Luce model, which models this probability as a product of normalized scores [43]:

$$\Pr(\text{top } K) = \prod_{k=1}^{K} \frac{S_{j_k}}{\sum_{\ell \in A_k} S_\ell} \,. \tag{27}$$

Here j_k is the index of the circuit with the k-th empirical Hellinger fidelity and $A_k = A_{k-1} \setminus \{j_k\}$ with $A_1 = \{1, 2, ..., L\}$ is the set of alternatives. To explain the notation with an example, suppose that for a circuit with $L = 4$ layouts $\{C_1, C_2, C_3, C_4\}$ the following ranking of empirical Hellinger fidelities is observed: $H_F(C_2) > H_F(C_4) > H_F(C_1) > H_F(C_3)$. Then

$$\Pr(\text{top } 3) = \left(\frac{S(C_2)}{\sum_{\ell \in \{1,2,3,4\}} S(C_\ell)}\right) \left(\frac{S(C_4)}{\sum_{\ell \in \{1,3,4\}} S(C_\ell)}\right) \left(\frac{S(C_1)}{\sum_{\ell \in \{1,3\}} S(C_\ell)}\right) . \tag{28}$$

C Circuit dataset

The circuit ensemble was designed with the intention of being representative of the types of circuits that users of near-term quantum devices are interested in running and encompassing a diverse set of

circuit contexts. For this approach to scale to larger devices, an additional criterion is that the circuits should have a known output or should be efficiently simulable, so that the ideal bitstring distribution, and hence the Hellinger fidelity, may be classically computed. This was accomplished by restricting attention (with one exception) to circuits whose ideal, non-noisy output state is guaranteed to be a single computational basis state, i.e. $U(C)|0\rangle = |i\rangle$, $i \in \{0, 1, ..., 2^n - 1\}$, with $U(C)$ the unitary implemented by the circuit C. Such "one-hot" or deterministic circuits are appealing because the Hellinger fidelity corresponds to the circuit success probability.

The Bernstein-Vazirani, Inverse QFT, and Clifford-conjugated Pauli circuits all satisfy the above criteria. The dataset was augmented with a fourth algorithm, QAOA [34], that does not satisfy these criteria. This choice was made to improve the relevance of the dataset to circuits of interest to users of near-term noisy devices today. However, it is important to note that the QAOA circuits are not efficiently simulable, and therefore this augmentation approach will not scale to larger devices.

As described in the main text, we generated such a dataset for the *ibmq_ guadalupe* device, consisting of 6020 initial circuits, 81% of which are the Clifford-conjugated Pauli circuits, 9% Bernstein-Vazirani, 6% QAOA, and 4% inverse QFT. After accounting for all allowed layouts, these 6020 initial circuits became 132,866 circuits after hardware transpilation. These specific proportions are chosen to ensure that a wide diversity of circuit contexts is present in the dataset and to avoid identical or close-to-identical circuits in the data set. Highly structured algorithms such as BV or QFT transpile to different circuits as the qubit count n is varied or due to the variations in the transpilation procedure. However, given that the dataset corresponds to circuits executed on a 16-qubit device, the number of distinct transpiled circuits for these structured algorithms is highly limited. Simply put, it is not possible to obtain a large set of unique transpiled BV circuits for $n \leq 16$. Consequently, the dataset is dominated by random circuits which can naturally generate a greater diversity of circuit contexts. Moving forward, larger and better-quality devices should allow for greater flexibility in designing the circuit ensemble.

Lastly, in Fig. 5 we show the distribution of layouts (per initial circuit) according to algorithm, as well as the distribution of mean Hellinger fidelities for each initial circuit.

References

[1] IBM. "Ibm quantum".

[2] Rigetti. "Rigetti computing".

[3] Amazon. "Amazon braket".

[4] Joseph Emerson, Robert Alicki, and Karol Życzkowski. "Scalable noise estimation with random unitary operators". Journal of Optics B: Quantum and Semiclassical Optics **7**, S347 (2005).

[5] Emanuel Knill, Dietrich Leibfried, Rolf Reichle, Joe Britton, R Brad Blakestad, John D Jost, Chris Langer, Roee Ozeri, Signe Seidelin, and David J Wineland. "Randomized benchmarking of quantum gates". Physical Review A **77**, 012307 (2008).

[6] Sergio Boixo, Sergei V Isakov, Vadim N Smelyanskiy, Ryan Babbush, Nan Ding, Zhang Jiang, Michael J Bremner, John M Martinis, and Hartmut Neven. "Characterizing quantum supremacy in near-term devices". Nature Physics **14**, 595–600 (2018).

[7] Charles Neill, Pedran Roushan, K Kechedzhi, Sergio Boixo, Sergei V Isakov, V Smelyanskiy, A Megrant, B Chiaro, A Dunsworth, K Arya, et al. "A blueprint for demonstrating quantum supremacy with superconducting qubits". Science **360**, 195–199 (2018).

[8] Timothy Proctor, Kenneth Rudinger, Kevin Young, Mohan Sarovar, and Robin Blume-Kohout. "What randomized benchmarking actually measures". Physical review letters **119**, 130502 (2017).

[9] Timothy Proctor, Kenneth Rudinger, Kevin Young, Erik Nielsen, and Robin Blume-Kohout. "Measuring the capabilities of quantum computers". Nature Physics **18**, 75–79 (2022).

[10] S. Mavadia, C. L. Edmunds, C. Hempel, H. Ball, F. Roy, T. M. Stace, and M. J. Biercuk. "Experimental quantum verification in the presence of temporally correlated noise". npj Quantum Information **4**, 7 (2018).

[11] Andre R. R. Carvalho, Harrison Ball, Michael J. Biercuk, Michael R. Hush, and Felix Thomsen. "Error-robust quantum logic optimization using a cloud quantum computer interface". Phys. Rev. Appl. **15**, 064054 (2021).

[12] C. L. Edmunds, C. Hempel, R. J. Harris, V. Frey, T. M. Stace, and M. J. Biercuk. "Dynamically corrected gates suppressing spatiotemporal error correlations as measured by randomized benchmarking". Phys. Rev. Res. **2**, 013156 (2020).

[13] Jaseung Ku, Xuexin Xu, Markus Brink, David C. McKay, Jared B. Hertzberg, Mohammad H. Ansari, and B. L. T. Plourde. "Suppression of unwanted zz interactions in a hybrid two-qubit system". Phys. Rev. Lett. **125**, 200504 (2020).

[14] Pranav S Mundada, Aaron Barbosa, Smarak Maity, Yulun Wang, Thomas Merkh, TM Stace, Felicity Nielson, Andre RR Carvalho, Michael Hush, Michael J Biercuk, et al. "Experimental benchmarking of an automated deterministic error-suppression workflow for quantum algorithms". Physical Review Applied **20**, 024034 (2023).

[15] Siyuan Niu, Aida Todri-Sanial, and Nicholas T. Bronn. "Multi-qubit dynamical decoupling for enhanced crosstalk suppression" (2024). arXiv:2403.05391.

[16] Harrison Ball, Thomas M. Stace, Steven T. Flammia, and Michael J. Biercuk. "Effect of noise correlations on randomized benchmarking". Phys. Rev. A **93**, 022303 (2016).

[17] Timothy Proctor, Stefan Seritan, Kenneth Rudinger, Erik Nielsen, Robin Blume-Kohout, and Kevin Young. "Scalable randomized benchmarking of quantum computers using mirror circuits". Physical Review Letters **129**, 150502 (2022).

[18] Victoria Zhang and Paul D Nation. "Characterizing quantum processors using discrete time crystals" (2023). arXiv:2301.07625.

[19] Giovanni Acampora and Roberto Schiattarella. "Deep neural networks for quantum circuit mapping". Neural Computing and Applications **33**, 13723–13743 (2021).

[20] Yikai Mao, Shaswot Shresthamali, and Masaaki Kondo. "Quantum circuit fidelity improvement with long short-term memory networks" (2023). arXiv:2303.17523.

[21] Daniel Hothem, Kevin Young, Tommie Catanach, and Timothy Proctor. "Learning a quantum computer's capability". IEEE Transactions on Quantum Engineering **5**, 1–26 (2024).

[22] Daniel Hothem, Jordan Hines, Karthik Nataraj, Robin Blume-Kohout, and Timothy Proctor. " Predictive Models from Quantum Computer Benchmarks ". In 2023 IEEE International Conference on Quantum Computing and Engineering (QCE). Pages 709–714. Los Alamitos, CA, USA (2023). IEEE Computer Society.

[23] Evan Peters, Prasanth Shyamsundar, Andy CY Li, and Gabriel Perdue. "Qubit assignment using time reversal". PRX Quantum **3**, 040333 (2022).

[24] Daniel Silver, Tirthak Patel, and Devesh Tiwari. "Qompose: A technique to select optimal algorithm-specific layout for neutral atom quantum architectures" (2024). arXiv:2409.19820.

[25] Irfansha Shaik and Jaco van de Pol. "Optimal layout synthesis for quantum circuits as classical planning". In 2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD). Pages 1–9. IEEE (2023).

[26] Tie-Yan Liu et al. "Learning to rank for information retrieval". Foundations and Trends® in Information Retrieval **3**, 225–331 (2009).

[27] Hang Li. "A short introduction to learning to rank". IEICE TRANSACTIONS on Information and Systems **94**, 1854–1862 (2011).

[28] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. "Learning to rank using gradient descent". In Proceedings of the 22nd international conference on Machine learning. Pages 89–96. (2005).

[29] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. "Learning to rank: from pairwise approach to listwise approach". In Proceedings of the 24th international conference on Machine learning. Pages 129–136. (2007).

[30] Luigi P Cordella, Pasquale Foggia, Carlo Sansone, and Mario Vento. "A (sub) graph isomorphism algorithm for matching large graphs". IEEE transactions on pattern analysis and machine intelligence **26**, 1367–1372 (2004).

[31] Alpár Jüttner and Péter Madarasi. "Vf2++—an improved subgraph isomorphism algorithm". Discrete Applied Mathematics **242**, 69–81 (2018).

[32] Ethan Bernstein and Umesh Vazirani. "Quantum complexity theory". In Proceedings of the twenty-fifth annual ACM symposium on Theory of computing. Pages 11–20. (1993).

[33] Sergey Bravyi and Dmitri Maslov. "Hadamard-free circuits expose the structure of the clifford group". IEEE Transactions on Information Theory **67**, 4546–4563 (2021).

[34] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. "A quantum approximate optimization algorithm" (2014). arXiv:1411.4028.

[35] Paul D Nation and Matthew Treinish. "Suppressing quantum circuit errors due to system variability". PRX Quantum **4**, 010327 (2023).

[36] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization" (2014). arXiv:1412.6980.

[37] Leslie N Smith and Nicholay Topin. "Super-convergence: Very fast training of neural networks using large learning rates". In Artificial intelligence and machine learning for multi-domain operations applications. Volume 11006, pages 369–386. SPIE (2019).

[38] Thomas Lubinski, Sonika Johri, Paul Varosy, Jeremiah Coleman, Luning Zhao, Jason Necaise, Charles H Baldwin, Karl Mayer, and Timothy Proctor. "Application-oriented performance benchmarks for quantum computing". IEEE Transactions on Quantum Engineering (2023).

[39] Teague Tomesh, Pranav Gokhale, Victory Omole, Gokul Subramanian Ravi, Kaitlin N Smith, Joshua Viszlai, Xin-Chuan Wu, Nikos Hardavellas, Margaret R Martonosi, and Frederic T Chong. "Supermarq: A scalable quantum benchmark suite". In 2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA). Pages 587–603. IEEE (2022).

[40] G. Hartnett, A. Barbosa, P. S. Mundada, M. Hush, M. Biercuk, and Y. Baum. "Learning to rank quantum circuits for hardware-optimized performance enhancement". Zenodo (2024).

[41] Philip Krantz, Morten Kjaergaard, Fei Yan, Terry P Orlando, Simon Gustavsson, and William D Oliver. "A quantum engineer's guide to superconducting qubits". Applied physics reviews **6**, 021318 (2019).

[42] Mathieu Blondel, Olivier Teboul, Quentin Berthet, and Josip Djolonga. "Fast differentiable sorting and ranking". In International Conference on Machine Learning. Pages 950–959. PMLR (2020).

[43] Robin L Plackett. "The analysis of permutations". Journal of the Royal Statistical Society Series C: Applied Statistics **24**, 193–202 (1975).