Predicting the fidelity of Quantum Circuits

Diogo Valada
diogovalada@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

October 2020

Abstract

The endeavor to build a working quantum computer is highly interdisciplinary in nature, requiring a complex software-hardware stack. In particular, given the hardware-agnostic properties of common quantum algorithms, a compiler is required to transform those algorithms into code runnable by the specific quantum devices. Such compilers need to take into account the architectural constraints of the said devices, such as the fact that quantum devices do not commonly offer all-to-all connectivity. This results oftentimes on gate number and circuit depth overhead. This can become problematic, since in the current NISQ (Noisy Intermediate-Scale Quantum) era, quantum devices are characterized by high error rates and reduced number of qubits, making quantum error correction techniques unpractical: this requires the compilation procedure to be as efficient as possible. In this work, we approach the qubit mapping problem of compilation, where the software's virtual qubits are assigned to the device's physical qubits: in order to leverage the power of the existing mapping algorithms, both existing and new metrics are developed and tested in this work. Among these, we propose a deep learning-based metric that is able to outperform all the previously proposed ones for some types of mapping algorithms. **Keywords:** Quantum computing; Qubit Mapping; Mapping metrics; Fidelity prediction; Deep Learning.

1. Introduction

Great strides have been made in developing a functional quantum computer, with several possible implementation technologies in development. However, all of the technologies still only support a small number of qubits, the basic unit of quantum information, meaning that only small programs can be executed. Furthermore, most implementations are plagued by short decoherence times (introducing low success rates for long circuits) and imperfect quantum operations (making circuits with many gates more prone to error). The main strategy to solve the errors introduced by qubit decoherence and faulty operations is known as Quantum Error Correction (QEC)[1], which proposes encoding one logical qubit using several physical qubits. This solution is however out of range for current devices, known as Noisy Intermediate-Scale Quantum (NISQ) devices, due to the small number of available qubits.

There exists a gap between quantum software designed with the circuit model (the most common quantum computation model) and quantum hardware. One such factor creating this gap is the fact that do not commonly exhibit all-to-all qubit connectivity, which quantum programs tend to assume. Such connections between pairs of qubits are however necessary to perform two-qubit gates, which are an important element of quantum computation. In order to overcome this, it is necessary to map the program's virtual qubits to the hardware's physical qubits, and change them dynamically throughout the computation, which typically incurs in computation overhead in the form of extra gates and longer duration. It is however necessary to minimize this overhead as much as possible, since it can lower the success probability of the computation, defined as the likelihood of measuring the correct outcome, due to the frailty of qubits and faulty gates. This dilemma is known as the Qubit Mapping problem, and is known to be NP-complete[2]. Due to the complexity of this problem, smart algorithms are necessary in order to be able to solve it. Additionally, due to the difficulty in predicting the success probability of a circuit, good metrics which correctly estimates or correlates with this quantity are also necessary, in order to drive the mapping algorithms to the best solutions. In this work, we summarize the existing metrics used to drive these algorithms, present a novel way to compare them in a algorithm independent and unbiased way. Additionally, we proposed novel metrics models, from which we highlight deep learning-based models capable of not only correlating better with the success probability than previous metrics, but also accurately estimating it.

2. Background

Qubits. In Classical Information, the basic unit of information is the bit, which has two possible states, '0' and '1'. In Quantum Information, on the other hand, the basic unit is a quantum bit, or *qubit*, which has two base states, commonly represented as $|0\rangle$ and $|1\rangle$. The qubit can also be in a state which is any linear combination of the two basis states, a property called *superposition*. An arbitrary qubit state $|\psi\rangle$ is therefore represented (in a specific basis) as $|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$, where $\alpha, \beta \in \mathbb{C}$ with $|\alpha|^2 + |\beta|^2 = 1$. Multi-qubit states can be constructed, and are represented by a 2^n dimensional vector, where n is the number of involved qubits, namely $|\phi\rangle = \alpha_0|000...0\rangle + \alpha_1|000...1\rangle + ... + \alpha_{2^n}|111...1\rangle$. If the multi-qubit state is non-factorable in individual qubit states, it is said that the state is *entangled*. The space of all qubit states is, therefore, a Hilbert space. Quantum states are acted upon using Quantum Operations, which are split into three types:

State preparations. State preparations change the state of the qubit into a known, useful one. Usually, state preparations leave qubit in one of the computational basis states.

Quantum Gates. Quantum gates are reversible, non-destructive, unitary operations applied to qubits. They change the qubit quantum state without making it collapse to the computational basis. Single-qubit gates (such as the X, Y, Z gates) act on just one qubit and can be understood as rotations around a given axis in the qubit's Hilbert space. Multi-qubit gates, on the other hand, may act on several qubits, and can be viewed as rotations in higher dimensional space. A particularly relevant subset of these are constituted by Two-qubit gates (like the CNOT, CZ gates), gates which act on two qubits.

Measurements. These operations are needed in order to gather information about a quantum state. However, unlike quantum gates, this operation is non-reversible, since the quantum state, and therefore the encoded information, is lost upon execution. Measurement makes the quantum state collapses into one of the basis states.

Quantum algorithms are commonly described as quantum circuits, representing qubits as horizontal lines, and quantum operations as blocks on those lines (exemplified in Fig. 1) and are hardware-agnostic, i.e. they do not take into consideration possible limitations present in the physical implementation of the qubits (quantum chip). Quantum circuits can be divided in *cycles*(timesteps), columns containing operations being performed in parallel.

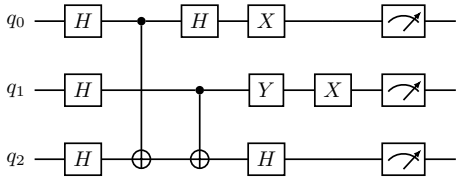


Figure 1: Example of a quantum circuit. It contains 5 qubits (q_0 to q_4), single-qubit gates (X, H) and two-qubit gates (S and CNOT, the latter being identified by the $\oplus$ symbol). The meter symbols on the far right represent measurements. The first gates on each qubit (H) are examples of parallel gates, being performed in the same cycle.

2.1. Physical Implementations

Unlike classical computers, whose implementation mainly relies on silicon transistors, quantum computers have several classes of candidate technologies under research, such as Superconducting qubits[3], Semiconducting qubits[4]. Physical implementation tend to exhibit three main limitations:

Qubits are fragile. Noise from the environment can induce unwanted state transitions and phase shifting. To be able to perform coherent computation on the qubits, they must be shielded against this environmental noise. Furthermore, phenomena such as relaxation limit the useful lifetime of a qubit state in a fundamental way.

Quantum operations are faulty. Additionally, the operations applied by quantum devices in their qubits are not the same as the theoretical ones. They are imperfect and may introduce errors in the computation. The reliability of the gates can be quantified using a concept call *Fidelity*: given a quantum operation, or a set of quantum operations (such as a quantum circuit), fidelity is a measure of distance in the Hilbert Space, namely the distance between the expected quantum state (assuming error-free computation) and the obtained state. The fidelity of a gate set can be obtained via tomographic techniques such as Gate Set Tomography (GST)[5].

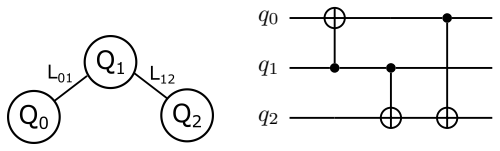
Qubit connectivity In order to perform multi-qubit gates, the relevant qubits need to be physically connected in such a way that it is possible for their quantum wave functions to interfere. For example, in the case of superconducting qubits this physical connection is often a photonic resonator. The most commonly implemented type of multi-qubit gate is two-qubit gates. Each of these gates requires a control operand qubit and a target operand qubit. Unlike the quantum circuit abstraction, quantum chips usually don't have all-to-all connectivity between qubits (a few exceptions apply, such as Trapped Ions in certain conditions).

That is, the qubit connectivity graph is not a complete graph, since such a feature poses great engineering challenges.

2.2. Qubit Mapping: Problem statement

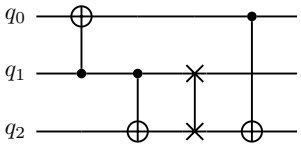
Quantum computers consist of different software and hardware layers that bridge the gap between quantum applications and quantum chips. In particular, a compiler is required to transform the hardware-agnostic quantum algorithms into quantum circuits runnable by the hardware. In this context, the Qubit Mapping Problem arises: the complete qubit connectivity graph assumed by hardware-agnostic quantum circuits needs to be emulated by the hardware's non-complete connectivity graph. In order to do this, one needs to smartly map the virtual qubits to the physical qubits at the beginning of the computation (known as Initial Placement), and possibly dynamically change this mapping during the computation (known as Qubit Routing). The Mapping Problem includes these two procedures.

An example of this problem is portrayed in Fig. 2. With the initial mapping $\{q_0 : Q_0, q_1 : Q_1, q_2 : Q_2\}$ the connectivity constraints for the first two two-qubit gates were respected, but not for the third one. In order to solve this, SWAP gates can be introduced, exchanging the quantum states of its operands, resulting in the mapping $\{q_0 : Q_0, q_1 : Q_2, q_2 : Q_1\}$. This last mapping now makes q_0 and q_2 adjacent on the device, enabling the execution of the third two-qubit gate.



(a) Connectivity graph.

(b) Unrouted circuit.



(c) Routed circuit.

Figure 2: Instance of the Qubit Mapping problem. In order to make the circuit in Fig. 2(b) runnable in the presented connectivity graph, a SWAP gate has to be added.

This mapping procedure results, specifically the Qubit Routing procedure may however introduce overhead in the computation, namely in the form of extra gates and longer duration. Given that qubits have a limited lifetime and quantum gates are faulty, this overhead will decrease the computation's success probability. It is therefore important to reduce this overhead as much as possible, resulting in the need for good mapping algorithms.

Additionally, the success probability of a quantum circuit is not easily modelable, due to the complex noise behaviour of quantum systems. As such, the development of appropriate metrics to guide the mapping algorithms is also necessary.

2.3. Mapping Algorithms/Metrics: State of the Art

Due to complexity aspects, finding an optimal solution to the qubit mapping problem quickly becomes an impossible task with a growing number of qubits and gates. For this reason, despite exact solvers like SMT-based[6] having been proposed for very small instances ($\lesssim 5$ qubits), scalable algorithms are required.

In order to make the problem tractable, two approaches are usually taken: (i) approximate algorithms are used; (ii) routing algorithms split the problem into smaller, more digestible problems. This is typically done at the quantum circuit level, by slicing it into smaller sections, usually up to 50 cycles long, which are more easily optimizable. Most algorithms then try to populate those slices with SWAP gates, such that the connectivity constraints for all gates are respected.

Some approaches look for a good set of SWAPs to be introduced in each slice via searches using dedicated algorithms[7, 8]. Other based use heuristic search algorithms such as A*[9, 2] and apply techniques such as bidirectional circuit mapping procedures[2], or use Spectral Graph Theory in order to discover the good mappings for each slice[10], and generate sets of SWAPs accordingly. Finally, the usage of other frameworks such as Temporal Planning and Constraint Programming has also been proposed[11].

As metric, due to the difficulty in modelling the Success Probability of circuits, the mentioned works tend use *circuit size* (the circuit's number of gates), *circuit depth* (the number of cycles in a circuit) or the number of two-qubit gates, which all negatively correlate with the success probability. A reduced amount of works try instead to model the success probability, whose proposals are formalized in Nishio et al[12], with the Estimated Probability of Success (ESP):

$$ESP(C) = \prod_{g \in C} (1 - \epsilon_g) = \prod_{g \in C} (f_g) \tag{1}$$

where C is an arbitrary quantum circuit, g a gate, ϵ_g the error rate associated with gate g, and f_g the corresponding fidelity. The fidelities can be extracted from the device's calibration data.

3. Models

The Quantumsim simulator[13] was the backend used in order to test the models. Due to the com-

plexity of performing a through GST procedure, the fidelities for each gate were not averaged for all states, but were instead extracted from the state yielding the worst case scenario. Additionally, the following assumptions were made:

- Qubit initializations are assumed to be perfect;
- Measurement operations are not included since running the same circuit thousands of times in order to obtain the output distribution would be computationally expensive, due to the fact that a simulator was used.
- CZ was the considered two-qubit gate when developing the models, since the native two-qubit gate that the backend supports. This gate is operand-symmetric, unlike the CNOT, for instance.

Four variations of the ESP model are proposed, along with other analytical models that also make use of the reliability data of devices such as gate fidelities:

- **ESP^0:** This is the variant implemented in Nishio et al[12]. It considers errors provenient solely from **explicit** single- and two-qubit gates. As such, it does not take into account the implicit idling gates – that is, the waiting periods during which the qubits are not active as a result of scheduling. Additionally, instead of defining fidelities on a per-gate basis, a single fidelity values is define for every gate within each gate type (gate types referring to whether each gate is a single- or two-qubit gate).
- **ESP^i:** Similar to ESP^0, but also takes into account the impact of the **implicit idling** gates.
- **ESP$^0_{sr}$:** Similarly to ESP^0, this variant does not consider implicit idling gates. However, it accounts for the information regarding the **specific rotation** of each gate, instead of solely considering the gate type. Although making no difference in the two-qubit gates case (given that typically a lone two-qubit gate is implemented per device), this approach assigns a different fidelity for each single-qubit gate.
- **ESP$^i_{sr}$:** Considers both the **implicit idling** gates and the **specific rotation** of each gate.

3.1. Tracked Estimated Success Probability (TESP)

The ESP models take into account device specific data such as gate fidelities, which was not the case for the previous metrics (circuit depth, circuit size and number of two qubit gates). Two additional models, named TESP1 and TESP2, where TESP stands for Tracked Estimated Success Probability, were developed using a different perspective and assumptions assumptions regarding how errors affect and propagate through quantum circuits.

These models attempt to track reliability on a per-qubit basis, until the end of the circuit where these can be merged into a single success probability value. In these models, it is assumed that the qubits are flawlessly prepared in the computational basis (usually in the $|0\rangle$ state), i.e., their initial fidelity is 1. The decoupled, per-qubit reliability after idling or a single-qubit gate is calculated for both models through the following expressions:

$$\text{TESP}_q^0 = 1.0, \quad \text{(Initial prob. succ.)} \tag{2}$$

$$\text{TESP}_q^{t+1}, = \text{TESP}_q^t, \times f_{idle} \quad \text{(Idling gates)} \tag{3}$$

$$\text{TESP}_q^{t+1} = \text{TESP}_q^t \times f_{1qbg} \quad \text{(Single-qubit g.)} \tag{4}$$

where TESP_q^t represents the reliability of an arbitrary qubit q at timestep t and f_{idle}, f_{1qbg} represent the reliability of idle- and single-qubit gates, respectively. As for the two-qubit gates, the rules differ for each of the models:

$$\text{TESP1}_{q_c}^{t+1} = \text{TESP1}_{q_c}^t \times f_{2qbg}, \tag{5}$$

$$\text{TESP1}_{q_t}^{t+1} = \text{TESP1}_{q_t}^t \times f_{2qbg}, \tag{6}$$

$$\text{TESP2}_{q_c}^{t+1} = \text{TESP2}_{q_c}^t \times \text{TESP2}_{q_t}^t \times f_{2qbg} \tag{7}$$

$$\text{TESP2}_{q_t}^{t+1} = \text{TESP2}_{q_c}^t \times \text{TESP2}_{q_t}^t \times f_{2qbg} \tag{8}$$

where q_c (q_t) represents the control (target) qubit, and f_{2qbg} represents the reliability of a two-qubit gate. The final step is merging of the per-qubit reliabilities into a final success probability to be used as the models' score:

$$\text{TESP}^T = \prod_q TESP_q^T \tag{9}$$

that is, the *TESP* value for a given circuit is the product of the values for each qubit at the last timestep (timestep T).

As can be seen from the expressions above, the TESP models differ on the assumption used to model the effect of two qubit gates. TESP1 assumes two-qubit gates to impact each of the operands equally, i.e. they both receive an error that is relative to the gate's fidelity and their reliability value. TESP2, on the other hand, attempts to assume the worst possible case of the effect of gates on qubits, making the resulting reliability of each operand a result of not only its initial reliability and the gate's fidelity, but also of the other operand's reliability. This model, based on [14], aims to reflect the worst case effect of gates (including idling gates) on the reliability of the qubits.

In the TESP1 case, the expressions can be implemented directly. In the TESP2 model case, however, due to floating point precision issues the displayed set of expressions had to be implemented using an arbitrary precision type, instead

of a *double* floating point number, and in logarithmic space, using the expression $\log(a \times b) = \log(a) + \log(b)$.

3.2. Deep Learning-based Models (NNESP)

Data-based models were an especially enticing type of model to apply to this problem, due to the complexity and variability of quantum devices: noise models can vary wildly even between devices from the same technology class. Data-based models may have the ability to learn these behaviours on a per-device basis, saving development time and costs. For the problem at hand, it is possible to generate an arbitrary number of random quantum circuits (input data) and run them in the quantum device in question in order to retrieve their probability of success (output data/labels), making it possible to tackle this problem in a supervised learning context. For further research in Deep Learning (DL), [15] is a recommended source.

Quantum circuits are essentially time-ordered schedules where each operation is assigned to a specific qubit on a specific timestamp, since the order of the operations matters and cannot be ignored, so quantum circuit structures can be seen as sequential data. For each circuit we can encode each cycle as a feature vector. Each feature vector will serve as data timesteps to be fed to the neural network.

Two deep learning model variants are considered: (i) NNESP, a variant where only gate types and idles are considered, and not the specific rotations they perform, similarly to the the ESP^i model; (ii) NNESP$_{sr}$, a variant where, similarly to ESP$^i_{sr}$, idle gates and the specific gate rotations are considered. Due to the simulation speed constraints, 5-qubit quantum circuits for the test of all models. Additionally, the two-qubit gate that is implemented in our model is be the CZ (Controlled Z, also known as Controlled Phase, or CPHASE), since it is the primitive gate of the device that the backend attempts to model. This allows for further simplification, since CZ is operand-symmetric, i.e. no directionality (control, target) needs to be specified.

For the NNESP model, we implemented the encoding present in Tab. 1. This encoding describes the operations each qubit undergoes in each of the circuit's cycles in an unequivocal manner. The way two qubits are encoded, assigns each two-qubit gate to its operands in an unambiguous manner. As for the NNESP$_{sr}$ model, in order to distinguish the multiple single-qubit gates, a different encoding, displayed in Fig. 2, was defined. This considered each single-qubit gate in the finite set provided by the target backend as a separate class.

For both encodings, we forewent the usage of 0 as class to encode the gates, since it was reserved for padding procedure of the sequential data. The encoding used by the NNESP$_{sr}$ can be seen as lossless, since no information regarding the circuit is lost: the gates and respective are not ambiguously defines, and the circuit's structure (the specific distribution of the gates throughout the circuit) is unequivocally defined via the feature vectors' indexes.

Type of gate	Feature label
Idle	1
Single-qubit	2
i-th Two-qubit gate (in cycle)	$i+2$

Table 1: Implemented integer encoding for the NNESP model, applied to the operations in each of the quantum circuit's cycles (timesteps). This is a suitable encoding in the case that CZ (CPHASE) or other operand-symmetric gate is the lone native two-qubit gate.

Type of gate	Feature label
Idle	1
Single-qubit	$[2, 1+N_{single}]$
i-th Two-qubit gate (in cycle)	$i+1+N_{single}$

Table 2: Lossless integer encoding used for the NNESP$_{sr}$ model. N_{single} represents the number of available single-qubit gates.

In order to be able to feed the encoded data to a neural network, we needed to apply one further transformation, namely encode each timestep in a one-hot manner (instead of the previous integer encoding).

3.2.1 Network Architecture

Recurrent Neural Networks (RNNs) are one of the types of models suited to handle sequential data. This type of neural networks allows the input of sequences of any size (number of timesteps), without the need to change the networks architecture. Furthermore, it also takes into account the order in which the data is fed, whereas simple feed-forward networks allow for no such time encoding. This fact allows it to take into account the circuit's structure when making a prediction.

A simplified overview of the proposed architecture is presented in Fig. 3. The architecture is composed of five fundamental layer blocks:

1. **Masking layer:** Masking allows the network to receive the padded data, while minimizing the impact on the network's performance.
2. **Embedding layers:** This block of layers receives the (one-hot) encoded data as input. It allows the network to convert the used encoding into a possibly more efficient one, which the network learns.
3. **Recurrent layers:** This block of layers is responsible for handling the sequential nature

of the data. There are a few variants of RNNs[16]. For this work Long Short-Term Memories (LSTMs)[17] with trainable initial hidden states[18], due to their expressiveness compared to other models.

4. **Hidden layers:** These layers receive the hidden state given by the recurrent block and outputs another hidden state. While not strictly necessary, this block may improve the networks prediction capability.
5. **Output layer:** Transforms the last hidden state, given by the hidden layers, into our output of interest, a prediction of the circuit's success probability. This layer is composed of a single neuron. A sigmoid activation was chosen in order to bound the output in the $[0, 1]$ interval. Binary Crossentropy (BCE) was used as cost function to train the model.

Additionally, Batch Normalization layers[19] were also added to the model, between blocks as shown in Fig. 3, and between layers of the same block as well (with the exception recurrent layer block), during the training procedure, in order to improve the training performance.

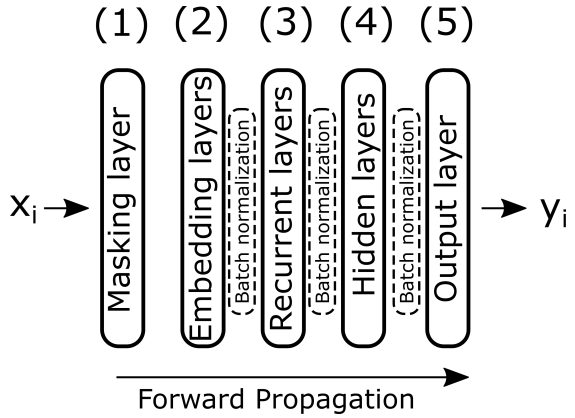


Figure 3: Final outline for the neural network architecture used for both the NNESP and NNESP$_{sr}$ models. The neural network receives each encoded circuit (x_i) and outputs the predicted probability of success (y_i). (2) and (4) might be constituted of several layers, depending on the specific hyperparameters used. The dashed Batch Normalization (BN)[19] layers are a training-specific layer meant to speed the training process up, and are not present in the trained network during inference. BN layers are also present between each layer of the layer blocks (2) and (4), but not between the layer blocks in (3).

In order to train the network, we need a dataset of circuits labelled with their expected probability of success. Since we already have this label for every circuit, and given that it is a single number in the [0,1] interval, no transformations to the label data are required.

4. Results & Discussion

Commonly, articles proposing metrics for the mapping problem tend to evaluate them in a algorithm-dependent fashion. As mentioned, this can result in an evaluation which can be biased by the ability (or lack thereof) of the algorithm to navigate that metric appropriately. Hence, this work proposes a new method to evaluate a given metric.

One of the major roadblocks in accomplishing an algorithm-independent evaluation of metrics is that not all metrics express the same quantity, they just output quantities which aim to correlate with the success probability. In order to compare them, a possible strategy is to look for monotonic correlations between the each of the metrics and the simulated success probability (ground truth). This can be done by ranking the circuits using a pointwise ranking method[20]: it is possible, for each metric, to construct a circuit ranking via the score attributed to each circuit. This then allows the comparison of the different metrics via the comparison of the induced circuit rankings. The resulting rankings can be compared with the ground truth ranking, induced by the obtained success probability, in order to extract a measure of correlation. Specifically, the proposed metric is described by the following steps:

1. Score a set of test circuits (random circuits, for example) using the various metrics to be compared, including the simulated success probabilities;
2. Rank the circuits according to the scores obtained for each metric, the end result being a ranking for each of the metrics;
3. Check which rankings are the closest to the one induced by the simulated success probabilities. This can be quantified using a measure of distance between rankings or rank correlation.

In order to measure the similarity between rankings, the Kendall τ correlation coefficient was used:

$$\tau(A, B) = 1 - \frac{2D_{A,B}}{N_p} = 1 - \frac{2D_{A,B}}{\binom{N}{2}} \tag{10}$$

where $D_{A,B}$ is the number of discordant pairs (pairs of elements appearing in opposite order in each ranking), and $N_p = \binom{N}{2}$ is the total number of pairs possible between any of the ranking's elements, with N being the number of elements in each ranking. A Kendall coefficient of $\tau = 1$ means that two rankings are perfectly correlated (i.e. are equal), $\tau = -1$ represents perfectly negatively correlated circuits.

Using this described method, it is possible to evaluate how good a metric is. Given a dataset with several quantum circuits, rankings are generated by both the candidate metric and the simulated success probability, with the latter acting as ground truth. Afterwards, the Kendall the τ correlation coefficient between the two rankings is calculated, and is used to estimate how good the can-

didate metric is. A value of $\tau = 1$ would mean the metric is perfect in the context of the given dataset.

In order to evaluate the models with the method above, a random circuit generator was created in order to generate probabilistic circuits datasets. Four datasets – A1, A2, B1 and B2 – were considered, varying in the number of samples and maximum circuit depth: the A1/B1 datasets contain circuits with depths up to 50 cycles, while the A2/B2 datasets contain circuits with depths up to 15 cycles. On the other hand, the B1/B2 datasets have 7-10 times as many samples per dataset when compared to their A1/A2 counterparts (which contained approx. 30000/15000 samples, respectively). These allow the evaluation of the models in different depth regimes, in order to verify the impact on algorithms with varying circuit slicing lengths. Additionally the different sizes allow the inspection of the impact of dataset size on the data-driven, deep learning models. Each of these datasets were split into training/validation/testing datasets with a 0.7/0.15/0.15 ratio, respectively. The circuit datasets were simulated using the Quantumsim simulator[13] using the DiCarlo Lab superconducting chip's parameter preset, and a success probability was extracted for each circuit using the Trace Distance between the distributions resulting from the noisy and noiseless simulation of the circuits.

4.1. Model Comparison

The ranking prediction accuracy of each of the presented models is displayed in Tab. 3.

The best faring analytical models were the variants of the Estimated Success Probability, proposed by Nishio et al.[12]. These models were able to beat the other experimental analytical models (TESP1, TESP2) by a reasonable margin.

It can be seen that, for the ESP implementations, taking the idling gates into consideration did marginally improve the predictive power of the model. However, taking the specific gates' errors into account, rather than a generalized per-gate-type error did not yield improvements, actually resulting in an accuracy reduction.

The deep learning models, based on supervised neural networks, did indeed outperform the competing analytical models in all cases. The relative accuracy improvements, as compared to the best analytical model (ESP^i), ranged from minimal improvements of around 2%-5% when considering only gate types, to significant improvements of 10%-18% when distinguishing between single-qubit gates. Regarding the training process, it can be seen that, while not much improvement resulted from a larger dataset in the gate type-only case, it did make a difference in the full-information case.

4.2. Absolute Circuit Scoring

Unlike the previous models, the developed deep learning models are not only able to correlate with the circuits' success probabilities, but are actually able to be good estimators of the absolute values of the success probabilities. As such they may become useful for new use cases which actually require this absolute value prediction.

Fig. 4 is presented comparing, for a sample circuits subset of the B1 dataset, the real success probabilities and those predicted by the best analytical and deep learning models (ESP^i_{sr} and NNESP_{sr}, respectively). The B1 dataset was used due to its higher number of samples, allowing the data-driven NNESP_{sr} model to achieve the best possible accuracy, and due to its higher maximum circuit depth, allowing its circuits to display a wider range of success probabilities. Tab. 4 quantizes the regression quality for the two models.

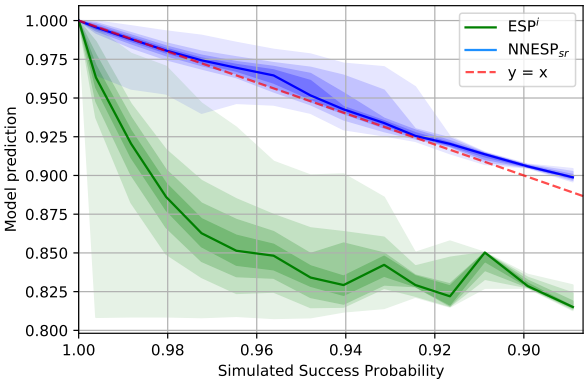


Figure 4: Demonstration of the deep learning models' capability to mimic the real success probability values. We compare the predictions from the ESP_{sr} and ESP^i models in the context of the B1 dataset. The 25%, 50%, 75% and 100% percentile bands are displayed. The minimum displayed real success probability value is ≈ 0.88, due to the dataset's maximum depth of 50 cycles, which are insufficient for the success probability to drop further than this.

It can be see that the NNESP_{sr} model is a considerably better estimator for the success probability, yielding very close values in most of the considered domain. Comparatively, the ESP models do not output a similar value, and tends to further diverge as the real success probability (as computed by the Quantumsim simulator[13]) drops, while presenting a large dispersion of predictions. As can be seen in Tab. 4, the MAE tells us that the ESP_{sr} model's error is on average more than one order of magnitude smaller than the ESP^i's error. Furthermore, the absolute value of the average difference is more than three orders of magnitude higher in the ESP^i case, expressing a much larger bias that its counterpart. The fact that it is negative means that ESP^i tends to underestimate the real success probability value, a fact that is not surprising due

		Kendall τ Correlation				Improvement relative to ESP^0			
		A1	A2	B1	B2	A1	A2	B1	B2
Metric	Circ. size	0.537	0.446	0.541	0.457	0.75	0.78	0.76	0.78
	Circ. depth	0.727	0.595	0.725	0.605	1.02	1.04	1.01	1.04
	N_{2qbg}	0.634	0.313	0.637	0.324	0.88	0.55	0.89	0.55
	TESP1	0.630	0.409	0.471	0.416	0.88	0.71	0.66	0.71
	TESP2	0.519	0.426	0.522	0.433	0.72	0.74	0.73	0.74
	ESP^0	0.716	0.573	0.714	0.584	1.00	1.00	1.00	1.00
	ESP^i	0.719	0.582	0.717	0.591	1.00	1.02	1.00	1.01
	ESP$^0_{sr}$	0.709	0.572	0.707	0.576	0.99	1.00	0.99	0.99
	ESP$^i_{sr}$	0.697	0.522	0.695	0.525	0.97	0.91	0.97	0.90
	NNESP	0.731	0.602	0.731	0.611	1.02	1.05	1.02	1.05
	NNESP$_{sr}$	0.783	0.657	0.817	0.696	1.10	1.13	1.14	1.18

Table 3: Ability of each model to correctly rank each circuit datasets. The Kendall τ correlation between each model and the ground truth is presented for every model-dataset combination (for all models: A1, A2, B1, B2). N_{2qbg} is the identifier for the two-qubit gate number metric. Additionally the normalized correlations relative to the ESP^0 model (Nishio et al.[12]) are also displayed. In the case of the NN-based models, the displayed correlation values were obtained after τ-driven hyperparameter optimization (number of layers of each type and neurons per layer). The boxed values represent the best correlation obtained within each dataset.

Model	MAE	Avg. diff. (10^{-2})	R^2
ESP^i	0.0638	-5.97	-45
NNESP$_{sr}$	0.0025	0.00691	0.85

Table 4: Regression quality of the best analytical (ESP^i) and the best deep learning (ESP$_{sr}$) models, for the B1 dataset. Displayed are the Mean Absolute Error (MAE), the Average Difference, and R^2.

to the simple recurrent gate fidelity multiplication method it relies on, which tends to quickly converge to zero. Finally, the R^2 tells us that how well the models fit the data compared to the constant model yielding the average real success probability for every circuit. The calculated values reinforce the suspicion of ESP$_{sr}$ being a good estimator of the success probability. On the other hand, ESP^i's negative R^2 value reflects the fact that it actually performed worse as a estimator than the constant model. The estimation capacity of the deep learning models didn't seem to hold for circuits longer than 50 cycles, however: the network appears to hit a plateau and attribute a success probability of ≈ 0.88 to such circuits. This is likely due to the fact it was not trained on circuits with success probabilities lower than this.

Additionally, Fig. 4 depicts the fact that the dataset's samples may not be regularly distributed across the presented success probability interval, which might contribute to model bias. We suggest, for future works, researching how to best construct datasets in order to make the most out of data-driven models.

These novel deep learning models are, as far as the author knows, the first capable of reasonably estimating the success probability of a short circuit in a single-shot evaluation, without requiring complex and costly simulations, or running the circuit in a quantum device a high number of times – usually in the order of thousands, to reduce statistical errors.

5. Conclusions

A novel method was proposed in order to evaluate different metrics, independently of a mapper algorithm: one that uses each model to score a set of circuits, and rank them accordingly; afterwards, a ranking correlation method (such as the Kendall τ correlation factor) can be used to calculate the correlation with the true circuit ordering (obtained via simulation). This new method allowed the unbiased comparison of several existing models, as well as the newly developed ones.

New models, both of analytical an deep learning nature were also proposed and evaluated. It was found that, within the analytical models, the ESP-based models were the best performing. Among these ESP-based models, the best results were obtained by considering only the gate-type of each gate (i.e. discarding the information regarding the specific rotation) and considering implicit idling gates. All the deep learning-based models, on the hand, performed consistently better than any of the analytical models, accuracy-wise. In particular, the NNESP$_{sr}$ model, discarding no circuit information, yielded improvements relative to the best analytical model (ESP^i) in the 10%-18% range.

Besides displaying the best accuracy (although at the cost of higher evaluation speed), the proposed deep learning models also exhibited a previously unseen perk: the capacity to accurately estimate the success probability of shallow quantum circuits. This may allow new use cases where such quantities are of interest (without the need to simulate/run the circuits in quantum devices and perform sampling). As an added advantage, the deep

learning model does not need parameters such as gate fidelities and coherence times to be extracted from the device.

Additionally, an important advantage of data-driven models such as those based on deep learning, is the ability to take into to account the specific noise dynamics of different devices, which might otherwise require the development of different analytical models, as well as the ability to capture hard to model noisy behaviours such as *leakage*[21] and *crosstalk*[22].

5.1. Future Work

Deep learning based-models still face a couple of hurdles, namely the qubit scalability problem: the current model requires a growing number of parameters, as the number of qubits and the connections between them grows, and more input nodes become therefore needed. One possible solution may lie in the usage of Graph Neural Networks (GNNs)[23], which take graphs as input (instead of vectors): this can prove beneficial since graphs are the natural way to represent a set of interactions between a number of qubits and other data of combinatorial nature. Additionally, convolutional approaches, such as the ones that constitute the basis of Convolutional Neural Networks (CNNs)[24], may improve learning by sharing the learned impact of the different gates among qubits.

Another scalability-related avenue is scalability with circuit depth, providing the ability for circuits longer than 50 cycles to be used during training and evaluation time. A way to tackle this may lie in the use of newer sequential models, such as Transformers or Reformers[25], which tend to solve similar sequence length-related issues present in RNNs, and allow the fidelity of whole quantum programs to be predicted. Such models may present themselves as new ways to benchmark quantum devices.

It is also of interest to develop a good method to encode arbitrary gates (and not just a finite set). One possible way to solve this might be to encode single qubit gates as a axis-angle pair of parameters: for example, encoding categorically encoding the axis (as one of 3 axis classes), while encoding the angle as a fractional, normalized number.

A third goal that is ultimately necessary to indicate overall feasibility of deep learning models, is to study of the behaviour of the presented deep learning techniques in a real-world setting, i.e. a real quantum device, instead of just relying on simulations. Such would require the addition of measurement operations to the proposed models, which should be straightforward since they can be encoded in a way similar to single qubit gates.

Additionally, the dataset generation methods for the deep learning models can itself be subject of study, since the quality of the dataset may impact the ability of data-driven models to be accurate and extrapolate.

Other avenues of research are also possible within the machine learning realm, both inside and outside the deep learning framework. On one hand, inside the deep learning framework, other Machine-Learned Ranking approaches[20] can be pursued, other than the current Pointwise approach, namely the Pairwise (learning to rank, by learning to perform binary comparisons between circuits) and Listwise (learning to rank by considering the entire set of circuits) approaches[20]. On the other hand, outside the deep learning framework, other noteworthy methods should be mentioned. Examples of such methods are decision tree based algorithms[26] (such as Random Forests and Gradient Boosting) and Genetic Programming[27]. Although it may be argued that these methods have less potential for accuracy that deep learning[28], the increased explainability they offer may make for easier scalability. Genetic programming, in particular, may be especially interesting, since it is a data-driven method that can find analytical expressions that model a specific problem, by combining different expressions using genetics-based processes such as mutations and crossovers.

Tackling the presented scalability issues, while still achieving high accuracy and reasonable training procedures, could yield a model easily portable to different quantum computing technologies, while empowering mapping algorithms with the ability to make better choices. Achieving such improved mapping procedures can improve the reliability NISQ devices, boosting their capability to solve real world problems in near future.

References

[1] D. Gottesman, “An introduction to quantum error correction and fault-tolerant quantum computation,” 2010, pp. 13–58.

[2] G. Li, Y. Ding, and Y. Xie, “Tackling the Qubit Mapping Problem for NISQ-Era Quantum Devices,” in *International Conference on Architectural Support for Programming Languages and Operating Systems - ASPLOS*, 2019, pp. 1001–1014.

[3] M. Kjaergaard, M. E. Schwartz, J. Braumüller, P. Krantz, J. I.-J. Wang, S. Gustavsson, and W. D. Oliver, “Superconducting Qubits: Current State of Play,” Tech. Rep., 2019.

[4] X. Zhang, H. O. Li, G. Cao, M. Xiao, G. C. Guo, and G. P. Guo, “Semiconductor quantum computation,” pp. 32–54, 2019.

[5] D. Greenbaum, “Introduction to Quantum

Gate Set Tomography," 2015. [Online]. Available: http://arxiv.org/abs/1509.02921

[6] P. Murali, A. Javadi-Abhari, F. T. Chong, and M. Martonosi, "Formal constraint-based compilation for noisy intermediate-scale quantum systems," *Microprocessors and Microsystems*, vol. 66, pp. 102–112, 2019.

[7] K. Bertels, C. G. Almudever, L. Lao, I. Ashraf, B. van Wee, N. Khammassi, and J. van Someren, "Mapping of lattice surgery-based quantum circuits on surface code architectures," *Quantum Science and Technology*, vol. 4, no. 1, p. 015005, 2018.

[8] S. G. V. Wee, "Mapping of quantum algorithms on a quantum chip," 2017.

[9] A. Zulehner, A. Paler, and R. Wille, "An Efficient Methodology for Mapping Quantum Circuits to the IBM QX Architectures," dec 2017. [Online]. Available: http://arxiv.org/abs/1712.04722

[10] J. X. Lin, E. R. Anschuetz, and A. W. Harrow, "Using Spectral Graph Theory to Map Qubits onto Connectivity-Limited Devices," 2019. [Online]. Available: http://arxiv.org/abs/1910.11489

[11] D. Venturelli, M. Do, J. Frank, E. Rieffel, K. E. C Booth, T. Nguyen, P. Narayan, and S. Nanda, "Quantum Circuit Compilation: An Emerging Application for Automated Reasoning," Tech. Rep., 2018. [Online]. Available: https://openreview.net/pdf?id=S1eEBO3nFE

[12] S. Nishio, Y. Pan, T. Satoh, H. Amano, and R. Van Meter, "Extracting Success from IBM's 20-Qubit Machines Using Error-Aware Compilation," 2019. [Online]. Available: http://arxiv.org/abs/1903.10963

[13] T. E. O'Brien, B. Tarasinski, and L. DiCarlo, "Density-matrix simulation of small surface codes under current and projected experimental noise," pp. 1–9, 2017. [Online]. Available: http://arxiv.org/abs/1703.04136

[14] A. Van Rynbach, A. Muhammad, A. C. Mehta, J. Hussmann, and J. Kim, "A Quantum Performance Simulator based on fidelity and fault-path counting," p. 13, 2012. [Online]. Available: http://arxiv.org/abs/1212.0845

[15] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, 2016.

[16] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: Lstm cells and network architectures," pp. 1235–1270, 2019.

[17] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A Search Space Odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2017.

[18] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM recurrent networks," *Journal of Machine Learning Research*, vol. 3, no. 1, pp. 115–143, 2003. [Online]. Available: www.idsia.ch

[19] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *32nd International Conference on Machine Learning, ICML 2015*, vol. 1. International Machine Learning Society (IMLS), feb 2015, pp. 448–456. [Online]. Available: https://arxiv.org/abs/1502.03167v3

[20] T. Y. Liu, "Learning to rank for Information Retrieval," *Foundations and Trends in Information Retrieval*, vol. 3, no. 3, pp. 225–231, 2009. [Online]. Available: http://dx.doi.org/10.1561/1500000016

[21] G. Graziano, "Quantum information leakage," *Nature Reviews Chemistry*, vol. 4, no. 4, p. 170, apr 2020. [Online]. Available: https://doi.org/10.1038/s41570-020-0178-z

[22] M. Sarovar, T. Proctor, K. Rudinger, K. Young, E. Nielsen, and R. Blume-Kohout, "Detecting crosstalk errors in quantum information processors," *Quantum*, vol. 4, 2020.

[23] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A Comprehensive Survey on Graph Neural Networks," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21, 2020.

[24] A. Ajit, K. Acharya, and A. Samanta, "A Review of Convolutional Neural Networks," in *International Conference on Emerging Trends in Information Technology and Engineering, ic-ETITE 2020*. Institute of Electrical and Electronics Engineers Inc., feb 2020.

[25] N. Kitaev, Ł. Kaiser, and A. Levskaya, "Reformer: The Efficient Transformer," jan 2020. [Online]. Available: http://arxiv.org/abs/2001.04451

[26] W.-Y. Loh, "Classification and Regression Trees," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, pp. 14–23, 2011.

[27] M. T. Ahvanooey, Q. Li, M. Wu, and S. Wang, "A survey of genetic programming and its applications," *KSII Transactions on Internet and Information Systems*, vol. 13, no. 4, pp. 1765–1794, apr 2019.

[28] T. J. Sejnowski, "The unreasonable effectiveness of deep learning in artificial intelligence," *Proceedings of the National Academy of Sciences*, p. 201907373, jan 2020. [Online]. Available: www.pnas.org/cgi/doi/10.1073/pnas.1907373117