



## Artificial neural network syndrome decoding on IBM quantum processors

Brhyeton Hall <sup>1,\*</sup>, Spiro Gicev,<sup>1,†</sup> and Muhammad Usman <sup>1,2,‡</sup><sup>1</sup>*School of Physics, The University of Melbourne, Parkville, 3010 Victoria, Australia*<sup>2</sup>*Data61, CSIRO, Research Way, Clayton, 3168 Victoria, Australia*

(Received 29 November 2023; accepted 21 June 2024; published 8 July 2024)

Syndrome decoding is an integral but computationally demanding step in the implementation of quantum error correction for fault-tolerant quantum computing. Here, we report the development and benchmarking of Artificial Neural Network (ANN) decoding on IBM quantum processors. We demonstrate that ANNs can efficiently decode syndrome measurement data from heavy-hexagonal code architecture and apply appropriate corrections to facilitate error protection. The current physical error rates of IBM devices are above the code's threshold and restrict the scope of our ANN decoder for logical error rate suppression. However, our work confirms the applicability of ANN decoding methods of syndrome data retrieved from experimental devices and establishes machine learning as a promising pathway for quantum error correction when quantum devices with below threshold error rates become available in the near future.

DOI: [10.1103/PhysRevResearch.6.L032004](https://doi.org/10.1103/PhysRevResearch.6.L032004)

*Introduction.* The development of quantum processors has made remarkable progress over the past few years with quantum devices consisting of more than 100 qubits currently accessible from multiple developers [1–3]. In principle, 100 qubits could allow computations intractable on classical supercomputers, yet the computational capabilities of the current generation of quantum processors are limited by high levels of physical noise [4]. Several studies have implemented and tested error mitigation strategies to suppress the detrimental impact of noise with varying levels of success [5–8]. Ultimately, the full power of quantum computers can only be realized when Quantum Error Correction (QEC) techniques are implemented. These will allow efficient and scalable detection and correction of errors in quantum circuits, leading to fault-tolerant quantum computations [9–12]. Over the recent decades, QEC codes have been theoretically developed to provide a means to suppress errors on logical information through the use of encoding in a larger Hilbert space [12–15]. One of the leading QEC codes is the surface code, which offers a high logical error rate threshold based on nearest neighbor interactions between qubits on a two-dimensional lattice [10,16]. The implementation of surface code-based QEC requires the classical processing of syndrome data—related to the physical error locations—to find appropriate corrections for physical qubits. However, this step, known as decoding, is a computationally intensive task. Recent work has theoretically shown

that Artificial Neural Network (ANN)-based decoders can facilitate fast and scalable decoding [17–24], which is crucial to prevent the accumulation of errors during any quantum computation. The next major milestone is to implement an ANN-based syndrome decoder on quantum processors to directly benchmark their performance. This has been reported by three recent papers to date, which are based on experimental data from devices developed by Google [25–27].

In this work, we develop an ANN-based syndrome decoder and demonstrate its implementation on IBM quantum processors. Further, we assess its performance through comparison against the well-established graph-based Minimum Weight Perfect Matching (MWPM) technique, using PyMatching [28]. Our work shows that, in principle, ANN-based syndrome decoders can efficiently process syndrome measurement data from IBM devices and suggest appropriate corrections—achieving a crucial step in the pipeline of QEC on quantum computational devices.

Historically, the development of surface code literature has been primarily based on the square lattice arrangement of qubits [10,16,29], yet the architecture of IBM quantum processors is built on a heavy-hexagonal (HH) arrangement of qubits, as shown in Fig. 1(a). The motivation for such a qubit layout was to reduce the local connectivity of qubits. This addressed the physical difficulty of controlling many connections to each qubit and aimed to reduce cross-talk noise [30]. However, the HH format required the modification of the traditional square surface code construction to a hexagonal architecture, with ancillary qubits—changing the underlying circuit structures for syndrome measurement. In 2020, Chamberland *et al.* laid out the foundational framework for QEC on HH and heavy-square lattices of low-degree locally connected qubits [30], introducing the HH QEC code. This original HH code was optimized to minimize the number of required physical qubits by removing some ancillary qubits on the boundaries of the hexagonal lattice and maintaining a lattice

\*Contact author: [brhyeton@unimelb.edu.au](mailto:brhyeton@unimelb.edu.au)†Contact author: [gicevs@unimelb.edu.au](mailto:gicevs@unimelb.edu.au)‡Contact author: [musman@unimelb.edu.au](mailto:musman@unimelb.edu.au)

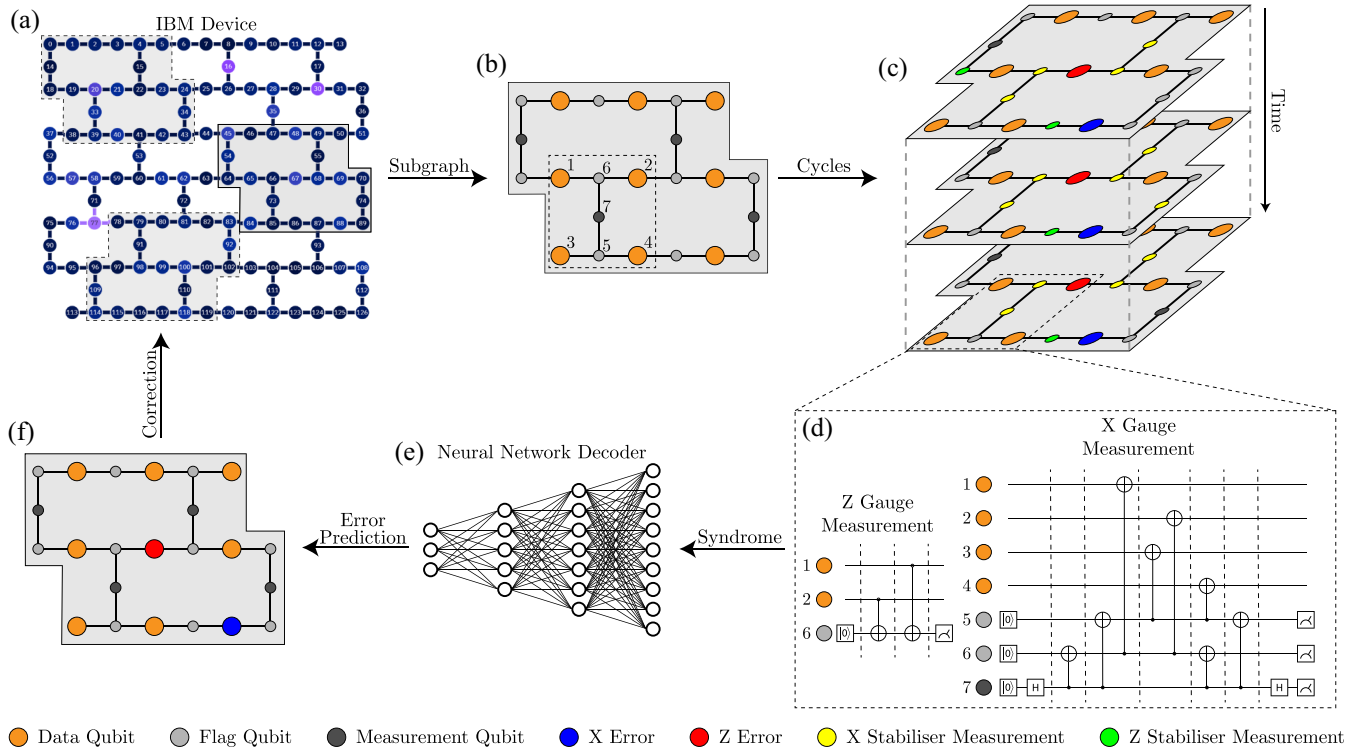


FIG. 1. Neural network decoder framework. (a) The lattice connectivity of qubits of a 127 qubit device developed by IBM with color range denoting error probabilities associated with single and two qubit gates. The shaded section represents a subsection of this device where the average error rate is lowest, in a region which supports a  $d = 3$  HH error correction code. Dotted outlines indicate some other possible subgraph locations. (b) The qubits of a HH code, with orange circles representing the data qubits and light/dark gray circles representing the ancillary flag and measurement qubits, respectively. Connecting lines represent the connectivity of two-qubit gates within the lattice. (c) Multiple cycles of the HH error syndrome measurement in the presence of circuit noise. (d) The circuits for  $X$  and  $Z$  gauge operator measurement of the HH code. (e) An ANN-based syndrome decoder as developed in this work. A large input layer takes the measurements over  $d$  cycles, and it linearly decreases over four layers to an output which is the size of the number of data qubits. (f) A possible correction being sampled from the prediction given by the ANN-based syndrome decoder. The appropriate correction is then applied to the IBM device.

connectivity of, at most, 3 [30]. However, IBM has developed increasingly large devices on HH lattices, without the original optimization of boundaries [31], shown in Fig. 1(a), as the original code layout was incompatible with being realized in the bulk of a HH lattice. This created a discrepancy between the HH code proposed in Ref. [30] and the HH layout of physical qubits in IBM devices. To address this disparity, we have modified the existing HH code, by adjusting the original prescription’s boundaries to fit with the bulk (see the Methods section for details on the adjustment made). This conforms with the IBM quantum processor layout, which is a crucial step in the direct implementation and benchmarking of our ANN decoder on IBM devices. A recent work by Sundaresan *et al.* has also looked at the modified HH code for distance 3 measurements [32]. However, our work is distinct, as we investigate HH code threshold plots and implementation comparison between distance 3 and 5 codes based on direct measurements on IBM devices.

**HH adjustment.** Across the structure of the HH code, qubits are labeled as either data, flag, or measurement qubits. These different qubit types are what facilitate the locating of errors in the HH code. These form the basis of the stabilizer formalism for QEC codes. Although IBM quantum processing devices have been developed for some years, the HH code which directly corresponds to the physical layout has not

been discussed often, with only a few current works directly implementing the adjusted HH structure on superconducting transmon qubits [32,33]. Within the main text, it is stated that the HH boundary optimization was not included when IBM physically realized their quantum devices.

In Supplemental Material Fig. S1 [34], the boundary optimization shown on the left is removed on the right. The structure shown on the right-hand side is physically implementable on IBM devices. Within the adjusted HH lattice on the right of Supplemental Material Fig. S1 [34], there are three types of stabilizer generator: the  $X$ -type Bacon-Shor style operators,

$$S_X = \prod_n X_{n,j} X_{n,j+1},$$

the weight-four  $Z$ -type plaquette operators, found in the bulk,

$$S_Z = Z_{i,j} Z_{i+1,j} Z_{i,j+1} Z_{i+1,j+1},$$

and the weight-two  $Z$ -type edge operators,

$$S_Z = Z_{2m,1} Z_{2m+1,1}, Z_{2m-1,d} Z_{2m,d},$$

where  $i, j \in \mathbb{N} \leq d - 1$ ,  $m \in \mathbb{N} \leq \frac{d-1}{2}$ , and  $n \in \mathbb{N} \leq d$ , and  $i + j = \text{even}$  in the second set. Here,  $i, j$  refer to the lattice of data qubits, with  $i$  as rows and  $j$  as columns. The stabilizer group, as used in QEC codes, is sufficiently defined by the

stabilizer generators which form the entire group after all multiple combinations. Given the boundary conditions of the device, the edge operators are found along the top and bottom of the lattice when arranged in the alignment of Supplemental Material Fig. S1 [34]. This is to ensure that operators do not act on nonexistent qubits. The result of measurement of the stabilizers across the lattice is the syndrome measurement. These generators mutually commute, allowing for their collective simultaneous measurement. Given that there are many ancillary qubits on the lattice, gauge operators are defined to localized areas to measure the local parity, and the stabilizers of each kind measure the parity of gauge operators of each kind. The gauge operators are defined as

$$G_X = X_{i,j}X_{i+1,j}X_{i,j+1}X_{i+1,j+1}, \\ X_{1,2m-1}X_{1,2m}, X_{d,2m}X_{d,2m+1}$$

and

$$G_Z = Z_{i,j}Z_{i+1,j}$$

for  $X$  and  $Z$  gauge operators, respectively, where  $i, j \in \mathbb{N} \leq d$ ,  $m \in \mathbb{N} \leq \frac{d-1}{2}$ . A constraint of  $i + j = \text{odd}$  must be used for the first term in the  $X$  gauge operator set. The measurements of these gauge operators and hence stabilizers can be facilitated by the gauge operator circuit diagrams illustrated in Supplemental Material Fig. S2 [34]. Supplemental Material Fig. S4 [34] illustrates the overall layout of a  $d = 5$  adjusted HH code with some data errors and corresponding stabilizer measurements.

In Supplemental Material Fig. S4 [34], the yellow and green stabilizers are shown for illustrative purposes. Examples of data qubit errors are shown and the corresponding stabilizers are ‘lit up’ from dull to bright, via the measurement of the gauge operators. The eigenvalues associated with the stabilizer operators in Supplemental Material Fig. S4 [34] are

$$[-1 +1 -1 +1 -1 +1 +1 +1 -1 -1 +1 -1] \\ [-1 -1 +1 +1] \quad (1)$$

corresponding to the  $Z$  and  $X$  operator, respectively. These are simplified to

$$[1 0 1 0 1 0 0 0 1 1 0 1] \\ [1 1 0 0] \quad (2)$$

for ease of ANN training. In Eq. (2), a zero is given where no change has occurred and 1 is given when a stabilizer change has occurred:  $-1$  eigenvalue to  $+1$  eigenvalue. When multiple errors occur within the same parity measurement of a single stabilizer, it may have its eigenvalue inverted twice, returning to its original state. Therefore, only stabilizers, at the end of chains have their values changed, as illustrated in Supplemental Material Fig. S4 [34]. This is less obvious for the  $Z$  errors, as the nature of the Bacon-Shor stabilizer allows for chains to be continued anywhere across entire columns.

Errors across the lattice which are of the same form as the stabilizer elements, generators or otherwise, commute with all stabilizer generators and hence do not change the underlying information in the lattice. This means that the encoded state of the lattice may only be affected by a global phase and encoded information is unaltered. Given that the state is unaltered, gates of the same kind can be applied to the lattice wherever

required, to correct for errors. This can be used to create sets of equivalent error chains from the same start and end points on the lattice.

*ANN construction and training.* In the case of square surface code lattices, it has been shown that ANN syndrome decoders can offer highly promising performance when suggesting suitable corrections [17,19,35–38], including testing on experimental data [25]. The low-level decoders developed in Refs. [17,35,36] were built in a similar manner to this work. They each show the ability of an ANN to learn the relationship between syndrome data and corrections after being given multiple training instances. Many ANN varieties have been developed for square surface codes, including dense Feed Forward Neural Networks (FFNN), Long Short Term Memory Networks (LSTM), and Convolutional Neural Networks (CNN). Varsamopoulos *et al.* showed that although slower than the FFNN, the LSTM was more accurate at decoding on average and both were faster and more accurate than the MWPM baseline [19]. Meinerz *et al.* and Gicev *et al.* have independently shown that implementing convolutional layers allows an ANN decoder to be compatible with larger code distances unseen in training [18,20]. These results showed that an ANN syndrome decoder is able to fit any size QEC square surface code.

The ANN developed for this work was built with dense layers, meaning each neuron within each layer is intricately connected with each neuron in the previous layers. The choice for the number of hidden layers is based on the decoder performance. Limited overfitting of training data occurred when two hidden layers were included. Utilizing exclusively dense layers is the simplest layer structure of a neural network and requires no additional pruning or alterations [17]. This methodology allows for the quick proof of concept construction of an ANN syndrome decoder for physical devices and can give suitable corrections with minimal pre/postprocessing. Given that the input layer takes the entirety of the syndrome measurement at once, there is no need to explicitly distinguish between bulk stabilizers and boundary stabilizers when training the network. The network is able to learn the direct relationship between observed syndrome patterns and appropriate corrections without needing to perform auxiliary tasks after corrections are applied—similar to the MWPM algorithm. The MWPM algorithm can provide exact corrections by pairing  $-1$  eigenvalue stabilizers without the need for any pre/postprocessing, but it lacks in the speed of suggestion, especially as the distance of the code increases.

At the smallest distance, 3, the size of input and output layers are the same, yet the input layer size grows significantly faster than the output layer when the distance of the code is increased. Each entry in the input corresponds to a single stabilizer measurement, with the total equaling the number of stabilizers,  $n$  multiplied by the number of cycles,  $d: \frac{d}{2}(d^2 + 2d - 3)$ . Similarly, each output pair corresponds to a single data qubit requiring  $X$  and  $Z$  correction, respectively. The total output size is  $2d^2$ .

Each layer is activated with the ReLU activation function, excluding the final layer which incorporates a Sigmoid function, to return values between 0 and 1. The BinaryCrossEntropy loss function and ADAM optimizer functions were used, allowing the network output to be interpreted as

a probability that an error was present at each qubit. During training and testing, 96 Intel Xeon Platinum 8274 CPU cores were used and four NVIDIA V100-SXM2-32GB GPUs were used. Each value in the output of the final layer will be a value between 0 and 1, which are then processed in two ways. First, the values are truncated such that each value in the correction suggestion is exactly 0 or 1, which corresponds to a given correction being not required or required, respectively. If this correction is consistent with the final syndrome measurement cycle, the truncated prediction is kept. If not, the predictions given are sampled using a Bernoulli Trial, and this is repeated until an appropriate correction is given [36]. Sampling of a prediction could take many re-tries if the network is uncertain with its prediction. Therefore a cut-off point is used, where after  $n$  re-samples, if no appropriate correction is given, re-sampling is stopped and it is assumed that a logical error has occurred in that instance [36]. Although there is theoretically a 50% chance that a logical error has occurred, for benchmarking purposes, the occurrence of a logical error is assumed and the additional logical errors are reflected in Fig. 3. Re-sampling can be a major overhead computationally, furthering the need to cut off early, before qubits decohere within the structure. Given that this work only considered small distances of the HH QEC code, truncation of predictions often produced an appropriate correction—not always requiring re-sampling. The coherence time of current qubits is on the order of microseconds and this work’s re-sample time is also on the order of microseconds, forcing re-sampling to be avoided as much as possible [39]. This dense ANN methodology is fast enough to produce corrections within the coherence time of physical qubits in the lattice for these small distances [39]. The average decode time per instance for MWPM is approximately 1 ms compared to 0.3 ms for the ANN in this work. The time taken to find corrections increases with code distance and may not be appropriate for large distance codes. Instead, CNN techniques can be employed for decoding large distance codes [18,20,40,41].

**Results and discussion.** Figure 1(b) schematically illustrates a distance 3 patch of the adjusted HH code shape as described within our work, where data qubits (orange) store useful information and ancilla qubits (gray) are used to facilitate syndrome measurements. These measurements are used to locate errors on physical qubits within the HH lattice. Typically, the syndrome measurements are collected in multiple rounds before they are decoded to find appropriate corrections for data qubit errors and also in the syndrome measurement process itself. Figure 1(c) schematically shows many cycles of the HH code being executed and corresponding syndromes measured for each cycle. The circuits that are used to measure the syndrome in both the  $X$  and  $Z$  basis are shown in Fig. 1(d), with the physical qubits numbered in Fig. 1(b) illustrated within a dashed box.

The data collected from syndrome measurement over several cycles is processed by a classical syndrome decoding method. This prescribes adequate corrections to fix physical errors in data qubits and restore the logical state of the lattice. The construction of an efficient and scalable syndrome decoder is a challenging computational problem and has recently been the focus of intensive research [42,43]. One of the leading syndrome decoder algorithms, MWPM, calculates

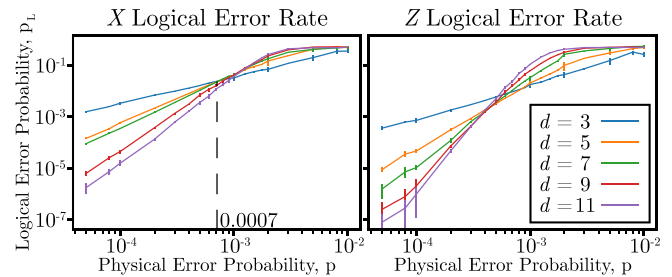


FIG. 2. Benchmarking of the adjusted HH code with MWPM. Both the threshold and pseudothreshold for  $X$  logical errors (left) and  $Z$  logical errors (right) for the adjusted HH code are shown, decoded by MWPM as implemented by PyMatching. The vertical dash line indicates the crossover point for  $d = 3$  and  $5$  curves.

corrections by matching pairs of changed stabilizers. It has received extensive development in many square lattice surface code studies [16,28,44–47]. Chamberland *et al.* implemented the MWPM algorithm to the original HH layout to compute logical error rate curves for both  $X$  and  $Z$  logical errors [30].

We benchmarked the adjusted HH code using the MWPM decoder from the Python package PyMatching [28] and compared it to the work of Chamberland *et al.* [30]. In Fig. 2, odd distances,  $d$ , of the code between 3 and 11 are tested and the lowest clear crossover point can be seen at approximately 0.0007 in the  $X$  logical error plot on the left. This will be the benchmark for thresholds for the adjusted HH QEC code, as only distances 3 and 5 are tested by demonstration on IBM devices. Note that the threshold if computed based on increasing code distance would be slightly higher ( $\sim 0.001$ ). We used MWPM as implemented by PyMatching to confirm the  $X$  logical error threshold of 0.0045 of Chamberland *et al.* [30] and a very similar threshold of 0.005 was found. Details of this can be found in Supplemental Material Sec. S1 [34]. The addition of  $2d - 2$  extra ancilla qubits and  $2d - 2$  of CNOTs has lowered the threshold physical error probability further by a small amount.

Despite promising performance, it has been regularly discussed that the MWPM algorithm may not be fast enough for quantum state coherence times on current devices [29,35,48,49]. Even the best adaptations of this algorithm are slow in the large distance regime of QEC codes. The development of fast and scalable syndrome decoders have been a topic of significant research, with proposals attempting to address the real-time decoding challenge [43]. Machine Learning (ML)-based syndrome decoder construction has gained significant momentum in recent years, with some studies indicating that a faster and scalable syndrome decoding method may be possible by leveraging the computational efficiency and flexibility of ANN algorithms. In terms of the HH architecture, there is no current demonstration on the measured syndrome data from IBM devices. Some theoretical studies has explored the implementation of dense ANN- and CNN-based decoders for the original HH code proposed by Chamberland *et al.*, yet their work is not directly applicable to the IBM hardware due to the aforementioned adjustment required to do so [50,51]. Our work is the first to implement and benchmark an ANN decoder on the adjusted HH code through theoretical simulations and demonstration on IBM devices based on cloud access.

The ANN decoder developed in this work was constructed using the Python package TensorFlow [52]. The decoder consists of an input layer, two hidden layers, and an output layer. More details about the construction of the network are provided in Methods section. In Fig. 1(e), a dense ANN is illustrated, where the number of neurons in each layer linearly decreases from the input layer to the output layer. The size of the input layer is adjusted to feed in each  $X$  and  $Z$  stabilizer measurement separately, for each measurement cycle. The size of the output layer allows a value for both  $X$  and  $Z$  errors for each physical qubit.

Our ANN decoder was rigorously trained on tens of millions of simulated noise patterns, using uniform depolarizing Pauli channels. The uniform depolarizing noise model was simulated with an even chance,  $\frac{p}{3}$ , to select from the three Pauli gate errors  $X$ ,  $Y$ , and  $Z$ . Each qubit can experience each of these errors, and each CNOT on the lattice can experience some tensor product of two Pauli gate errors and the identity, excluding  $I \otimes I$ . No bias or other error factors were included in this training. During training, circuits were modeled such that when Pauli errors occur on a state,  $|\psi\rangle$ , it may be denoted as  $E|\psi\rangle$ , where  $E$  is the combination of errors on a single qubit. The goal of error correction is to detect and apply the appropriate correction to  $|\psi\rangle$  to turn the string of errors  $E$  into the identity,  $I$ , or to return the lattice to an equivalent logical state. We compute a correction  $E_c$  such that the correction succeeds if  $E_c E \in G$ , where  $G$  is the corresponding gauge group. This is simulated within this work by tracking each error which occurs on every qubit and multiplying the Pauli gate errors, where two of the same give the identity;  $X^2 = Y^2 = Z^2 = I$ , and  $XZ = ZX = Y$  up to a global phase.

The ANN decoder developed in our work provides appropriate corrections based on the syndrome measurements over  $d$  cycles of the adjusted HH code. The ANN is able to functionally learn how stabilizer inversions are related to error chains within the lattice, including on the boundaries of the lattice where chains abruptly end. This work has explicitly shown that a dense ANN syndrome decoder can input an exact stabilizer syndrome measurement and return a prediction related to an appropriate correction on par or better than suggestions from the MWPM algorithm.

First evaluation of the model was done with a similar error model to the training: a uniform depolarizing noise model. The underlying physical error,  $p$ , was varied to test the performance at different rates. Further, the decoders were then tested on imported device error models from IBM quantum experience; each physical error rate was given for qubits and two-qubit gates which was then used as the underlying error probability  $p$ . This was implemented for each individual qubit and CNOT, instead of uniformly across the lattice. Finally, the circuits as defined in Fig. 1(d) were constructed to fit distance 3 and 5 HH QEC codes and executed on multiple IBM devices. Figure 3 displays IBM demonstration and theoretical results from our ANN syndrome decoder. The decoder is tested on a simulated lattice of qubits in the form of IBM devices which suffer from uniform circuit-based depolarizing noise (blue and orange line plots) and also on device noise models derived from error rates provided by five of the IBM quantum processors (marked with open circle points). In these

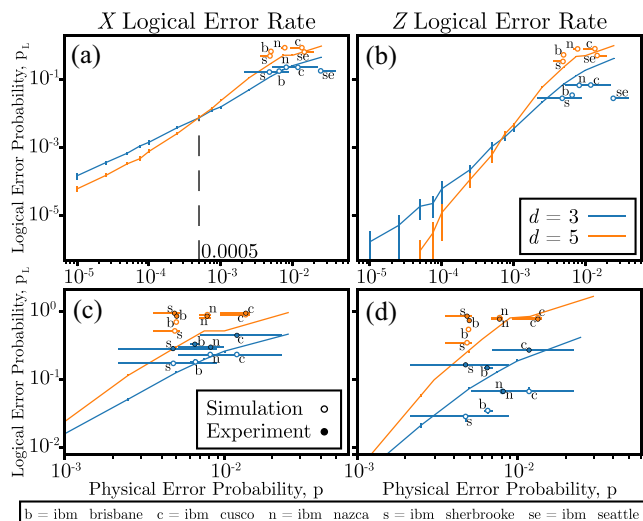


FIG. 3. Neural network decoder implementation on adjusted HH code. Threshold plot for the adjusted HH code decoded by an ANN showing error rates of the  $X$  logical operator (a) and  $Z$  logical operator (b). Each point refers to an error model derived for each IBM device. The horizontal value of the points shown are the overall error rate of the specific subgraph location chosen, and the horizontal uncertainty shows the range of overall error rates of each possible subgraph location on each device, with the point placed on the median heuristic subgraph score. (c), (d) The HH QEC code IBM measurement circuit plots., in which the top right-hand corner of (a) and (b) is enlarged, and the points which refer to the circuits running on the IBM devices are also marked. Unfilled circles refer to the simulated noise model corrections, and filled circles refer to the transpiled circuits run on devices.

plots, similar crossover behavior is observed, and thus it can be inferred that the ANN syndrome decoder is able to decode the HH QEC code with the same overall properties as the MWPM algorithm. Note that the threshold for the ANN syndrome decoder is approximately 0.0005 for  $X$  logical errors, and hence reduced by a small amount compared to the MWPM threshold of 0.0007 from Fig. 2(a). In the future, more sophisticated ML-based syndrome decoders, such as CNN decoders, can be designed to improve the threshold and scale to larger distances [20,38,40,51].

In Figs. 3(a) and 3(b), the blue circle markings correspond to distance 3 subgraphs and orange for distance 5. Each data point has an alphabetical label showing the name of IBM device:  $b = \text{ibm\_brisbane}$ ,  $c = \text{ibm\_cusco}$ ,  $n = \text{ibm\_nazca}$ ,  $s = \text{ibm\_sherbrooke}$ , and  $se = \text{ibm\_seattle}$ . The horizontal uncertainty for each marking corresponds to the possible values of average physical error for each available subgraph location, chosen with a heuristic described in Supplemental Material Sec. S2 [34], with the marking corresponding to the median location. Interestingly, the markings are in the approximate region of the simulated noise curves. This suggests that the ANN syndrome decoder is likely to be able to decode actual noise approximately as well as simulated noise. Due to the preceding threshold error rates of current physical machines, distances above 5 were not tested, since this would only increase the logical error rate and may not provide additional insight.

Figures 3(c) and 3(d) plot results based on direct measurements from the IBM quantum processors. The plots show both device noise simulations (open circles) from Figs. 3(a) and 3(b), as well as IBM demonstration points (colored circles) for a direct comparison. For the data points based on IBM measurements, the adjusted HH QEC code syndrome measurement circuits were created and run on physically realized IBM devices. Each circuit was initialized twice, once for  $X$  measurements and once for  $Z$  measurements, and 10,000 shots were run for each case. The number of logical errors which occurred after the pass through of the ANN syndrome decoder was lower on average than the simulated noise models of the same devices for distance 3, and roughly similar for distance 5. Given that the points are still all within the same area or lower, it would follow that if the devices error rates were below the threshold of approximately 0.0005, then increasing the distance of the code, and using a suitable ANN syndrome decoder, would facilitate fault-tolerant quantum computation [45].

Note that in Fig. 3(b), the device derived error models seem to consistently provide lower logical error rates than equivalent uniform error models. This suggests that there is some intricate phenomenon occurring which may be related to subgraph location choice. Compared to what is expected under the uniform noise model, this results in the reduction of the rate of  $Z$  logical errors, which corrupt  $X$  logical operator values. This is not observed in Fig. 3(d), however, as the measured data from IBM devices is not lower than the simulated uniform noise curve on average. Crosstalk and relaxation errors are missing in the simulated noise model but are possibly present on the physically realized devices, perhaps leading to this variation between IBM devices and simulation [33,53].

*Conclusions.* Despite the expeditious advances in quantum hardware, fault-tolerant quantum computation still requires significant research in the coming years to achieve scalable practical applications to real-world problems. However, this work, to the best of our knowledge for the first time, showed

that the adjusted HH code that matches the IBM quantum machine structure is able to be decoded by both the MWPM algorithm and an ANN syndrome decoder. A dense ANN was shown to be compatible with the adjusted HH code and to perform measurements in accordance to the error rates present on the devices. The IBM demonstration results in this work showed that stabilizer circuit decoding approximately followed the theoretical curve's trend. It is therefore likely that lowering the physical error rate below the threshold will allow for arbitrary suppression of logical errors with code distance increase. This work's dense-style ANN lays the foundation of ANN decoding on physically realized IBM quantum machines.

In the future, our work could be extended with the benchmarking of larger distance code implementations on IBM devices to demonstrate the expected drop in logical error rates with respect to code distance. However, this would require larger physical devices and devices with error rates below the code threshold. A second line of study could be to implement and test more sophisticated ML-based decoders—such as CNN syndrome decoders—on quantum devices. In summary, our work has opened new avenues for experimentally realized, ML-based syndrome decoder implementation on quantum processors. This will be instrumental in realizing fault-tolerant quantum computing in the near future, where larger size and lower error rate devices are anticipated to be available.

*Acknowledgments.* This research was supported by the University of Melbourne through the establishment of the IBM Quantum Network Hub at the university. Computational resources were provided by the National Computing Infrastructure (NCI) and the Pawsey Supercomputing Research Center through the National Computational Merit Allocation Scheme (NCMAS).

M.U. planned and supervised the project. B.H. developed the ANN decoders with input from S.G. B.H. carried out all simulations, benchmarking, and experimental implementation. All authors contributed to the analysis of data. B.H. and M.U. wrote the manuscript with contributions from S.G.

- 
- [1] H. Collins and C. Nay, IBM Unveils 400 Qubit-Plus Quantum Processor and Next-Generation IBM Quantum System Two, <https://newsroom.ibm.com/2022-11-09-IBM-Unveils-400-Qubit-Plus-Quantum-Processor-and-Next-Generation-IBM-Quantum-System-Two> (2022).
  - [2] L. S. Madsen *et al.*, Quantum computational advantage with a programmable photonic processor, *Nature (London)* **606**, 75 (2022).
  - [3] K. Barnes *et al.*, Assembly and coherent control of a register of nuclear spin qubits, *Nat. Commun.* **13**, 2779 (2022).
  - [4] M. E. Beverland, P. Murali, M. Troyer, K. M. Svore, T. Hoefler, V. Kliuchnikov, G. H. Low, M. Soeken, A. Sundaram, and A. Vaschillo, Assessing requirements to scale to practical quantum advantage, [arXiv:2211.07629](https://arxiv.org/abs/2211.07629).
  - [5] Y. Kim *et al.*, Evidence for the utility of quantum computing before fault tolerance, *Nature (London)* **618**, 500 (2023).
  - [6] S. Endo, S. C. Benjamin, and Y. Li, Practical quantum error mitigation for near-future applications, *Phys. Rev. X* **8**, 031027 (2018).
  - [7] A. Strikis, D. Qin, Y. Chen, S. C. Benjamin, and Y. Li, Learning-based quantum error mitigation, *PRX Quantum* **2**, 040330 (2021).
  - [8] S. Bravyi, S. Sheldon, A. Kandala, D. C. McKay, and J. M. Gambetta, Mitigating measurement errors in multiqubit experiments, *Phys. Rev. A* **103**, 042605 (2021).
  - [9] P. W. Shor, Scheme for reducing decoherence in quantum computer memory, *Phys. Rev. A* **52**, R2493 (1995).
  - [10] A. Y. Kitaev, Fault-tolerant quantum computation by anyons, *Ann. Phys.* **303**, 2 (2003).
  - [11] D. Aharonov and M. Ben-Or, Fault-tolerant quantum computation with constant error rate, *SIAM J. Comput.* **38**, 1207 (2008).
  - [12] E. Knill, R. Laflamme, and W. H. Zurek, Resilient quantum computation: Error models and thresholds, *Proc. R. Soc. A: Math. Phys. Eng. Sci.* **454**, 365 (1998).
  - [13] D. Gottesman, Stabilizer codes and quantum error correction, [arXiv:quant-ph/9705052](https://arxiv.org/abs/quant-ph/9705052).
  - [14] A. M. Steane, Active stabilization, quantum computation, and quantum state synthesis. *Phys. Rev. Lett.* **78**, 2252 (1997).

- [15] A. Y. Kitaev, Quantum error correction with imperfect gates, in *Quantum Communication, Computing, and Measurement*, edited by O. Hirota, A. S. Holevo, and C. M. Caves (Springer, Boston, MA, USA, 1997), pp. 181–188.
- [16] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, Surface codes: Towards practical large-scale quantum computation, *Phys. Rev. A* **86**, 032324 (2012).
- [17] S. Varsamopoulos, B. Criger, and K. Bertels, Decoding small surface codes with feedforward neural networks, *Quantum Sci. Technol.* **3**, 015004 (2017).
- [18] K. Meinerz, C.-Y. Park, and S. Trebst, Scalable neural decoder for topological surface codes. *Phys. Rev. Lett.* **128**, 080505 (2022).
- [19] S. Varsamopoulos, K. Bertels, and C. G. Almudever, Comparing neural network based decoders for the surface code, *IEEE Trans. Comput.* **69**, 300 (2020).
- [20] S. Gicev, L. C. L. Hollenberg, and M. Usman, A scalable and fast artificial neural network syndrome decoder for surface codes, *Quantum* **7**, 1058 (2023).
- [21] R. W. J. Overwater, M. Babaie, and F. Sebastiano, Neural-network decoders for quantum error correction using surface codes: A space exploration of the hardware cost-performance tradeoffs, *IEEE Trans. Quantum Eng.* **3**, 1 (2022).
- [22] T. Wagner, H. Kampermann, and D. Bruß, Symmetries for a high-level neural decoder on the toric code, *Phys. Rev. A* **102**, 042411 (2020).
- [23] X. Ni, Neural network decoders for large-distance 2D toric codes, *Quantum* **4**, 310 (2020).
- [24] M. Zhang, X. Ren, G. Xi, Z. Zhang, Q. Yu, F. Liu, H. Zhang, S. Zhang, and Y.-C. Zheng, A scalable, fast and programmable neural decoder for fault-tolerant quantum computation using surface codes, [arXiv:2305.15767](https://arxiv.org/abs/2305.15767).
- [25] J. Bausch *et al.*, Learning to decode the surface code with a recurrent, transformer-based neural network, [arXiv:2310.05900](https://arxiv.org/abs/2310.05900).
- [26] M. Lange, P. Havstrom, B. Srivastava, V. Bergentall, K. Hammar, O. Heuts, E. van Nieuwenburg, and M. Granath, Data-driven decoding of quantum error correcting codes using graph neural networks, [arXiv:2307.01241](https://arxiv.org/abs/2307.01241).
- [27] B. M. Varbanov, M. Serra-Peralta, D. Byfield, and B. M. Terhal, Neural network decoder for near-term surface-code experiments, [arXiv:2307.03280](https://arxiv.org/abs/2307.03280).
- [28] O. Higgott, PyMatching: A Python package for decoding quantum codes with minimum-weight perfect matching, *ACM Trans. Quantum Comput.* **3**, 1 (2022).
- [29] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, Topological quantum memory, *J. Math. Phys.* **43**, 4452 (2002).
- [30] C. Chamberland, G. Zhu, T. J. Yoder, J. B. Hertzberg, and A. W. Cross, Topological and subsystem codes on low-degree graphs with flag qubits, *Phys. Rev. X* **10**, 011022 (2020).
- [31] P. Nation, H. Paik, A. Cross, and Z. Nazario, The IBM Quantum Heavy Hex Lattice, <https://research.ibm.com/blog/heavy-hex-lattice> (2021).
- [32] N. Sundaresan, T. J. Yoder, Y. Kim, M. Li, E. H. Chen, G. Harper, T. Thorbeck, A. W. Cross, A. D. Corcoles, and M. Takita, Demonstrating multi-round subsystem quantum error correction using matching and maximum likelihood decoders, *Nat. Commun.* **14**, 2852 (2023).
- [33] E. H. Chen, T. J. Yoder, Y. Kim, N. Sundaresan, S. Srinivasan, M. Li, A. D. Corcoles, A. W. Cross, and M. Takita, Calibrated decoders for experimental quantum error correction, *Phys. Rev. Lett.* **128**, 110504 (2022).
- [34] See Supplemental Material at <http://link.aps.org/supplemental/10.1103/PhysRevResearch.6.L032004> for additional details on methods.
- [35] G. Torlai and R. G. Melko, Neural decoder for topological codes, *Phys. Rev. Lett.* **119**, 030501 (2017).
- [36] S. Krastanov and L. Jiang, Deep neural network probabilistic decoder for stabilizer codes, *Sci. Rep.* **7**, 11003 (2017).
- [37] P. Baireuther, T. E. O’Brien, B. Tarasinski, and C. W. J. Beenakker, Machine-learning-assisted correction of correlated qubit errors in a topological code, *Quantum* **2**, 48 (2018).
- [38] A. Davaasuren, Y. Suzuki, K. Fujii, and M. Koashi, General framework for constructing fast and near-optimal machine-learning-based decoder of the topological stabilizer codes, *Phys. Rev. Res.* **2**, 033399 (2020).
- [39] IBM, IBM Quantum, <https://quantum-computing.ibm.com/> (2022).
- [40] C. Chamberland, L. Goncalves, P. Sivarajah, E. Peterson, and S. Grimberg, Techniques for combining fast local decoders with global decoders under circuit-level noise, *Quantum Sci. Technol.* **8**, 045011 (2023).
- [41] Y. Ueno, M. Kondo, M. Tanaka, Y. Suzuki, and Y. Tabuchi, NEO-QEC: Neural network enhanced online superconducting decoder for surface codes, [arXiv:2208.05758](https://arxiv.org/abs/2208.05758).
- [42] L. Skoric, D. E. Browne, K. M. Barnes, N. I. Gillespie, and E. T. Campbell, Parallel window decoding enables scalable fault tolerant quantum computation, *Nat. Commun.* **14**, 7040 (2023).
- [43] F. Battistel, C. Chamberland, K. Johar, R. W. J. Overwater, F. Sebastiano, L. Skoric, Y. Ueno, and M. Usman, Real-time decoding for fault-tolerant quantum computing: Progress, challenges and outlook, *Nano Futures* **7**, 032003 (2023).
- [44] A. G. Fowler, Minimum weight perfect matching of fault-tolerant topological quantum error correction in average  $O(1)$  parallel time, *Quantum Inf. Comput.* **15**, 145 (2015).
- [45] R. Acharya *et al.*, Suppressing quantum errors by scaling a surface code logical qubit, *Nature (London)* **614**, 676 (2023).
- [46] A. G. Fowler, A. C. Whiteside, and L. C. L. Hollenberg, Towards practical classical processing for the surface code, *Phys. Rev. Lett.* **108**, 180501 (2012).
- [47] D. S. Wang, A. G. Fowler, A. M. Stephens, and L. C. L. Hollenberg, Threshold error rates for the toric and planar codes, *Quantum Inf. Comput.* **10**, 456 (2010).
- [48] N. Delfosse, A. Paz, A. Vaschillo, and K. M. Svore, How to choose a decoder for a fault-tolerant quantum computer? The speed vs accuracy trade-off, [arXiv:2310.15313](https://arxiv.org/abs/2310.15313).
- [49] C. Chamberland and P. Ronagh, Deep neural decoders for near term fault-tolerant experiments, *Quantum Sci. Technol.* **3**, 044002 (2018).
- [50] D. Bhoumik, R. Majumdar, D. Madan, D. Vinayagamurthy, S. Raghunathan, and S. Sur-Kolay, Efficient machine-learning-based decoder for heavy hexagonal QECC, [arXiv:2210.08730](https://arxiv.org/abs/2210.08730).
- [51] A. Li, F. Li, Q. Gan, and H. Ma, Convolutional-neural-network-based hexagonal quantum error correction decoder, *Appl. Sci.* **13**, 9689 (2023).
- [52] M. Abadi *et al.*, TensorFlow: A system for large-scale machine learning, in *Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation*, OSDI’16 (ACM, New York, 2016), pp. 265–283.
- [53] Y. Tomita and K. M. Svore, Low-distance surface codes under realistic quantum noise, *Phys. Rev. A* **90**, 062320 (2014).