DeepQ Decoding for Fault Tolerant Quantum Computation

Ryan Sweke,[1] Markus S. Kesselring,[1] Evert P.L. van Nieuwenburg,[2] and Jens Eisert[1]

[1] *Dahlem Center for Complex Quantum Systems, Freie Universität Berlin, 14195 Berlin, Germany*
[2] *Institute for Quantum Information and Matter, Caltech, Pasadena, CA 91125, USA*

(Dated: September 28, 2018)

Topological error correcting codes, and particularly the surface code, currently provide the most feasible roadmap towards large-scale fault tolerant quantum computation. As such, obtaining fast and flexible decoding algorithms for these codes, within the experimentally relevant context of faulty syndrome measurements, is of critical importance. In this work we show that the problem of decoding such codes, in the full fault tolerant setting, can be naturally reformulated as a process of repeated interactions between a decoding agent and a code environment, to which the machinery of reinforcement learning can be applied to obtain decoding agents. As a demonstration, by using deepQ learning, we obtain fast decoding agents for the surface code, for a variety of noise-models, within the fully fault tolerant setting.

I. INTRODUCTION

In order to implement large scale quantum algorithms it is necessary to be able to store and manipulate quantum information in a manner that is robust with respect to the unavoidable errors introduced through the interaction of physical qubits with a noisy environment. A typical strategy for achieving such robustness is to encode a single logical qubit into the state of many physical qubits, via a quantum error correcting code, from which it may be possible to actively diagnose and correct errors that might occur [1, 2]. While many quantum error correcting codes exist, topological quantum codes [1–8] in which only local operations are required to diagnose and correct errors, are of particular interest as a result of their experimental feasibility [9–15]. Recently the surface code has emerged as an espescially promising code for large scale fault tolerant quantum computation, due to the combination of its comparitively low overhead and locality requirements, coupled with the availability of convenient strategies for the implementation of all required logical gates.

Within any such code based strategy for fault tolerant quantum compuation, decoding algorithms play a critical role. At a high level, throughout the course of a computation these algorithms take as input the outcome of diagnostic syndrome measurements and should provide as output suggested corrections for any errors which might have occurred, which can then be tracked through the computation and later used to apply corrections to any obtained results. It is particularly important to note that in any physically realistic setting the required syndrome measurements are themselves obtained via small quantum circuits, and are therefore also generically faulty. As such, while the setting of perfect syndrome measurements provides a good test-bed for the development of decoding algorithms, any decoding algorithm which aims to be experimentally feasible should also be capable of dealing with such faulty syndrome measurements. Additionally, such algorithms should also be capable of dealing with experimentally relevant noise models, as well as be fast enought to not present a bottleneck to the execution of computations, even as the size of the code scales to larger code distances.

Due to the importance of decoding algorithms for fault tolerant quantum computation, many different approaches have been developed, each of which tries to satisfy as many of the experimentally required criterion as possible. Perhaps most prominent are algorithms based on minimum-weight perfect matching subroutines [16], however alternative approaches based on techniques such as the renormalization group [17] and locally operating cellular automata [18] have also been put forward. Furthermore, recently techniques from classical machine learning have begun to find application in diverse areas of quantum physics - such as in the efficient representation of many-body quantum states, the identification of phase transitions, and the autonomous design of novel experimental setups MK: TODO: add citations - and various neural-network based decoders have also been proposed [19–25]. However, despite the diversity of decoding algorithms now available, there is not as of yet an algorithm which clearly satisifes all the required criteria, or a clear consensus as to which technique would be the most experimentally feasible in any given experimental context. In particular, while the so-far proposed neural network decoders promise extremely fast decoding times, flexibility with respect to the underlying noise model and the potential to scale to large code distances, all such decoders are so far restricted either to the setting of perfect syndrome measurements, or to the setting in which one is trying to store a logical qubit as long as possible, without the requirement of performing a subsequent logical gate. As such, while this approach seems promising, there remains room for improvement and generalization.

Simultaneously, the last few years have also seen extremely impressive advances in the development of deep reinforcement learning algorithms, which have allowed for the training of neural network based agents capable of obtaining super-human performance in domains such as Atari [26], Chess [27] and Go [28]. These techniques are particularly powerful in situations where it is necessary to learn strategies for complex sequential decision making, involving consideration of the future effects of ones actions. At a surface level, the problem of decoding within the context of fault-tolerant quantum computation seems like exactly such a setting and as such it natural to ask to what extent reinforcement learning techniques could be used to obtain decoding agents, and what advantages such agents might have over existing approaches. In this work we set out to provide answers to these questions.

In particular, we reformulate the problem of decoding within the setting of fault-tolerant quantum computation as a process of sequential competitive interaction between a decoding agent and a code environment. This reframing provides a conceptual framework which allows for the application of various deep reinforcement learning algorithms to obtain neural network based decoding agents. As a proof-of-principle, we then utilize to deepQ learning to obtain fast surface code decoders, for a variety of noise models, within the fully fault-tolerant setting. These results then provide a foundation for extension via both more sophisticated reinforcement learning techniques and more sophisticated neural network models.

In this work we begin by providing an introductory overview of the surface code in Section II, before presenting a description of the decoding problem for fault-tolerant quantum computation in Section III. After a brief introduction to the formalism of reinforcement learning and Q-functions in Section IV we are then able to provide the conceptual reframing of decoding as a reinforcement learning problem in Section V, which is one of the primary results of this work. In Section VI we then present deepQ surface code decoders for a variety of noise models, before finally in Section VII discussing both the advantages and disadvantages of the approach presented here, along with various potential strategies for building upon the results presented in this work.

II. THE SURFACE CODE

We begin by providing a brief description of the surface code. The framework and methods presented in this work are not restricted to the surface code, and could be applied to any stabilizer code, but we choose to restrict ourselves to this code here for both simplicity of presentation and experimental relevance. We will focus on presenting the elements of the surface code necessary for understanding the decoding problem, in the process omitting many details, for a more detailed treatment refer to refs *a,b, and c.*

As illustrated in Fig. 1, we will consider $d \times d$ lattices with a *physical data qubit* on each vertex v, such that the collective state of all qubits on the lattice is an element of the Hilbert space $\mathcal{H} = \mathbb{C}^{2^{(d\times d)}}$. We associate stabilizer operators with each coloured plaquette of the lattice. Stabilizers on blue (orange) plaquettes are operators which apply Pauli X (Z) flips to all qubits on the vertices of the plaquette. Specifically, denoting the set of all blue (orange) plaquettes as B_p (O_p) we define the stabilizer S_p on plaquette p as,

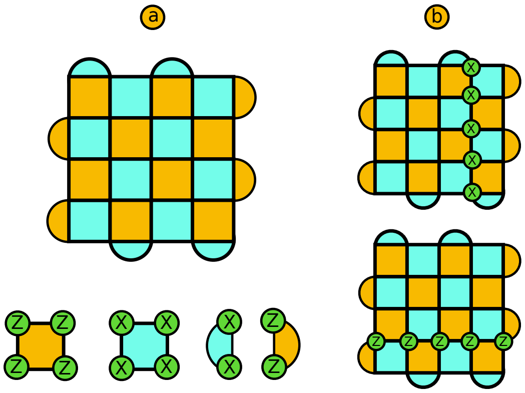

FIG. 1. An overview of the 5×5 surface code. (a) We consider square lattices, with a data qubit on each vertex of the lattice. The coloured plaquettes indicate stabilizer operators as defined in Eq. (1). (b) Logical X_L and Z_L operators for the surface code are given by continuous strings of single qubit X or Z operators connecting the top and bottom or left and right boundaries of the code respectively.

$$S_p = \bigotimes_{v\in p} \sigma_v \quad \text{where} \begin{cases} \sigma_v = X_v & \text{if } p \in B_p, \\ \sigma_v = Z_v & \text{if } p \in O_p. \end{cases} \tag{1}$$

All stabilizers are mutually commuting and have eigenvalues ±1. If we define the Hamiltonian $H = -\sum_p S_p$, then the surface code $\mathcal{H}_{\rm sc} \subset \mathcal{H}$ is the ground state space of this Hamiltonian, which is the space consisting only of simultaneous $+1$ eigenstates of all the stabilizer operators. This subspace is two dimensional, i.e. $\mathcal{H}_{\rm sc} \simeq \mathbb{C}^2$, and as a result we can encode a single *logical qubit* into this code space. Logical operators are then operators which preserve the code space and manipulate the state of the logical qubit. As shown in Fig. 1, logical X (Z) operators, denoted X_L (Z_L), are continuous strings of single vertex X (Z) operators which connect the top and bottom (left and right) boundary of the lattice.

To illustrate the motivation behind such an encoding, let's examine the consequences of a single qubit Pauli flip on a physical data qubit. If we assume that the initial state $|\psi\rangle \in \mathcal{H}_{\rm sc}$ is an element of the code space, then the subsequent state $|\psi'\rangle$ will no longer be an element of the code space. In particular $|\psi'\rangle$ will be an eigenstate with eigenvalue -1 of at least one stabilizer. We say that $|\psi'\rangle$ violates these stabilizers, as illustrated by red circles in Fig. 2 (a). The *syndrome* of a state is the string encoding the outcomes of a simultaneous measurement of all the stabilizers, each of which takes the value ±1. As such, unlike a truly single qubit encoding of a quantum state, if a Pauli flip error occurs on a single physical data qubit, we may be able to identify and correct this error, in the process conserving the logical qubit state, by examining the syndrome of the post-error state. This process of *decoding* is discussed in the next section.

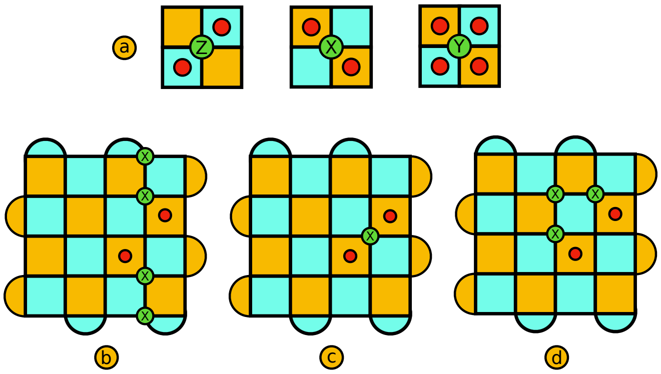

FIG. 2. (a) Single qubit Pauli flips violate surrounding stabilizers. (b-d) Strings of Pauli flips only violate stabilizers at the endpoint of the string. Multiple error configurations can give rise to the same syndrome. They can differ by stabilizers, as for example in (c) and (d), or by logical operators, see (b) and (c).

III. THE DECODING PROBLEM

Given the foundations from the previous section, we are able to succinctly state a simple preliminary version of the decoding problem:

Assume that at $t = 0$ one is given a state $|\psi\rangle = \alpha|0_L\rangle + \beta|1_L\rangle \in \mathcal{H}_{\mathrm{sc}}$. At some time $t_1 > 0$ a syndrome measurement is performed which indicates that one or more stabilizers are violated - i.e. some errors have occurred on physical data qubits. From the given syndrome, determine a set of corrections which should be applied to the code lattice such that the subsequent state $|\psi'\rangle$ is equal to the initial state $|\psi\rangle$.

Before proceeding to discuss more subtle and technical versions of the decoding problem, let's examine why even the above problem is indeed potentially difficult. The most important observation is that the map from error configurations to syndromes is many-to-one, i.e. many different sets of errors can lead to the same syndrome. As an example, consider the error configurations illustrated in lattices (b), (c) and (d) of Fig. 2, all of which lead to the same syndrome. If the probability of an error on a single physical data qubit is low, given such a syndrome one might reasonably assume that the error in lattice (c) occurred, as one error on a single qubit is a more likely event than errors on multiple qubits. Given this reasoning, one might then suggest to correct by applying an X flip on the physical data qubit in the third row and fourth column. If indeed the error in lattice (c) occurred then this would be the correct operation. However, if the error in lattice (d) occurred, then the combination of the original error with the correction would implement a stabilizer. As the original state was a simultaneous +1 eigenstate of all the stabilizers, stabilizer operators act trivially on logical states, and hence the proposed correction preserves the initial logical state. Finally, if the error in lattice (b) occurred, then the combination of the original error with the correction would implement the logical X_L gate. As a result, even though the post-correction state is back in the code space, it will be in a different state to the original state and the information we were trying to preserve would have been corrupted. From this simple example one can see that most often solving the decoding problem as stated above involves deciding, given an inherently ambiguous syndrome and (possibly imperfect) knowledge of the underlying error model, which error configuration was most likely to have occurred.

In addition to the inherent difficulty resulting from syndrome ambiguity, in experimental settings the process of extracting the syndrome is also subject to noise, and as such the syndrome one receives may also contain errors. In practice, each stabilizer is associated with a physical ancilla qubit, and the syndrome value for that particular stabilizer is obtained by first executing a small quantum circuit which entangles the ancilla with each of the physical data qubits on which the corresponding stabilizer is supported, before extracting the syndrome value via a measurement of the ancilla qubit. In order to fully account for errors during the process of syndrome extraction one should therefore model this entire circuit, in which errors can occur on both the data qubits and ancilla qubits at each time step, and importantly, errors on the ancilla qubits can propagate onto the data qubits via the required entangling gates. However, the essential aspect of the additional difficulty from faulty syndrome measurements can be phenomenologically modelled by imagining each time step as consisting of two distinct error processes, as illustrated in Fig. 3. In the first error process, an error occurs on each data qubit with some probability. One then imagines extracting the perfect syndrome before a second error process occurs, in which with a given probability an error occurs on each stabilizer measurement outcome. As single syndromes are no longer reliable, decoding within the setting of faulty syndromes typically requires providing the decoding algorithm with multiple sequential syndromes.

Finally, in the context of surface code based fault tolerant quantum computing, all logical gates are implemented either via protocols which also involve an inherent decoding procedure or do not spread errors. To be specific, it is sufficient to be able to implement both Clifford and T gates. In contemporary proposals for surface code based quantum computing protocols are known for implementing Clifford gates transversally, via code deformation or via lattice surgery. While transversal gates do not spread errors, code deformation and lattice surgery require decoding cycles. Non-Clifford gates, such as the T gate, can be performed fault tolerantly via gate teleportation using magic states. High quality magic states can be obtained via magic state distillation, which requires only Clifford gates and initially faulty magic states. As such the goal of decoding idling logical qubits within a quantum compu-

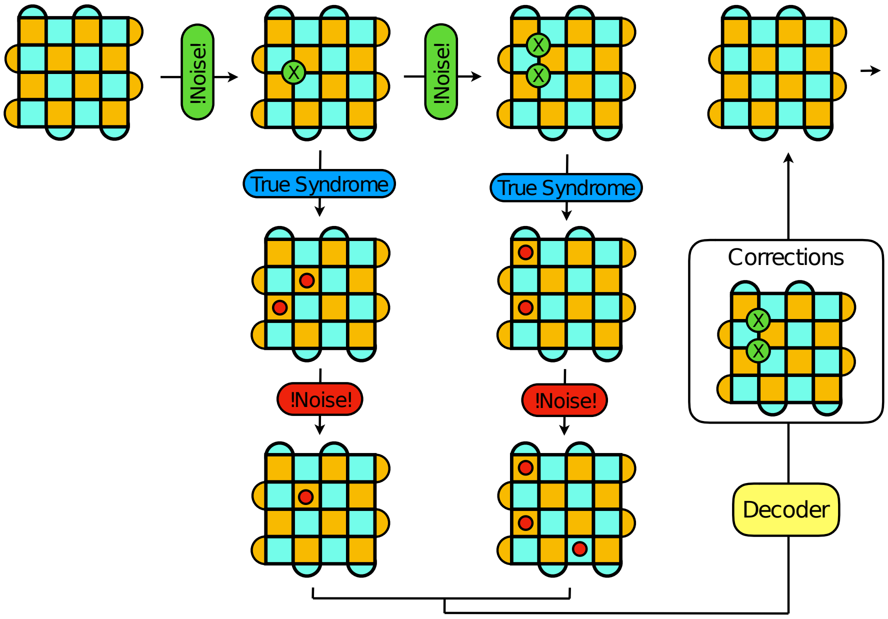

FIG. 3. In the left hand panel, the syndromes associated with various error configurations are illustrated. Note that the error configurations in (a) and (b) lead to the same syndrome. Also, as can be seen in (c), note that continuous strings of errors result in violated stabilizers only at the endpoint of the string. In the right hand panel a typical decoding cycle is illustrated, for the simplified faulty measurements scenario in which one imagines each time step consisting of an initial physical error process generating errors on the data qubits, followed by a second measurement error process which corrupts the true syndrome. The decoding algorithm then has access to a sequence of potentially faulty syndromes.

tation should be to suppress errors to the extent that any of the above procedures can succeed with high probability. Therefore, we can relax the requirement that the decoding process should return the post-error state to the initial state in the code space. In Section V we discuss a proxy criterion for decoding success within this framework.

IV. REINFORCEMENT LEARNING AND Q-FUNCTIONS

In this section we shift focus and introduce some of the fundamental concepts of reinforcement learning and Q-functions, which will be crucial to our conceptual reframing of the decoding problem in Section V. Again, we will omit many interesting details, which can be found in the standard textbook. As illustrated in Fig. 4, we imagine an agent interacting with some environment in discrete time steps. In each time step t the agent is in some state S_t, which is generically a combination of its knowledge of the environment $S_{E,t}$, its memory of its own previous actions, and possibly the state of additional internal variables. From this state the agent must then choose some valid action A_t, which it applies to the environment. In response to this action, via some process which is typically not directly known by the agent, the environment updates and provides the agent with a potentially reduced description of its new state $S_{E,t+1}$ (i.e. a description of the part of the environment which is *visible* to the agent), along with a a scalar reward R_{t+1} and a boolean signal which indicates whether the game is over - i.e. whether the agent has failed, died or lost. In principal it is also possible to imagine settings in which the sequence of interactions can carry on indefinitely, but we will consider here only episodic actions, which end once the environment is placed into a terminal state (which may not be unique) by the actions of the agent. Generically, we consider to goal of the agent to be to avoid dying, while accumulating as much reward as possible.

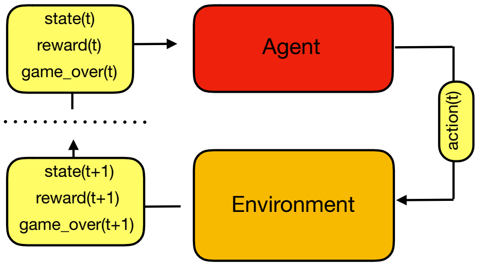

FIG. 4. An illustration of the signals passed between an agent and the environment through the duration of the sequential turn based episodes we consider here.

Typically the way that the agent chooses its action, the way that the environment is effected by the action, and the reward that is generated can all be stochastic, and in the context of finite state and action spaces, denoted $\mathcal{S}$ and $\mathcal{A}$ respectively, this process can be formalized within the framework of finite Markov decision processes (FMDP):

$$p(s',r|s,a) \equiv \mathrm{pr}(S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a). \tag{2}$$

To formalize the decision making process of the agent, we define an agent's *policy* π, in essence the agent's strategy, as a mapping from states to probabilities of specific actions - i.e. $\pi(a|s)$ is the probability that $A_t = a$ if $S_t = s$. For FMDP's we then define the *value* of a state s under policy π as,

$$v_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s] = \mathbb{E}_\pi\Big[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}\Big|S_t = s\Big] \qquad \forall S_t \in \mathcal{S} \tag{3}$$

where G_t is the discounted return (discounted cumulative reward), with discount factor $0 \leq \gamma \leq 1$, and in practice the infinite sum terminates whenever state S_{t+k+1} is a terminal state. We call v the *state-value function*, which provides the expected discounted cumulative reward the agent would obtain when following policy π from state s. It is an important conceptual point to note that by using the metric of the discounted cumulative reward the value of any given state depends not only the immediate rewards an agent would obtain from following a specific policy, and hence strategies which involve some element of succesful future planning may lead to higher state values. As such, we see that the value of a state with respect to a given policy reflects accurately the ability of an agent to achieve its long-term goals when following that policy from that state. Similarly, we can define the *action-value function* (referred to as the Q-function) for policy π via:

$$q_\pi(s,a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a] = \mathbb{E}_\pi\Big[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}\Big|S_t = s, A_t = a\Big] \tag{4}$$

Clearly, the Q function with respect to a given policy is conceptually similar to the value function, except that it provides the value (expected discounted cumulative reward) associated with following a given policy after taking a certain action from a certain state. Importantly, value functions allow us to place an order over policies, i.e. $\pi > \pi' \iff v_\pi(s) > v_{\pi'}(s) \quad \forall s \in \mathcal{S}$. Given such an ordering we can then define an optimal policy π^*, with respect to which the action-value function will be the unique solution of the following system of equations:

$$q_*(s,a) = \mathbb{E}\big[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1},a')\big|S_t = s, A_t = a\big] \tag{5}$$

Note that given the optimal Q-function it is easy to obtain the optimal strategy; When in a given state s simply choose the action $a = \mathrm{argmax}_{a'}[q_*(s,a')]$ Now, the idea of iterative Q-learning is to start from an arbitary q function, and then use eqn. (5) as an update function for an agent which uses the q function to make decisions as to how to interact with the environment. The hope (can be proven under certain constraints) is that the q function will eventually converge to q_*, which is a stationary point of eqn. (1).

More specifically, for *deep*-Q learning we parameterize q with a neural network, and use eqn. (1) to construct the cost function on which we train the network. In particular we let the agent interact with the environment, using some ϵ-greedy policy, in the process generating tuples of experience of the following form:

$$[S_t, A_t, R_{t+1}, S_{t+1}]$$

From these tuples we can construct a loss on which to train the Q-network, by using the cost function:

$$\begin{aligned} C &= y_{\text{pred}} - y_{\text{true}} && (6)\\ &= q(S_t,A_t) - \big[R_{t+1} + \gamma \max_{a'} q(S_{t+1},a')\big] && (7) \end{aligned}$$

which, by staring at eqn. (1), we can see will be 0 only for the optimal policy.

V. DECODING AS A REINFORCEMENT LEARNING PROBLEM

VI. RESULTS

VII. CONCLUSION

ACKNOWLEDGMENTS

The authors gratefully acknowledge helpful and insightful discussions with Daniel Litinski, Nicolas Delfosse, Paul Baireuther, and Hendrik Poulsen Nautrup. Additionally, the authors would like to thank Jörg Behrmann for incredible technical support, without which this work would not have been possible. RS acknowledges the financial support of the Alexander von Humboldt foundation. MSK is supported by the DFG (CRC183, project B02). EvN is supported by ... JE is supported by DFG (CRC 183, EI 519/14-1, and EI 519/7-1), the ERC (TAQ), the Templeton Foundation, and the BMBF (Q.com).

[1] B. M. Terhal, “Quantum error correcton for quantum memories,” Rev. Mod. Phys. **87**, 307 (2015).

[2] Earl T Campbell, Barbara M Terhal, and Christophe Vuillot, “Roads towards fault-tolerant universal quantum computation,” Nature **549**, 172 (2017).

[3] Alexei Yu. Kitaev, “Fault-tolerant quantum computation

by anyons," Ann. Phys. **303**, 2 (2003).

[4] Eric Dennis, Alexei Kitaev, Andrew Landahl, and John Preskill, "Topological quantum memory," J. Math. Phys. **43**, 4452 (2002).

[5] J. Preskill, "Topological quantum computation," (2017), (Chapter 9 of Lecture Notes on Quantum Computation).

[6] Chetan Nayak, Steven H. Simon, Ady Stern, Michael Freedman, and Sankar Das Sarma, "Non-Abelian anyons and topological quantum computation," Rev. Mod. Phys. **80**, 1083 (2008).

[7] Jiannis K Pachos, *Introduction to topological quantum computation* (Cambridge University Press, 2012).

[8] B. J. Brown, D. Loss, J. K. Pachos, C. N. Self, and J. R. Wootton, "Quantum memories at finite temperature," Rev. Mod. Phys. **88**, 045005 (2016).

[9] M. D. Reed, L. DiCarlo, S. E. Nigg, L. Sun, L. Frunzio, S. M. Girvin, and R. J. Schoelkopf, "Realization of three-qubit quantum error correction with superconducting circuits," Nature **482**, 382 (2012).

[10] R. Barends, J. Kelly, A. Megrant, A. Veitia, D. Sank, E. Jeffry, T. C. White, J. Mutus, A. G. Fowler, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, C. Neill, P. O'Malley, P. Roushan, A. Vainsencher, J. Wenner, A. N. Korotkov, A. N. Cleland, and J. M. Martinis, "Superconducting quantum circuits at the surface code threshold for fault tolerance," Nature **508**, 500 (2014).

[11] D. Nigg, M. Müller, E. A. Martinez, P. Schindler, M. Hennrich, T. Monz, M. A. Martin-Delgado, and R. Blatt, "Quantum computations on a topologically encoded qubit," Science **345**, 302–305 (2014).

[12] A. D. Córcoles, Easwar Magesan, Srikanth J. Srinivasan, Andrew W. Cross, M. Steffen, Jay M. Gambetta, and Jerry M. Chow, "Demonstration of a quantum error detection code using a square lattice of four superconducing qubits," Nat. Comms. **6**, 6979 (2015).

[13] S. M. Albrecht, A. P. Higginbotham, M. Madsen, F. Kuemmeth, T. S. Jespersen, J. Nygård, P. Krogstrup, and C. M. Marcus, "Exponential protection of zero modes in Majorana islands," Nature **531**, 206 (2016).

[14] Maika Takita, A. D. Córcoles, Easwar Magesan, Baleeg Abdo, Markus Brink, Andrew W. Cross, Jerry M. Chow, and Jay M. Gambetta, "Demonstration of weight-four parity measurements in the surface code architecture," Phys. Rev. Lett. **117**, 210505 (2016).

[15] Norbert M. Linke, Mauricio Gutierrez, Kevin A. Landsman, Caroline Figgatt, Shantanu Debnath, Kenneth R. Brown, and Christopher Monroe, "Fault-tolerant quantum error detection," Sci. Adv. **3**, e1701074 (2017).

[16] Austin G Fowler, "Minimum weight perfect matching of fault-tolerant topological quantum error correction in average $o(1)$ parallel time," arXiv preprint arXiv:1307.1740 (2013).

[17] Guillaume Duclos-Cianci and David Poulin, "Fast decoders for topological quantum codes," Physical review letters **104**, 050504 (2010).

[18] Michael Herold, Earl T Campbell, Jens Eisert, and Michael J Kastoryano, "Cellular-automaton decoders for topological quantum memories," npj Quantum Information **1**, 15010 (2015).

[19] Giacomo Torlai and Roger G Melko, "Neural decoder for topological codes," Physical review letters **119**, 030501 (2017).

[20] Savvas Varsamopoulos, Ben Criger, and Koen Bertels, "Decoding small surface codes with feedforward neural networks," Quantum Science and Technology **3**, 015004 (2017).

[21] Stefan Krastanov and Liang Jiang, "Deep neural network probabilistic decoder for stabilizer codes," Scientific reports **7**, 11003 (2017).

[22] Nikolas P Breuckmann and Xiaotong Ni, "Scalable neural network decoders for higher dimensional quantum codes," Quantum **2**, 68 (2018).

[23] Paul Baireuther, Thomas E O'Brien, Brian Tarasinski, and Carlo WJ Beenakker, "Machine-learning-assisted correction of correlated qubit errors in a topological code," Quantum **2**, 48 (2018).

[24] P Baireuther, MD Caio, B Criger, CWJ Beenakker, and TE O'Brien, "Neural network decoder for topological color codes with circuit level noise," arXiv preprint arXiv:1804.02926 (2018).

[25] Xiaotong Ni, "Neural network decoders for large-distance 2d toric codes," arXiv preprint arXiv:1809.06640 (2018).

[26] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," Nature **518**, 529 (2015).

[27] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, *et al.*, "Mastering chess and shogi by self-play with a general reinforcement learning algorithm," arXiv preprint arXiv:1712.01815 (2017).

[28] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, *et al.*, "Mastering the game of go without human knowledge," Nature **550**, 354 (2017).