Quantum Error Mitigation with Deep Learning and Sequence Models

Andrej Hulák, Vincenzo Lipardi[0009−0000−3243−7969], Domenica Dibenedetto[0000−0002−2538−3170], and Kurt Driessens[0000−0001−7871−2495]

Department of Advanced Computing Sciences,
Maastricht University, The Netherlands
{a.hulak,vincenzo.lipardi,domenica.dibenedetto,
kurt.driessens}@maastrichtuniversity.nl

Abstract. The primary promise of quantum computing lies in its potential to surpass classical computation by leveraging quantum mechanical effects. However, the presence of noise in quantum hardware is a crucial problem that limits the time and size of any quantum computation. The field of Quantum Error Mitigation emerged as an alternative approach to Quantum Error Correction in order to reduce errors on noisy intermediate-scale quantum devices at the cost of additional computation time. This paper presents a proof of concept deep learning approach to mitigate quantum noise and includes preliminary experimental results that demonstrate the potential of sequential models in this domain. Different from current state of the art, these models allow the sequential nature of quantum circuits to be encoded and used to predict and mitigate quantum noise. Experiments using Clifford circuits show that sequential models can deal with different structures of circuits with limited computation overhead during the inference stage. Results are compared to the current state of the art machine learning technique for Quantum Error Mitigation as well as to the conventionally used zero-noise extrapolation. Preliminary experimental results show that the use of sequence models should not be overlooked as they can provide advantages over current state of the art methods.

Keywords: Quantum Error Mitigation · Deep Learning · Sequence Models.

1 Introduction

Achieving a practical quantum advantage in the Noisy Intermediate-Scale Quantum (NISQ) era is one of the most desired objectives in the field of quantum computing. A primary obstacle in this pursuit is the presence of quantum noise that corrupts the information flow in the hardware. Quantum noise refers to different types of sources and consequent errors that can affect quantum systems:

- **Coherent Noise**: errors that maintain phase relationships and can be caused by systematic issues like control errors and unwanted interactions.

- **Incoherent Noise**: random errors that result from interactions with the environment, leading to loss of coherence in the quantum state.
- **State Preparation and Measurement (SPAM) Noise**: errors that occur during the initialization of qubits and during the readout process.
- **Projection Noise**: errors associated with the probabilistic nature of quantum measurement, affecting the precision of the measurement outcomes.

To address these issues, concurrent advancements are being made on both hardware and software fronts. Hardware improvements focus on developing devices with a high number of qubits, enhanced connectivity between them, and one above all reduced gate errors. On the software side, efforts concentrate on algorithmic solutions that mitigate computing errors by leveraging additional computational resources and on the design of algorithms that employ shallower quantum circuits.

In this scenario, there are two main approaches to address quantum noise in quantum hardware:

- **Quantum Error Correction (QEC)** has seen fast development yielding pivotal theoretical [4,17] and experimental results [1]. Pivotal is the threshold theorem that demonstrates that fault-tolerant quantum computation on noisy quantum hardware is possible if errors could be reduced below a finite threshold [2]. However, implementing these QEC techniques requires a significantly larger number of physical qubits in order to realise the logical qubits required for computation. As such, QEC represents the natural solution for achieving fault-tolerant quantum computers [5], but it shows limitations on NISQ devices [16].
- **Quantum Error Mitigation (QEM)** emerges as an approach tailored on NISQ devices. In fact it does not require extra qubits but a post-processing procedure [10] that leverage classical computational resources. Famous examples include Zero-Noise Extrapolation (ZNE) [7], Probabilistic Error Cancellation (PEC) [19] and Virtual Distillation (VD) [8]. Nevertheless, QEM techniques may face challenges in fault-tolerant quantum computers, as they require a large number of quantum measurements on several circuits that grow exponentially with the number of qubits, resulting in substantial runtime overheads [3]. The issue of the runtime overhead is not only eventually significant on fault-tolerant devices but also on NISQ devices. This is one of the main challenges for QEM methods nowadays.

We remark the role that QEM plays on quantum computing for near-term applications, as shown in the previous noteworthy attempts [6,12].

Nevertheless, the exponential sampling overhead associated with QEM, due to the increasing number of quantum circuits to run in order to have good error mitigation estimator, pose significant challenges [3]. The solution to these sampling overheads lies in the generalisation of noise modelling and making a QEM approach that can be circuit independent, or at least, that can predict the noisy behaviour of a circuit based on data collected for other circuits. This is the main idea behind QEM-ML methods [11,18,13], that employ Machine Learning and its generalisation capabilities to offer a cost-effective alternative. The idea is to trade training overheads for inference time overhead. Current state of the art has tested various machine learning methods, including linear regression, random forests, multi-layer perceptrons, and graph neural networks, and compared their performance across different circuits, noise models, and applications [13]. Random forests (RF), as per usual, are quite successful at balancing complexity and efficacy. They notably also outperform ZNE in terms of performance as well as running time.

This early state of the art comes with many interesting research questions that might be addressed [13]. The best-performing model, random forests, extracts simple quantum circuit features but underutilised the circuit's structure. Alternative models, like graph neural networks, that did use the circuit's structure, did not perform as well as RF and ZNE [13].

In this work, we focus on deep learning, addressing the question of whether and to what extent the ordered structure of a quantum circuit can be used to learn and subsequently mitigate noise effects to achieve a specific objective under limited resources and varying noise levels.

Contributions In this paper, we perform an initial investigation into the use of the sequential circuit structure [13] and how can we expand the set of features used to represent the quantum circuit, with the intend of creating a QEM-ML method that can better generalise to different and most importantly longer, quantum circuits. We show that even a simplified sequential deep learning model is capable of mitigating the effects of quantum noise without the need for additional mitigation circuits at inference time.

We present a model structure that leverages the sequential nature of the quantum circuit and information on the backend that executed the quantum circuit and even without taking the connectivity between different qubit paths into account, reaches the same performance as conventional zero-noise extrapolation and the recently proposed random forests approach. This new model combines long short-term memory (LSTM) with a classical feed-forward network, enabling it to mitigate errors in circuits with greater circuit depth more effectively. We illustrate this performance on circuits of various depths. The analysis indicates that circuit structure heavily influences the quality of the mitigated result and is something that should not be overlooked. We aim to stimulate discussion in the emerging field of QEM-ML and show, through limited experiments, that deep learning models present a promising approach to enhance the accuracy of quantum error mitigation while maintaining relatively low training overhead.

2 QEM-ML method

In this chapter, we delve into the core methodology of our contributions to QEM-ML. We begin by examining the process of encoding quantum circuits into a set of features that effectively capture their sequential gate structure. The aim is

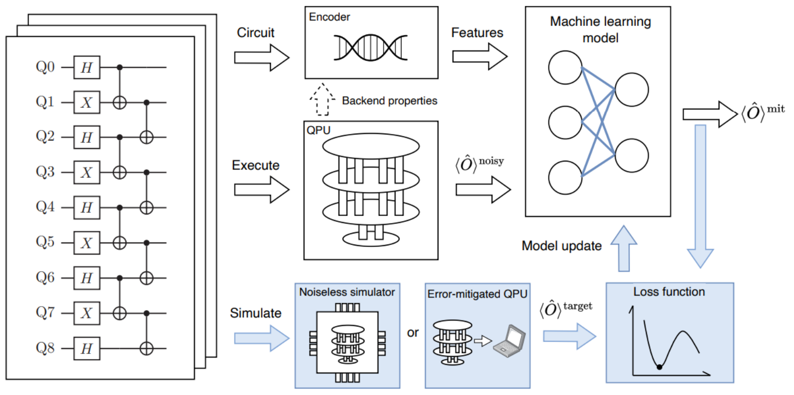


Fig. 1: The Quantum Error Mitigation with Machine Learning (QEM-ML) framework by the paper Machine Learning for Practical Quantum Error Mitigation [13]. We pass a quantum circuit into an encoder which creates a set of features that are then, together with a noisy expectation value passed into a ML model. This model then makes a prediction which is compared with the ideal expectation value. The error then backpropagates through the parameters of the ML model that updates its weights accordingly. The result should then be a model that is capable of predicting the noise-free expectation value [13].

to transform quantum circuits into a latent vector space, from which a neural network can predict expected noise and produce mitigated expectation values (see Fig.1). This chapter is structured to guide the reader through the encoding strategies, the analogy used for understanding, and the specific technical steps involved.

2.1 Encoding: Quantum Circuits as Sequences

Encoding a quantum circuit into a set of features is at the heart of every QEM-ML method [13]. We want to be able to encode quantum circuits into latent vector space from which a neural network can estimate the expected noise, and, when combined with the noisy expectation value, produce a mitigated expectation value. Because we assume that information about the full sequential shape of the quantum circuit is important to accomplish this, we want the encoding to be able to retain the circuit structure. To do so, we will make use of an analogy that will hopefully help the reader understand the thought process behind the encoding strategy.

An NLP Analogy We can imagine the sequence of operations applied to a qubit as a sentence, where each gate functions as a word within that sentence. In this analogy, a quantum circuit becomes a collection of parallel, interacting sentences, one for each qubit. Each sentence begins with a *word* that symbolizes the characteristics of the particular qubit on which the rest of the sentence will act. Subsequent words, representing gates, are defined by a combination of 7 features that describe the gate.

Where the analogy becomes a bit obfuscated, is in the fact that these sentences are indeed interacting, namely at any point where a two-qubit gate, such as a CNOT, is used. For now, we will represent CNOT gates only at the single sentence level, meaning for each qubit separately. Modelling the interactions between accumulated noise information between different qubits is left for future work.

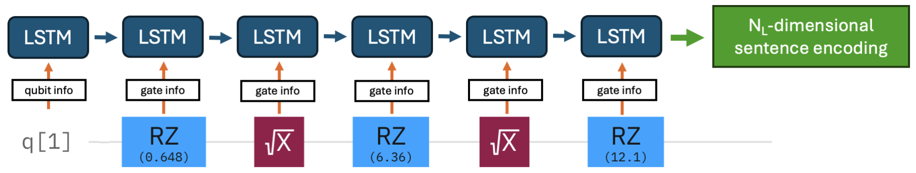


Fig. 2: Encoding the quantum circuit influencing a qubit as a sequence of gates. The Figure shows a Recurrent Neural Network — in this case an LSTM, see Section 2.2 — rolled out in the "time-dimension" to which all information about the used gates is given in sequence. All connections processing gate-information and passing forward the partial encoding of the circuit use the same weight matrix.

Tokenizing Quantum Gates The quantum circuit encoding is preceded by the choice of a backend. Then, the circuit is transpiled into hardware-native gates that adhere to the used quantum device's connectivity. We used the native transpilation procedure in Qiskit [9]. The transpilation process dictates the order of gates (words) in the sentences and is crucial because it allows the model to construct a hardware specific noise profile, as circuit noise profiles are heavily hardware dependent [9,14]. After transpiling the circuit on the desired backend, we obtain the hardware specific operations through the data attribute of the QuantumCircuit object in Qiskit [9]. Each operation specifies the qubit it is applied to, the type of operation, and other relevant parameters, all of which can then contribute to the encoding and the overall noise profile.

Gate/Qubit Representations The transpiled quantum circuit provides the required backend information and the feature list for each gate and qubit. Each feature list has length 7. Gates use the first value to represent the type of gate used. This uses a label encoding with a dictionary for each unique gate-integer pair available in the Qiskit library [9,13]. The 2nd entry represents the angle parameter of rotation gates. If the gate is not a rotation gate, this entry is set to 0.

Qubits use the 3rd, 4th and 5th values. At the start of a sentence representing a qubit and its operations, we record a special "word" for the parameters of the corresponding qubit. The parameters available for the qubit are the relaxation time T1, the dephasing time T2 and the readout error [9].

Additionally, we also make use of the gate error and gate length for each gate used in the quantum circuit (features 6 and 7) that are native to that backend. The main difference for two-qubit gates is that gate errors and gate lengths are different depending on the specific qubit of the pair (again obtained from the backend).

2.2 The Network Architecture

To encode all operations on a qubit in sequence, we employ a recurrent neural network (RNN) that can encode the sequence of gates. Information about how the circuit manipulates a specific qubit is fed to the recurrent network sequentially. The RNN uses that sequential information to encode the part of the circuit that deals with that qubit and its resulting noise profile.

Long Short-Term Memory Long Short-Term Memory networks (LSTMs) are a specialized type of recurrent neural network designed to overcome the limitations of traditional RNNs (such as the Elman Recurrent Neural Networks) in capturing long-term dependencies in sequential data [20]. Introduced by Hochreiter and Schmidhuber in 1997, LSTMs excel in tasks requiring context over extended sequences. They achieve this by incorporating gating mechanisms to selectively retain or forget information across different steps of the sequence to be encoded, resolving issues like vanishing gradients [20].

Figure 2 shows how the LSTM is used to encode qubit operations. We instantiate a separate sequence model (LSTM) for each qubit available to the backend. The sequence model processes the circuit specifics into an N_L-dimensional latent space that should help predict the noise-profile of the circuit.

Complete Network Architecture The different latent noise-profile representations for each of the qubits are simply concatenated and fed into a standard feedfoward network together with the index and unmitigated, noisy, expectation value of the qubit we want to predict the mitigated value for.

Hyperparameters such as the number of LSTM units and layers, the number of hidden units in the feed-forward network, dropout and learning rates were determined through a grid search described in the experimental set-up.

This network architecture is overly simple and only suggested for the proof of concept in this paper. This simple approach nicely illustrates the promise of sequential encodings of the quantum circuit as even in this limited approach, experiments will show that it can reach the same level of performance as state of the art techniques.

The fact that a different LSTM is used for each qubit might seem superfluous at first glance, and certainly isn't training data efficient. However, the use of a

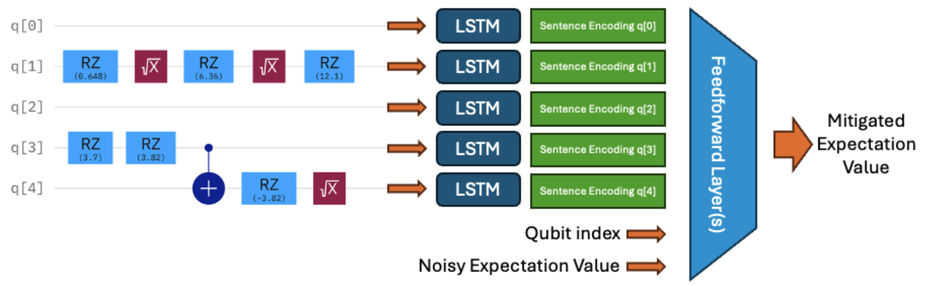


Fig. 3: The QEM model architecture. The encoded qubit operations are concatenated and extended with a qubit index and the measured, noisy expectation value for that qubit. The goal of the network is to predict a mitigated, less noisy expectation value for the qubit.

single LSTM to encode the operations on all qubits might also be suboptimal as actual hardware implementations of different qubits might show differences in their noise profiles beyond what is captured by the qubit and gate encodings we currently feed into the LSTM.

Additional architectural design that for example can take qubit interconnections such as those caused by two-qubit gates, or that offer a more elegant solution to qubit identification compared to the used qubit index input in the feed-forward network are expected to bring the performance of sequentially capable methods beyond the current state of the art.

3 Experimental Setup

In this section we describe the experimental setup consisting of the data generation process and the details of the deep learning model. We evaluate our deep learning model for quantum error mitigation across two types of quantum circuits: unstructured random circuits and structured circuits derived from the Ising model. For both circuit types, we vary the circuit depth to assess the model on different quantum noise conditions. Circuit depth refers to the number of two-qubit gates applied in a quantum circuit, that in our case are CNOTs. This is because the noise generated by those gates is several orders of magnitudes bigger than one-qubit gates. Varying the circuit depth allows to evaluate how our model performs under different noise conditions, with shallow circuits being less affected by noise but limited in computational complexity, and deeper circuits being more prone to noise but capable of handling more complex computations.

We evaluated a set of quantum circuits using a noise-free setting and obtaining a noisy expectation value through simulation. We then allow the model to train on predicting the noise-free expectation values using the circuit details and the noisy expectation value as input. We calculate the error of the prediction as the squared difference between the mitigated expectation value and the expectation value obtained in the noise free setting.

We instantiate, train and test the prediction models using the PyTorch library [15]. The models are trained using a mean squared error (MSE) loss function and the Adam optimizer. The parameters tuned by the optimizer are the sequence model and the feed-forward network parameters. The number of epochs has been set to 5 which seemed sufficient to reach convergence.

3.1 Data Generation

To ensure a rigorous comparison with previous work, we generated data as discussed by Daley et al. [6], as they share a public Github repository including data and the code to generate it. We work with two main classes of data. The first class consists of random 4-qubit quantum circuits, where we measure the expectation value of the observable given by the $\hat{Z}$ Pauli on each qubit: $\hat{O} = \hat{Z}_0 \otimes \hat{Z}_1 \otimes \hat{Z}_2 \otimes \hat{Z}_3$. For this class we generated 500 training circuits and 200 testing circuits for three different *circuit sizes* as defined by the number of two-qubit gates:

- *shallow* with a number of CNOTs ranging from 1 to 18;
- *medium* with a number of CNOTs ranging from 20 to 30;
- *deep* with a number of CNOTs ranging from 30 to 40.

The second class of circuits consists of Trotterized circuits. Specifically, we use first-order Trotterization to approximate the evolution of the 1D transverse-field Ising model, where the continuous time evolution is broken into discrete steps. Each step alternates between applying the Hamiltonian's interaction terms and the transverse-field terms. Unlike the first class of circuits, this class imposes a strong structure on the quantum circuit, as there are patterns of gates that repeat. This circuit form was chosen because it represents a number of different quantum computing applications. This class was subdivided in terms of *circuit depth*, that is a measure of how many "layers" of quantum gates, executed in parallel, it takes to complete the computation defined by the circuit [21]. Analogue to the first class of circuits, we generated 500 training circuits and 200 testing circuits for each size:

- *shallow* with a depth ranging from 1 to 60;
- *medium* with a depth ranging from 60 to 120;
- *deep* with a depth higher than 120.

In order to train our supervised model we had to generate both the noisy and ideal expectation value for each circuit. For this purpose we used Qiskit, that gives access to back-ends that simulate the error in IBM's real devices. In our case, we simulated the noisy output on the FakeLima back-end, which simulated the incoherent noise. We used a noise-less qasm simulator to calculate the ideal expectation values. The choice of this dataset facilitates direct performance comparisons and reproducibility of results. We align our experimental setup with existing benchmarks, thereby enabling a transparent assessment of our contribution. Note that in theory this process can be employed to study Clifford circuits but it runs into key scaling limitations on circuits that are not efficiently simulated classically.

3.2 Training and Hyperparameter Search

We employed a grid search to tune the hyperparameters, including the number of layers, hidden units, dropout rates, and learning rate. The models have been evaluated using the Root Mean Squared Error (RMSE).

We train the model for each qubit for 4 training epochs, which proved sufficient for the chosen the learning rates. In fact, increasing the number of epochs resulted in the model starting to overfit and a lower accuracy on the validation set. After training, we evaluate the performance of the model and save the results, the model parameters and the corresponding configuration of hyperparameters. We do this for each hyperparameter combination to obtain the hyperparameters and the best model parameters that yield the lowest RMSE.

The obtained hyperparameters are the same for each dataset and reported in Table 1.

Table 1: Hyperparameters of the model chosen after a grid search.

	Parameter Name	**Value**
LSTM	Hidden Layers	2
	Hidden Units	32
	Dropout Rate	0.05
FFN	Hidden Layers	2
	Hidden Units	64
	Dropout Rate	0
	Learning Rate	0.07

4 Experimental Results

We evaluated the performance of the model with the Random Forest implemented in [13], representing the state of the art on the datasets chosen. The metric used for the comparison is the Root Mean Square Error (RMSE), and the main results are provided in Table 2. Although the structure of the sequential model implemented is still naive it reaches comparable results to RF methods that do not incorporate sequential information. This hints at the importance of capturing the order of operations in quantum error mitigation. Additionally, we analyzed the impact of circuit depth on performance. Our model maintained strong performance with increasing depth, whereas other models showed a decline. However, when comparing circuits with varying numbers of CNOT gates, we observed a slight decrease in performance for our model. This was anticipated, as our approach does not account for channel interference effects introduced by multiple CNOT operations. Despite this, our method still performs competitively, highlighting its robustness and the importance of incorporating sequential information in quantum circuit analysis.

We observe that for the Ising model, our approach does not outperform RF. We assume this can be attributed to the ability of RF to effectively capture patterns, even when they are repetitive or redundant, due to their randomness in feature selection, which prevents overemphasis on any single feature and makes them better-suited for homogenous data with repeated patterns. In contrast, while neural networks can learn complex patterns, they are prone to overfitting, particularly with redundant information, as they may over-focus on repeated patterns, leading to poor generalization and performance on unseen data.

Table 2: Results on unstructured and structure class of quantum circuits. The table presents the RMSE values averaged over the four qubits.

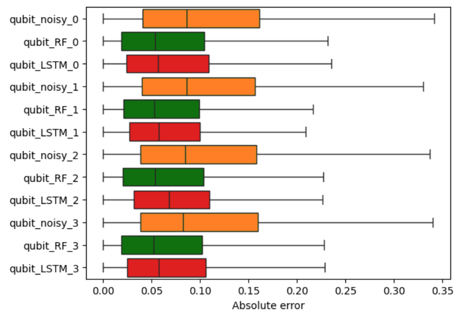
Random Circuits			
Model	Shallow	Medium	Deep
Noisy	0.15	0.19	0.22
Random Forest	0.10	0.15	0.19
LSTM+FFN	0.10	0.15	0.18
Ising Circuits			
Model	Shallow	Medium	Deep
Noisy	0.07	0.09	0.12
Random Forest	0.01	0.02	0.03
LSTM+FFN	0.03	0.04	0.06

5 Discussion

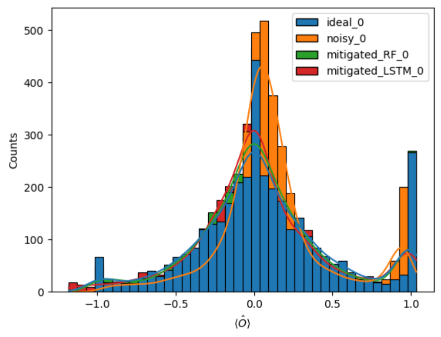
We found that the new model architecture provides performance that is on par with, or slightly superior to, zero-noise extrapolation and random forest predictions. This performance was achieved with reasonable training times and without the need for additional mitigation circuits.

The experiments demonstrate that the order of gates and operations significantly impacts model accuracy, and that simple machine learning models can be enhanced by incorporating sequential circuit information.

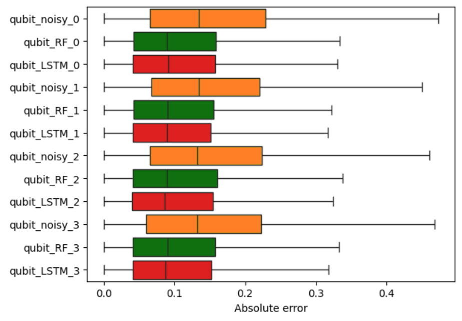
The study primarily uses simulated quantum circuits for training and testing. Applying the model to real quantum devices with their inherent noise characteristics and variability is a critical next step. This will help validate the model's practical applicability and identify any additional challenges that may arise.



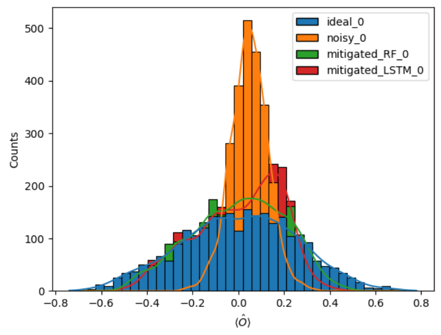
(a)



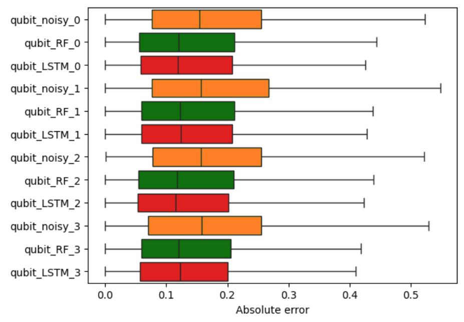
(b)



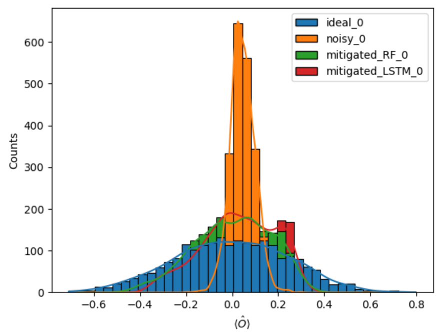
(c)



(d)

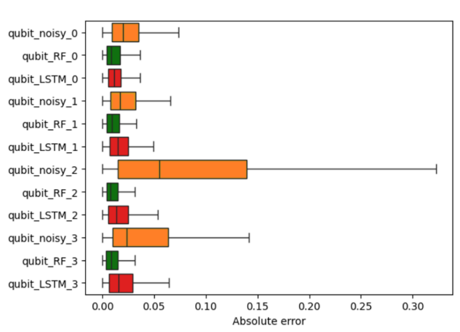


(e)

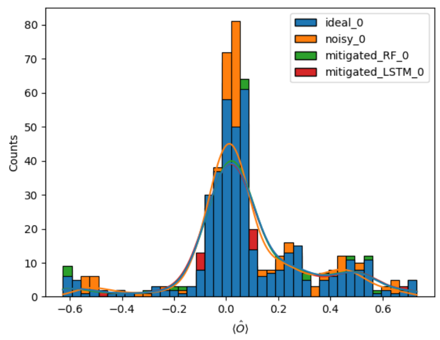


(f)

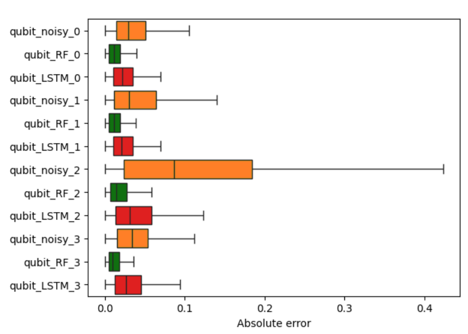
Fig. 4: Comparing model performance on the Random Circuits Dataset. The bar plots on the left show the absolute error obtained by the LSTM model compared to Random Forest (RF) and the noisy version for each of the four qubits (0, 1, 2, 3) of the circuits. The plots on the right display the error distribution of the noisy and mitigated models against the ideal expectation for qubit 0, across different circuit depths: shallow ((a), (b)), medium ((c), (d)), and deep ((e), (f)).



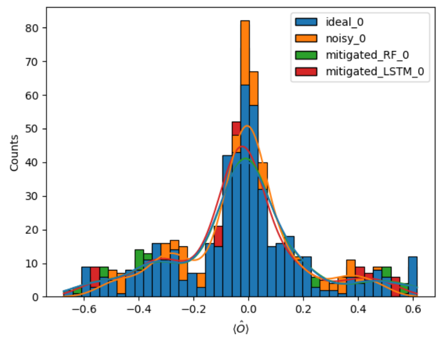
(a)



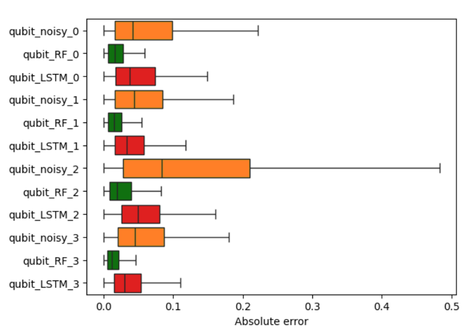
(b)



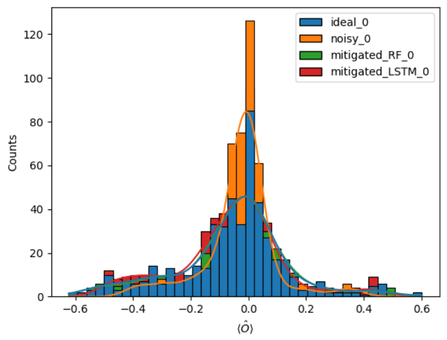
(c)



(d)



(e)



(f)

Fig. 5: Comparing model performance on Trotterized Ising Circuit Dataset. The bar plots on the left show the absolute error obtained by the LSTM model compared to Random Forest (RF) and the noisy version for each of the four qubits (0, 1, 2, 3) of the circuits. The plots on the right display the error distribution of the noisy and mitigated models against the ideal expectation for qubit 0, across different circuit depths: shallow ((a), (b)), medium ((c), (d)), and deep ((e), (f)).

We acknowledge the importance of hyperparameter tuning for model performance. A more exhaustive exploration of the hyperparameter space, possibly using automated tuning techniques, could further enhance the model's accuracy and efficiency. Furthermore, we want to highlight limitations on the scalability of the data generation even with the use of a quantum hardware. Since we employ supervised learning we must know the noiseless target values, which are efficiently computable only on the restricted class of stabilizer states.

6 Conclusions

In this article we have investigated the encoding of quantum circuits for the implementation of a deep learning model for quantum error mitigation. Specifically, we explored the potential of leveraging the ordered sequence of gates using sequential models. We compared our model with the state-of-the-art Random Forest implemented in [13], obtaining comparable results on different quantum circuit size. Our preliminary numerical results are promising and warrant further investigation.

Here are several key directions for future work. First, enhancing the model's architecture could involve developing more sophisticated approaches that better integrate qubit interconnections and more precisely handle interactions between qubits. Additionally, finding more effective feature representations could improve the model's ability to capture relevant circuit characteristics. Improving the scalability of the data generation process is also crucial for advancing the application of these models. Future research could involve testing the model's performance through multiple experiments that explore various methods of defining circuit depth, taking into account the different classifications of shallow, medium, and deep circuits used for the two classes of circuits investigated, as well as employing different matrices for comparison. Finally, we can compare the performance of different machine learning architectures using various types of circuit classes. This approach would offer a deeper understanding of how the models perform across different inputs, particularly as they vary in complexity and homogeneity.

We hope that this work will contribute to the ongoing discussion around quantum error mitigation within the machine learning community.

Data Availability The data and code of this study are openly available at GitHub repository.

Acknowledgments. The project has been carried out by A. Hulák under the supervision of V. Lipardi, K. Driessens and D. Dibenedetto during his participation to the "MaRBLe" honours programme as part of his Bachelor in Data Science at the Maastricht University.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Acharya, R., Aghababaie-Beni, L., Aleiner, I., Andersen, T.I., Ansmann, M., Arute, F., Arya, K., Asfaw, A., Astrakhantsev, N., Atalaya, J., et al.: Quantum error correction below the surface code threshold. arXiv preprint arXiv:2408.13687 (2024)
2. Aharonov, D., Ben-Or, M.: Fault-tolerant quantum computation with constant error. In: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing. pp. 176–188 (1997)
3. Cai, Z., Babbush, R., Benjamin, S.C., Endo, S., Huggins, W.J., Li, Y., McClean, J.R., O'Brien, T.E.: Quantum error mitigation. Reviews of Modern Physics **95**(4), 045005 (2023)
4. Calderbank, A.R., Shor, P.W.: Good quantum error-correcting codes exist. Physical Review A **54**(2), 1098 (1996)
5. Campbell, E.T., Terhal, B.M., Vuillot, C.: Roads towards fault-tolerant universal quantum computation. Nature **549**(7671), 172–179 (2017)
6. Daley, A.J., Bloch, I., Kokail, C., Flannigan, S., Pearson, N., Troyer, M., Zoller, P.: Practical quantum advantage in quantum simulation. Nature **607**(7920), 667–676 (2022)
7. Giurgica-Tiron, T., Hindy, Y., LaRose, R., Mari, A., Zeng, W.J.: Digital zero noise extrapolation for quantum error mitigation. In: 2020 IEEE International Conference on Quantum Computing and Engineering (QCE). pp. 306–316. IEEE (2020)
8. Huggins, W.J., McArdle, S., O'Brien, T.E., Lee, J., Rubin, N.C., Boixo, S., Whaley, K.B., Babbush, R., McClean, J.R.: Virtual distillation for quantum error mitigation. Physical Review X **11**(4), 041036 (2021)
9. Javadi-Abhari, A., Treinish, M., Krsulich, K., Wood, C.J., Lishman, J., Gacon, J., Martiel, S., Nation, P.D., Bishop, L.S., Cross, A.W., et al.: Quantum computing with qiskit. arXiv preprint arXiv:2405.08810 (2024)
10. Kandala, A., Temme, K., Córcoles, A.D., Mezzacapo, A., Chow, J.M., Gambetta, J.M.: Error mitigation extends the computational reach of a noisy quantum processor. Nature **567**(7749), 491–495 (2019)
11. Kim, C., Park, K.D., Rhee, J.K.: Quantum error mitigation with artificial neural network. IEEE Access **8**, 188853–188860 (2020)
12. Kim, Y., Eddins, A., Anand, S., Wei, K.X., Van Den Berg, E., Rosenblatt, S., Nayfeh, H., Wu, Y., Zaletel, M., Temme, K., et al.: Evidence for the utility of quantum computing before fault tolerance. Nature **618**(7965), 500–505 (2023)
13. Liao, H., Wang, D.S., Sitdikov, I., Salcedo, C., Seif, A., Minev, Z.K.: Machine learning for practical quantum error mitigation. arXiv preprint arXiv:2309.17368 (2023)
14. Nielsen, M.A., Chuang, I.L.: Quantum computation and quantum information, vol. 2. Cambridge university press Cambridge (2001)
15. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library (2019), `https://arxiv.org/abs/1912.01703`
16. Preskill, J.: Quantum computing in the nisq era and beyond. Quantum **2**, 79 (2018)
17. Steane, A.M.: Error correcting codes in quantum theory. Physical Review Letters **77**(5), 793 (1996)

18. Strikis, A., Qin, D., Chen, Y., Benjamin, S.C., Li, Y.: Learning-based quantum error mitigation. PRX Quantum **2**(4), 040330 (2021)
19. Van Den Berg, E., Minev, Z.K., Kandala, A., Temme, K.: Probabilistic error cancellation with sparse pauli–lindblad models on noisy quantum processors. Nature physics **19**(8), 1116–1121 (2023)
20. Vennerød, C.B., Kjærran, A., Bugge, E.S.: Long short-term memory rnn (2021), `https://arxiv.org/abs/2105.06756`
21. Wille, R., Van Meter, R., Naveh, Y.: Ibm's qiskit tool chain: Working with and developing for real quantum computers. In: 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE). pp. 1234–1240. IEEE (2019)