Noise influence on Quantum Machine Learning models' performance

Michał Bączyk, supervisor Dr Sofia Vallecorsa
AI and Quantum group at CERN openlab

Abstract
We analyse how the training and performance of VQC models is affected by noise inherent to NISQ devices. In particular, we study the influence of three different types of quantum hardware noise: measurement errors, single qubit gate errors, and two-qubit gate errors (e.g., CNOT gate). Furthermore, we train the previously mentioned QML algorithms using noise models that emulate the behaviour of available quantum computers with high accuracy. We conclude that the tested QML models are suitable for operation on current NISQ devices.

Keywords
Variational Quantum Classifier, QML architectures, noise, errors, learning, ROC curve, AUC score

1 Introduction and outline

1.1 Opening remarks

In this work, we study Quantum Machine Learning architectures called Quantum Classifiers in the particle physics context of trying to distinguish from the background the events in which Higgs boson was produced. The study is a continuation of the work done inside the AI and Quantum group at CERN openlab which recently was published [1]. This particular research focuses solely on the Variational Quantum Circuit architecture proposed for the given classification task in [1]. The effects of the different noise components in NISQ devices are explored as we transition from ideal simulation studies to implementing the developed algorithms on available quantum computers. This work serves as a crucial step in the group's ongoing efforts towards robust QML applications in High Energy Physics.

1.2 Report outline

This report presents the results of the simulations for the noise studies. It is also complemented by the technical details valuable for future members of the group.

In Section 2 we describe used computing resources, details of how to use the cluster's computing abilities and how to run multiple machine learning jobs at the same time. Section 3 constitutes the main part of the report in which we present the performed experiments with regards to noise effects studies and we discuss them in the light of the state of the current quantum computing hardware capabilities. We conclude the work in Section 4.

2 Technical setup

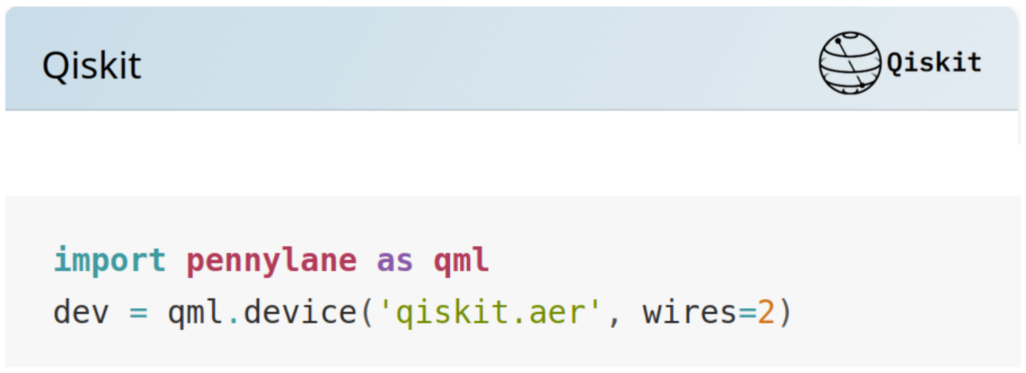
In this section we introduce the computing platform that allowed for extensive training of Quantum Machine Learning models.

2.1 Quantum computing toolkit

The studies presented in this report are based upon the code written by Samuel Gonzalez Castillo. The code, to incorporate the possibility of training quantum circuits, employs the cross-platform Python library *Pennylane*[1]. *Pennylane* software does allow for efficient simulation of quantum circuits and differentiation thereof.

For the studies of how different noise models affect the training of the QML architectures we decided to use open source SDK (software development toolkit) *Qiskit*[2]. Qiskit provides means of implementing custom noise models with the parameter and specifics chosen by the user and furthermore it allows to test the models with the noise emulating the real hardware environment. Hence, such a solution offers a comprehensive playground for testing the suitability of using QML methods on current quantum hardware.

To take advantage of the combined use of both *Pennylane* and *Qiskit*, we utilized the *Qiskit* plug-in inside *Pennylane*. To do so, one has to specify that the quantum device used for circuit classical simulation is the qiskit simulator as in Figure 1.



```
import pennylane as qml
dev = qml.device('qiskit.aer', wires=2)
```

Figure 1: Figure presents how to specify the device(*dev*) used by *Pennylane* to the the *Qiskit* quantum simulator. This embedding allows us to make use of both of the libraries simultaneously.

Technical remarks (as of August 2021) for using the *Qiskit* plug-in inside the *Pennylane*:

- qiskit.aer requires approximately 60 times more time than *Pennylane*'s default.qubit for training 4-qubit VQC – ZZ feature map encoding, 4 repetitions of 2local variational form, ZZ feature map reuploading, 4 repetitions of 2local variational form with 2000 points of training data.
- qiskit.ibmq simulators (simulatorstatevector, qasmsimulator) experience additional 30 fold slowdown with respect to qiskit.aer.
- The slowdown occurs both for the hybrid NNVQC and purely quantum version of the VQC.
- qiskit.aer and qiskit.ibmq are only compatible with hardware compatible differentiation methods (see link[3]).
- The reason for the slowdown is not the chosen differentiation method. *Pennylane* can run equally fast (factor of 2 slowdown which in negligible compared to the *Qiskit* slowdowns) with hardware compatible differentiation methods as with simulator and default differentiation methods.
- There is not slowdown when *Qiskit* plug-in in *Pennylane* when is used solely to transpile and run circuits on their own. Slowdown seems to occur only if the optimization algorithms are taken into account (contact Samuel Gonzalez Castillo for more details regarding this insight).

[1]https://pennylane.ai/
[2]https://qiskit.org/
[3]https://pennylane.readthedocs.io/en/stable/introduction/interfaces.html

2.2 Assessing the performance of quantum classifiers

The evaluation of the investigated QML classifier models in Section 3 follows the same procedure as in [1]. The main indicator of the ”goodness” of the classifier, hence, will be taken to be its AUC score.

3 Noise effects on the models' performance

3.1 Outline of the study

We divide our study into two interconnected parts. Firstly, we investigate the custom noise models in which we can set the error rates values and decide which qubits might be affected by the noise persisting in the circuit. This allows us to examine distinct types of noise in isolation with respect to one another. We perform experiments with three different types of noise arising because of: measurement/readout error, single qubit rotation error and two qubits C-NOT error. Secondly, to understand the joint effect of all types of errors including the ones not studied in the first part (for example emerging from relaxation and decoherence), we import noise models (from the IBMQ platform) that truly emulate all noise influences present in the current quantum hardware. The concise summary of how to implement all of the mentioned noise models is presented in Figure 2. Lines 59 – 82 present the application of distinct custom noise models: 64 –72 show how to implement measurement/readout error and 74–82 how to obtain noise coming from gate errors (77 and 81 define the single qubit gate error and 78 and 82 define two qubit C-NOT error). Lines 53–57 describe how to use the IMBQ noise models.

In Section 2.1 we observed that utilizing the *Qiskit* plug-in inside *Pennylane* produces a slowdown in the timing of the learning process. To overcome this retardation of training we make use of the observation that VQC can obtain equally good results in the metric of the AUC classification score even when it is exposed to much smaller datasets (See Figure 3). Hence we perform all of our noise experiments with 120 signal datapoints and 120 background datapoints with batch size chosen to be equal to 24.

3.2 Deployed quantum neural network architecture

All of the noise model tests were run using the VQC architecture presented in Figure 4. We uses 8 features that we encoded in 4 qubits with the data re-uploading technique. As a variational form we chose the *2local form* with only nearest neighbours gates connections.

To further reduce the training time we used only two repetitions in each variational form which is less than was used previously in [1]. The shrinkage of the quantum neural network size was motivated by the fact that no changes in AUC score were observed after decreasing the quantum circuit size.

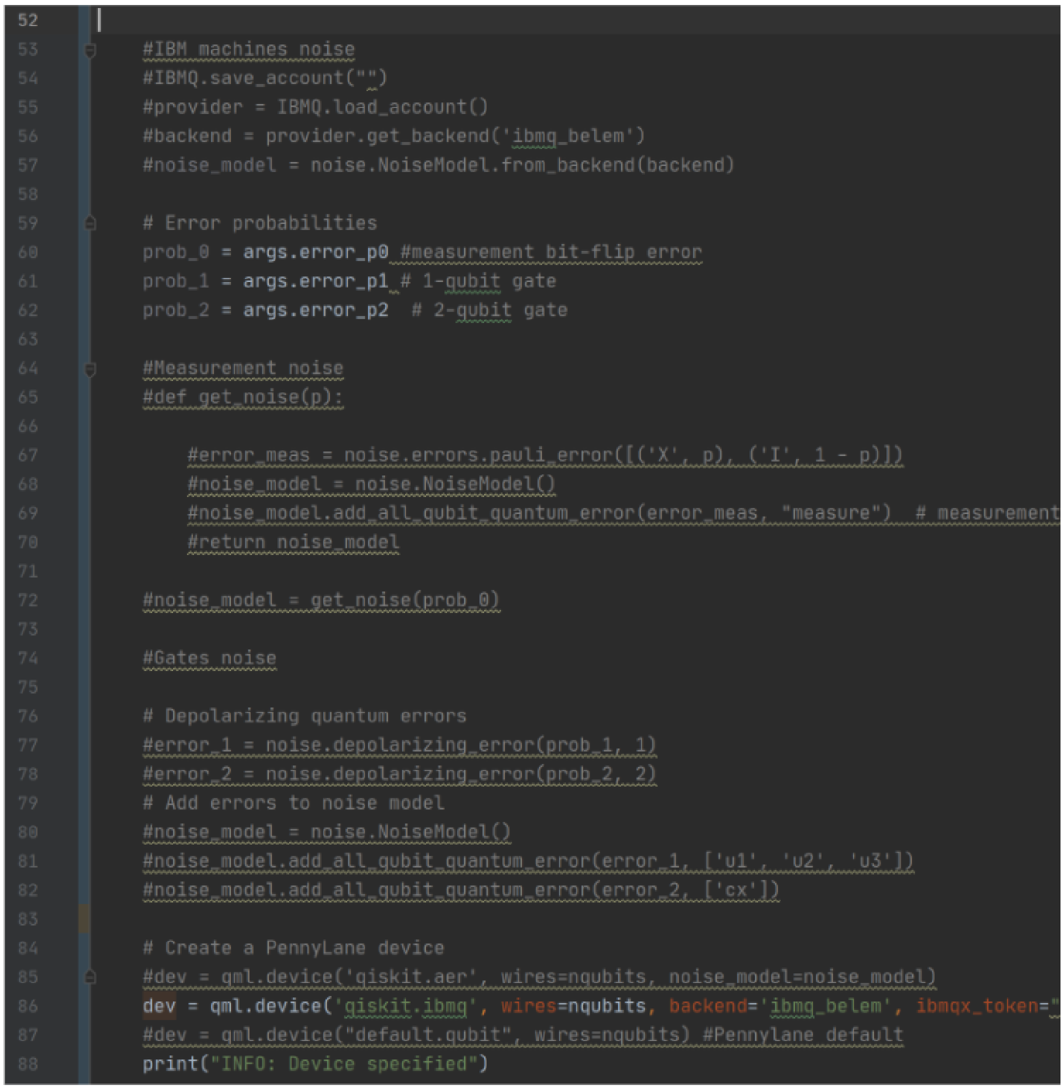
3.3 Simulation results

For each of the custom noise models presented in Section 3.1 we obtain a plot (Figures 5, 6, 7) presenting how the AUC score varies with respect to the noise level which is expressed in terms of the error probabilities introduced in Figure 2.

Based on the simulations results we can presume that the studied VQC quantum classifier architecture is resistant to each of the noise types (considered in isolation with respect to one another) up to the following levels:

$$p_0 = 0.35, \quad p_1 = 0.025, \quad p_2 = 0.025.$$

We also investigated the performance of the VQC classifier when exposed to noise models taken from IBMQ backends. We present the results in Table 1. We consistently observe unvarying results in terms of AUC score on the level when no noise would be present in the system.



```

    #IBM machines noise
    #IBMQ.save_account("")
    #provider = IBMQ.load_account()
    #backend = provider.get_backend('ibmq_belem')
    #noise_model = noise.NoiseModel.from_backend(backend)

    # Error probabilities
    prob_0 = args.error_p0 #measurement bit-flip error
    prob_1 = args.error_p1 # 1-qubit gate
    prob_2 = args.error_p2  # 2-qubit gate

    #Measurement noise
    #def get_noise(p):

        #error_meas = noise.errors.pauli_error([('X', p), ('I', 1 - p)])
        #noise_model = noise.NoiseModel()
        #noise_model.add_all_qubit_quantum_error(error_meas, "measure")  # measurement
        #return noise_model

    #noise_model = get_noise(prob_0)

    #Gates noise

    # Depolarizing quantum errors
    #error_1 = noise.depolarizing_error(prob_1, 1)
    #error_2 = noise.depolarizing_error(prob_2, 2)
    # Add errors to noise model
    #noise_model = noise.NoiseModel()
    #noise_model.add_all_qubit_quantum_error(error_1, ['u1', 'u2', 'u3'])
    #noise_model.add_all_qubit_quantum_error(error_2, ['cx'])

    # Create a PennyLane device
    #dev = qml.device('qiskit.aer', wires=nqubits, noise_model=noise_model)
    dev = qml.device('qiskit.ibmq', wires=nqubits, backend='ibmq_belem', ibmqx_token="
    #dev = qml.device("default.qubit", wires=nqubits) #Pennylane default
    print("INFO: Device specified")
```

Figure 2: Figure presents a piece of the code responsible for creating different noise models. *Qiskit* operates using the class *noise* which allows to create various noise models either from scratch ($noise.NoiseModel()$ is an empty noise model which does not affect the circuit in any way) or importing such from specified quantum hardware backends ($noise.NoiseModel.from_backend(backend)$). The noise model is then passed to the $qml.device()$ function – e.g. $dev = qml.device('qiskit.aer', wires = 2, noise\_ model = noisemodel)$.

3.4 Discussion

Studying different types of noise in isolation we observed that for each of them there is a threshold beyond which the AUC score starts to decrease. We estimated these threshold values to be:

$$p_0 = 0.35, \quad p_1 = 0.025, \quad p_2 = 0.025.$$

In the Table 2 we put forward the corresponding[4] error rates present in the current IBMQ hardware (as of October 2021). We observe that the thresholds obtained for the investigated VQC architecture are much bigger than the values encountered in the today's state of the hardware. That remark confirms the suitability of running QML models for HEP applications on quantum hardware.

[4]The exact definitions of error rates might differ from the IBM definitions.

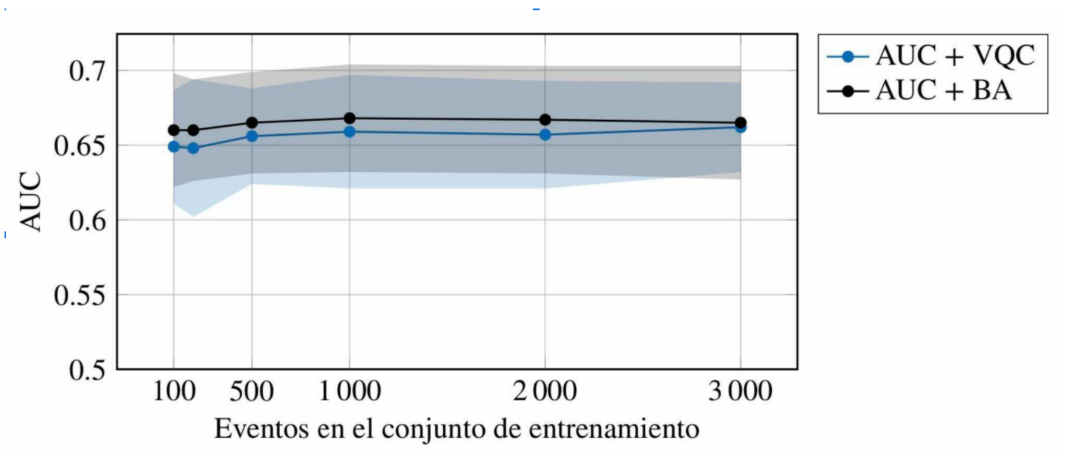
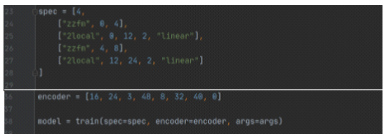


Figure 3: Plot shows that the AUC score of the VQC classifier is not sensitive to the size of the training set. The AUC score stays Credit to: Samuel Gonzalez Castillo. Figure comes from Samuel's thesis written in Spanish. "Eventos en el conjunto de entrenamiento" translates to "Number of datapoints in the training set".



```
spec = [4,
    ["zzfm", 0, 4],
    ["2local", 0, 12, 2, "linear"],
    ["zzfm", 4, 8],
    ["2local", 12, 24, 2, "linear"]
]

encoder = [16, 24, 3, 48, 8, 32, 40, 0]

model = train(spec=spec, encoder=encoder, args=args)
```

Figure 4: Figure presents the portion of the code in which the structure of the quantum neural network is specified. Code also indicates which 8 features from the initial dataset were chosen (see [1]).

Table 1: Table presents obtained AUC scores for the training of VQC model with the noise models taken from the IBMQ real hardware backends. We notice no influence of the noise on the AUC scores. The results of different runs also provide compatible results.

IBMQ noise model	**Run 1**	**Run 2**	**Run 3**	**Run 4**	**Run 5**
belem	0.6598 ± 0.0181	0.6508 ± 0.0183	0.6571 ± 0.0209	0.6582 ± 0.0186	0.6561 ± 0.0192
bogota	0.6590 ± 0.0181	0.6598 ± 0.0191	0.6608 ± 0.0205	cluster error	0.6576 ± 0.0169
lima	0.6574 ± 0.0179	0.6577 ± 0.0187	0.6582 ± 0.0194	0.6578 ± 0.0189	0.6551 ± 0.0175
manila	0.6592 ± 0.0198	0.6576 ± 0.0209	0.6515 ± 0.0188	0.6585 ± 0.0190	0.6586 ± 0.0197
quito	0.6558 ± 0.0218	0.6579 ± 0.0196	0.6567 ± 0.0178	0.6586 ± 0.0197	0.6567 ± 0.0208
santiago	0.6562 ± 0.0197	0.6580 ± 0.0188	0.6603 ± 0.0204	0.6602 ± 0.0181	0.6577 ± 0.0184

4 Conclusions

We have presented a study of quantum classifiers on the specific example of Variational Quantum Circuit. We proposed and followed a method to study how the noise, which is inherent to current quantum hardware, might affect the training of VQC architectures. We concluded, that for each type of the separately introduced error (measurement error, arbitrary one-gate error, C-NOT error), the considered models are resistant to noise even up to the levels that are are not encountered in the existing IBM hardware machines. To check the behaviour of the models when all of the investigated noise sources are present at the same time we tested the VQC models with IBM noise models accurately emulating the rates of errors

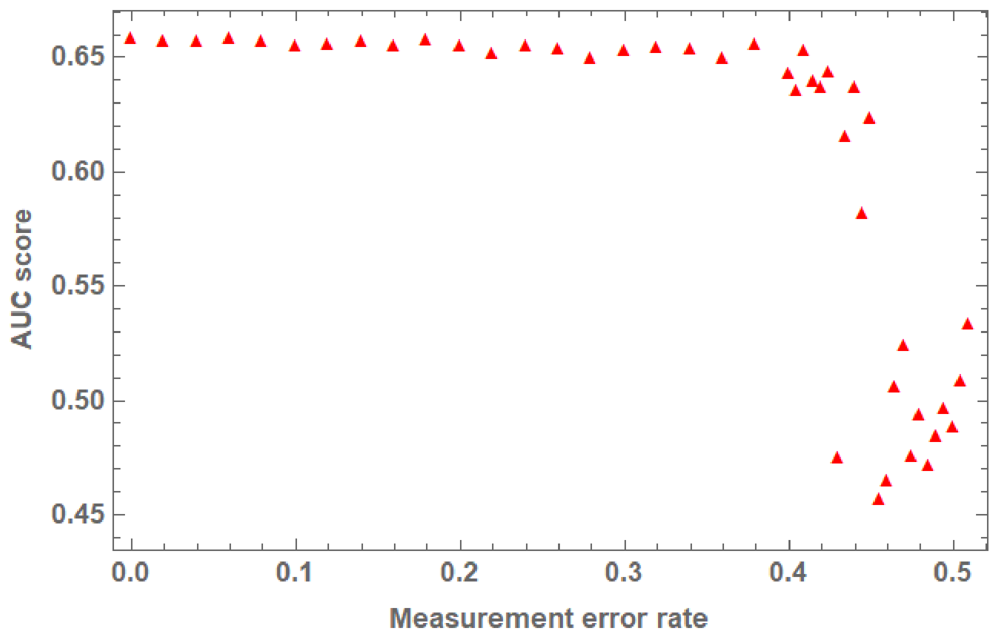


Figure 5: Figure presents how measurement/readout errors influences the AUC score of the VQC classifier. We observe that this QML model is resistant to the studied type of noise up to the level of $p_0 = 0.35$.

Table 2: Table shows various average (per qubit) error rates present in the IBMQ hardware machines (as of October 2021).

IBMQ hardware	p_0	p_1	p_2
sydney	0.04	0.0004	0.013
guadalupe	0.02	0.0003	0.012
casablanca	0.03	0.0003	0.011
manila	0.03	0.0003	0.008

present in current quantum hardware. We achieved a strong indication that QML algorithms being capable of solving demanding High Enrgy Physics data analysis tasks might be successfully operated from the level of the state of the art quantum hardware.

Acknowledgements

First of all, I would like to thank Dr Sofia Vallecorsa for giving me this one of a kind opportunity to work in a thriving CERN community on the research topics encapsulating the latest advancements in the fascinating field of Quantum Machine Learning. I would like to say huge thank you to my wonderful colleagues: Vasilis Belis, Patrick Odagiu and Samuel Gonzalez Castillo from the team from whom I obtained constant support and guidance, highly useful tips and help with the technical side of the work. Last but not least, thank you, Kristina Gunne, for being the kindest coordinator of the program ever. It was highly rewarding experience for me to be part of the summer school CERN openlab program – I am grateful to every person who made this journey possible and I heartily recommend this program to everyone.

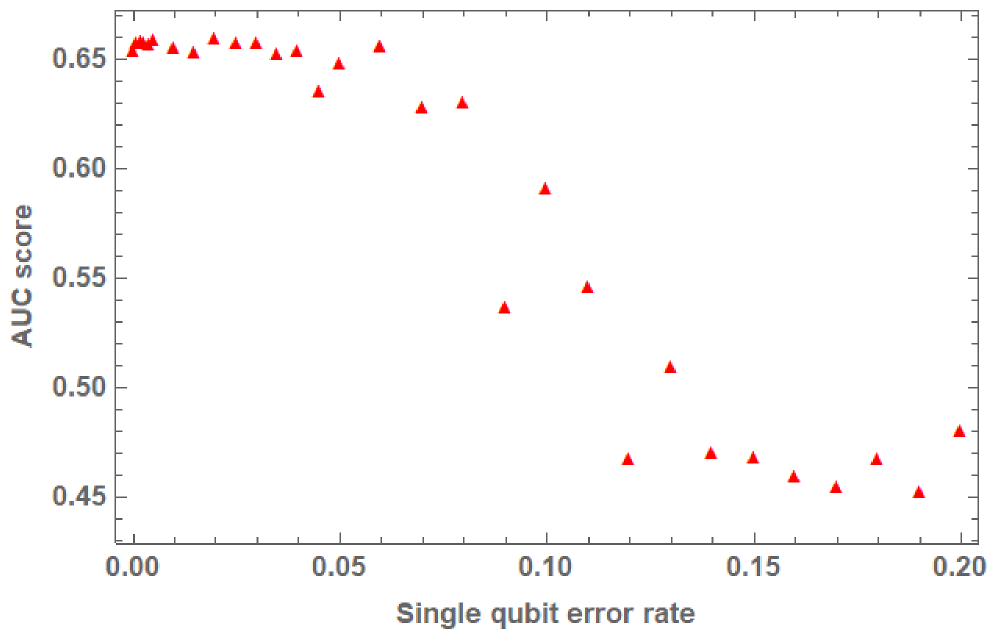


Figure 6: Figure presents how single qubit gate errors influences the AUC score of the VQC classifier. We observe that this QML model is resistant to the studied type of noise up to the level of $p_1 = 0.025$.

Bibliography

[1] V. Belis, S. Gonzalez-Castillo, C. Reissel, S. Vallecorsa, E. F. Combarro, G. Dissertori, and F. Reiter, *Higgs analysis with quantum classifiers*, EPJ Web of Conferences 251 (2021) 03070.

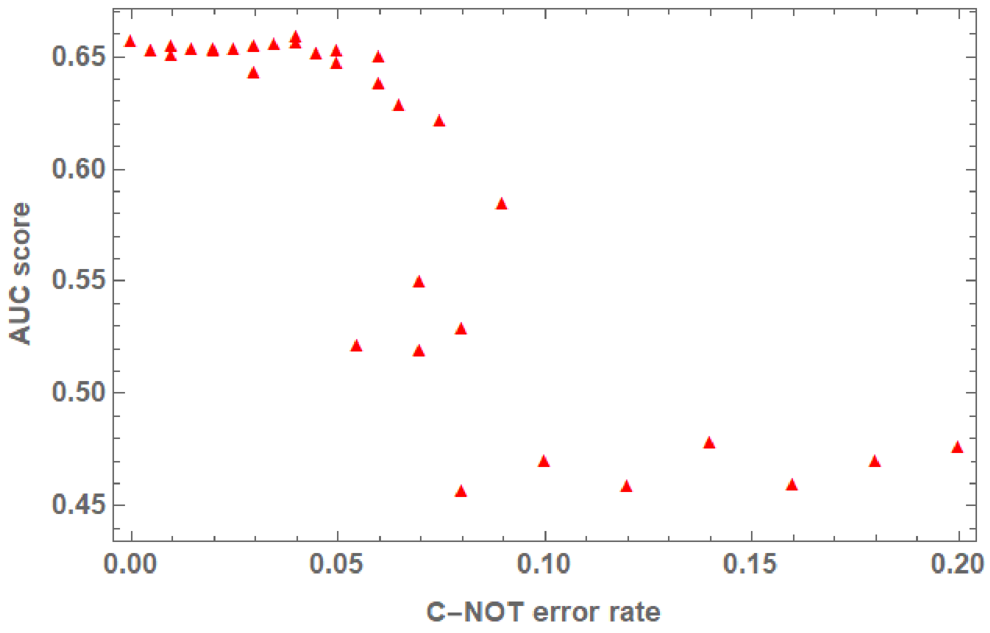


Figure 7: Figure presents how C-NOT errors influences the AUC score of the VQC classifier. We observe that this QML model is resistant to the studied type of noise up to the level of $p_2 = 0.025$.