Improving Noisy Hybrid Quantum Graph Neural Networks for Particle Decay Tree Reconstruction

Melvin Strobl[1,*], Eileen Kuehn[1], Max Fischer[1], and Achim Streit[1]

[1]Karlsruhe Institute of Technology

Abstract. With the emergence of the research field of Quantum Machine Learning, interest in finding advantageous real-world applications is growing as well. However, challenges concerning the number of available qubits on Noisy Intermediate Scale Quantum (NISQ) devices and accuracy losses due to hardware imperfections still remain and limit the applicability of such approaches in real-world scenarios. Therefore, for simplification, most studies assume nearly noise-free conditions as they are expected with logical, i.e. error-corrected, qubits instead of real qubits provided by hardware. However, the number of logical qubits is expected to scale slowly as they require a high number of real qubits for error correction. This is our motivation to deal with noise as an unavoidable, non-negligible problem on NISQ devices. As an application, we use the example of particle decay tree reconstruction as a highly complex combinatoric problem in High Energy Physics. We investigate methods to reduce the noise impact of such devices and propose a hybrid architecture that extends a classical graph neural network by a parameterized quantum circuit. While we have shown that such a hybrid architecture enables a reduction of the amount of trainable parameters compared to the fully classical case, we are now specifically interested in the actual performance in more realistic, i.e. noise prone scenarios. Using simple synthetic Decay Trees, we train the network in classical simulations to allow for efficient optimization of the parameters. The trained parameters are validated in noisy simulations based on devices by "IBM Quantum" and are used in interpretability and significance studies, enabling improvements in the accuracy on real devices.

1 Introduction

In recent years, there is a growing interest in the field of Quantum Machine Learning (QML) with application in High Energy Physics (HEP) [1–4]. As the size of publicly available quantum computers presumably will grow within the next few years [5], noise will remain a non-negotiable part without proper error correction. The large computational space that is accessible by even small quantum circuits, motivates application of QML in HEP [1]. The Full Event Interpretation (FEI) [6] as a highly non-trivial combinatorical problem is such an application and usually addressed using Boosted Decision Trees.

The authors in [7] introduced an end-to-end trainable solution based on the encoder of a Graph Neural Network (GNN) approach as proposed in [8] for predicting particle trajectories. Based only on the momenta of a set of particles, this architecture is able to reconstruct

*e-mail: melvin.strobl@kit.edu

the Lowest Common Ancestor Generation (LCA) matrix that contains information of the structural properties of the decay tree. In our previous work, we investigated the bene ts of a hybrid quantum-classical model and added a Quantum Multi Layer Perceptron (QMLP) layer as a rst part of this architecture for preprocessing [9]. We observed that with an equal number of parameters we are able to achieve better results with this hybrid approach instead of a purely classical approach.

In this work we evaluate di erent techniques to improve the performance of the hybrid quantum GNN whilst reducing the time of individual trainings:

1. A hyperparameter study to ne tune and evaluate the most important hyperparameters for both the classical and quantum part of the model;
2. Di erent optimizers for the classical and quantum part of the architecture;
3. Pruning of parameters to reduce the overall training time required for gradient calculation in the quantum part of the model.

2 Related Work

The authors in [10] introduced entangled dropout, where they randomly removed entanglement gates during the training process to mimic the behaviour of classical dropout, with the aim of improving the generalisation ability tested in small-scale experiments. It may have comparable e ects to our proposed pruning of parameters for gradient calculation. However, dropout generally disables the parameters completely, i.e. during the forward and backward pass during the training process. In contrast, we keep parameters active during the forward path and only disable them during the backward pass. Moreover, our approach does not rely on randomness and rather has the goal to reduce training time instead of improving the generalization capability.

In the Qiskit framework [11] that we used for the implementation of our approach, a transpilation process adapts a circuit such that it is executable on quantum hardware. During this process, it is ensured that all gates are either available or get translated into the available set of gates. In addition, gates that do not contribute to the algorithm, e.g. rotary gates with a parameter close to or equal to 0, are removed from the circuit. The general aim of this process is to reduce the total number of gates and thus the noise introduced.

Optimizing hyperparameters not only causes a lot of overhead, but can also have a huge impact on the outcome of training. Thus, they are rigorously studied in both classical Machine Learning (ML) [12] and QML [13]. However, the combination of both elds poses new challenges for trainability, as the importance of hyperparameters as well as the magnitudes of their values can di er. This can, for example, a ect the performance of the chosen optimizer. However, to the best of our knowledge, no speci c research is known that focuses on analyzing hyperparameters for hybrid models.

Solutions for hybrid quantum-classical architectures are presented for various use cases including HEP [2, 14, 15]. These solutions often build upon Parameterizable Quantum Circuits (PQCs) to allow for training the proposed ansätze by parameter tuning. However, current research focuses on using a single optimizer for the hybrid models. To the best of our knowledge, there is no previous work that focuses on having di erent optimizers for a single optimization task.

3 Approach

Our approach visualized in Figure 1 extends the previous work by [9] a preceding QMLP layer for preprocessing of the data. This quantum part takes the four-momenta of particles (L

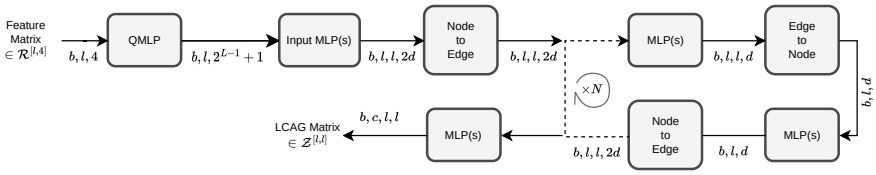

Figure 1. The QMLP takes an input of b batches of l particles (l in total) each described by its four-momenta. It is follows the architecture from [7] which is build around the encoder approach in [8]. This GNN is characterized by N iterations MLPs of size d surrounded by initial and final MLPs.

in total) in form of b batches and processes it and passes the result onto the a series of initial Multi Layer Perceptrons (MLPs) followed by a GNN of N iterations. A final set of MLPs then returns the LCAG matrix which describes the structural properties of the decay graph. Details on the exact approach can be found in [9].

This work focuses on the trainability and explores techniques such as different optimizers for quantum- and classical parts as well as parameter pruning to reduce overall training time. Furthermore we conduct hyperparameter optimization using Optuna to reveal insights on the importance of certain hyperparameters for the overall training performance.

3.1 Hyperparameter Optimization

Generally, hyperparameters are any non-trainable parameters of an architecture and its training process that have to be adjusted prior to the actual training. However, their configuration can have a significant impact on the training performance. Besides tuning hyperparameters manually, more structured methods such as grid-search or Tree-structured Parzen Estimator (TPE) [16] exist. While grid search ensures that all valid parameter configurations are tested, it can result in a reasonable overhead as it tests configurations that may be negligible in the first place. The TPE is an independent sampler that fits a Gaussian Mixture Model (GMM) to the hyperparameter configuration that produced the best objective value in each trial. On the one hand, this approach therefore reduces the overall number of evaluations as likely unfavorable hyperparameter configurations are not being considered any further. On the other hand it is not guaranteed that neglected configurations would not yield an even better objective value. As the LCAG value corresponds to our classes, the physical usability of the model's prediction is best reflected by the accuracy metric, which we therefore chose for our single-objective studies. Other metrics such as the perfect LCAG score would introduce unnecessarily hard constraints on the prediction as only a fully correct LCAG would improve this score.

3.2 Reducing the number of circuit evaluations

When training a PQC, gradients of the parameters are being calculated using the Parameter-Shift Rule (PSR) [17, 18]. This however is very costly in terms of training time as two (four) evaluations per rotational (controlled rotational gate) are required to obtain the gradient value of its parameter. During training, the gradients of the parameters usually decay when approaching a (global) optimum. As certain parameters have a less significant impact on the overall loss value, compared to others, this can happen already in early epochs of the training.

Ideally, if a parameter value remains unchanged for a couple of epochs, one might assume that it already approached a (globally) optimal value. However, with the output of a PQC being (at least) subject to statistical noise, the same noise is being reflected on the gradients and thus their values eventually will not become exactly zero.

In our approach, we therefore track the gradient variance for each parameter over a specified number of epochs. If the variance falls below a certain threshold, the gradient of a particular gate is not computed and the parameter value remains unchanged in that particular update step. As calculating gradients with the PSR is very expensive, this approach can significantly reduce the number of quantum circuits executed and thus the training time.

It shall be noted that skipping parameter updates due to small gradients is not necessarily bound to low parameter values and the approach therefore differs from optimization in transpilation processes [11]. If a parameter value approaches zero, the transpiler can remove this particular gate which would otherwise likely introduce noise while its contribution to the overall computation can be neglected. As we calculate the gradients using the PSR at least one/three of the two/four evaluations will result in a parameter value unequal zero. Thus the transpilation process will not be as effective as disabling the overall gradient calculation.

3.3 Separate Optimizers for Quantum and Classical parts

To further improve the trainability of our approach we consider carrying out separate optimizers with differently adjustable learning rates for the classical and quantum part of our model. This approach is mainly motivated by the different sensitivity and behaviour of parameter adjustments between Classical- and Quantum-Neural Networks (NNs). Different parameters requiring different learning rates is not a new concept in ML and optimizers such as Adam address this by introducing individual step sizes for each parameter. However, this optimizer is still sensitive to the initial learning rate. This could cause the optimization process to diverge when being setting the learning rate too high, or to slow down convergence rate when setting it too low.

4 Experiment Results

Using Optuna [21], we gave the TPE a set of hyperparameters and respective ranges as can be seen in Table 1. Amongst common parameters such as batch size, learning rate and NN specific parameters (number of blocks and -(additional) MLPs) we also evaluated different quantum circuit templates introduced in [20], data reuploading [19] and number of shots (i.e. number of measurements). Each study consists of 20 trials where multiple studies run concurrently. By using seeds, we guarantee that two independently run trials with the same hyperparameter configuration would yield the same objective value. The results are stored in a common database which, together with the reproducibility of our training, allows further estimation of a TPE using the results of other trials. We will refer to this set of trials as a single hyperparameter study. To reduce the time taken by each trial, we use a median pruner with 10 initial trials and 5 warm-up epochs each. This means that the pruner waits 5 epochs and then checks in each epoch whether the currently running trial should be pruned (i.e. cancelled) based on the current accuracy compared to the median of other trials.

From the results of this study we observed that in the best trial the hyperparameter optimizer selected Adam for both the quantum and classical part of the model but chose an order of magnitude higher learning rate for the quantum part. This effect is not only observed in case of the best trial but stays consistent amongst the trials with the best objective value. An importance analysis of the hyperparameter study, which visualises the importance of the hyperparameters on the change in the objective value of a trial, highlighted the batch size and learning rate (of both quantum and classical optimizers) as having a strong influence on the objective value of the optimization process. The hyperparameters of the best trial among these studies are used in following experiments. All simulations include the noise model of the IBM Quantum device ibm_perth.

Table 1. Hyperparameters considered in the hyperparameter study with their according range of values. The gradient curvature history and threshold are additional hyperparameters that we introduce to minimize the amount of required gradient calculations. The batch size, feedforward dimension and number of blocks are variables as used in the architecture described by Figure 1. The parameter con guration that is related to the trial with the highest accuracy is stated in the rightmost column.

Hyperparameter	Range	Optimal Value
Batch size (b)	f4; 8; 16; 32g	32
Data Reuploading [19] enabled	True, False	True
Dropout rate	f0:05 k j k 2 f1; : : : ; 10gg	0:1
Gradient Curvature history	f0; 4; 8g	4
Gradient Curvature threshold	f1e^{k} j k 2 f 12; : : : ; 9gg	1e^{11}
Feedforward dimension in MLPs (d	f16; 32; 64g	64
Number of blocks (N)	f1; 2; 3g	3
Number of MLPs within a block	f1; 2; 3; 4g	2
Number of additional MLPs	f1; 2; 3; 4g	2
Number of nal MLPs	f1; 2; 3; 4g	2
Prede ned PQC	"Circuit [16-19]" [20]	"Circuit 19"
Number of PQC layers	f1; 2; 3g	2
Number of shots	f64; 256; 1024g	1024
Learning rate of classical optimizer	4je^{k} k 2 f 4; : : : ; 1gg	4e^{3}
Learning rate decay of classical optimizer	f1e^{2} ; 1e^{1} ; 1g	0:1
Learning rate of quantum optimizer	4je^{k} k 2 f 4; : : : ; 1gg	4e^{4}
Learning rate decay of quantum optimizer	f1e^{2} ; 1e^{1} ; 1g	0:5
Type of the quantum optimizer	Adam, SGD	Adam

Figure 2 on the left shows the absolute gradient values of the individual parameters within the QMLP over a whole training process of 20 epochs. We use these gradient values to calculate variances which we use to decide whether to calculate the gradient of a parameter in subsequent epochs. The parameter indices are counted from left to right and from top to bottom in the circuit representation. One can see that e.g. parameter 4 does not receive any updates after epoch 9. Any subsequent calculation of its gradient can therefore be considered obsolete and would yield very small gradient values that are negotiable in the presence of noise. Here, noise refers to both, stochastic uctuations from measurements as well as derivation from the theoretical true value due to hardware imperfections.

Our approach introduces two hyperparameters that are required to determine whether the subsequent gradient calculation of a parameter should be skipped or not:

1. Gradient curvature history which de nes the number of epochs that are being taken into account when calculating the variance of the gradient values. Note that this also sets the number of epochs at the beginning of a training during which no pruning will happen.

2. Gradient curvature threshold which sets the limit below which subsequent parameter updates are being disabled.

Note that when gradient calculation is skipped, the parameter still contributes to the overall calculation which is not the case with the commonly used dropout technique as used in [10].

Figure 2 on the right shows the gradient values with pruning enabled. Here, the threshold was set to 1e^{11} and the curvature was calculated based on 4 epochs. Those values also originate from the previous hyperparameter study.

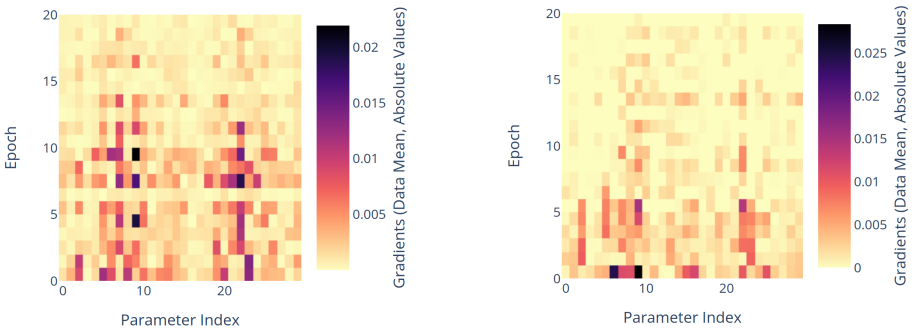

Figure 2. Absolute gradient values of the parameters within the QMLP over training epochs. Parameter indices are counted from left to right and from first to last qubit. The gradients are averaged over the batch size. The Figure on the left shows training without pruning and reaches an accuracy of 89% of 80 for 7.3h of training. The Figure on the right shows gradients with pruning enabled and reaches an accuracy of 80.4 for 3.4h of training. Curvature of gradients has been calculated based on 4 epochs with a threshold of $1e^{-11}$.

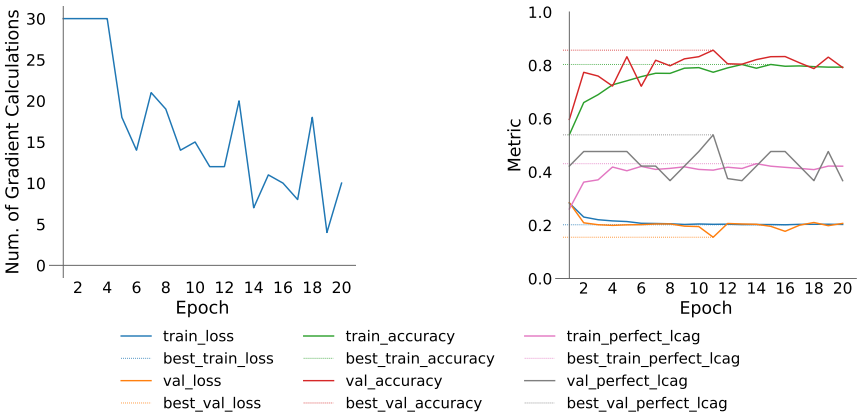

Figure 3. The number of selected parameters for gradient evaluation is visualized on the left. The Figure on the right shows the overall performance, both on training and validation data, of the hybrid architecture for various metrics. Besides the accuracy and loss we also evaluated the perfect LCAG score that was introduced in [7]. The peak performance is achieved in epoch 11.

In Figure 3 on the left we show the number of actually selected parameters of which the gradient is calculated in the backpropagation path. Note that after initially dropping the number of selected parameters in epoch 4, the number of parameters can increase if setting the gradient value to zero (which happens if a parameter is deselected) would cause the variance of its gradient to increase above the preset threshold. The reduced set of parameters reduced the training by 54% while the accuracy decreased only by 0.5%.

The overall training performance is shown in Figure 3 on the right where losses, accuracy and perfect LCAG score of both training and validation are visualized. Perfect LCAG here describes an exact match of the ground truth and the predicted LCAG.

5 Discussion & Conclusion

Based on the proposed methods, the results of this paper hint further interesting research directions such as 1. Separate optimizers for quantum and classical parts in hybrid 2. Pruning of parameter updates that become irrelevant during the training process due to vanishing gradients However, the signi cance of these approaches should be investigated in dedicated experiments with a reduced set of variables. Especially in case of separate optimizers, the capability of the classical NN is too high in the given architecture to allow for nal conclusions regarding this approach. This goes along with the impact of the QMLP to the overall capability of the model. In its current setup it is hard to measure and might be relatively small. Therefore, changes in the architecture should be made in further work on the task of FEI.

Skipping the gradient updates for certain parameters turned out to be a very e ective approach as it signi cantly reduced the overall training time. The e ect vanishing gradients due to the Barren Plateau [22] e ect is unlikely as the size of our circuit (4 qubits, 2 QC layers) does not fall into the scale discussed in the mentioned paper. However, more advanced pruning techniques and small scale experiments should be made to validate this approach. It should also be compared with a purely random dropout applied on the gradient calculation to see if the decision based on the gradients variance is actually bene cial. Another feasible addition might be the random activation of previously deactivated parameter updates which would correspond to an inverse dropout.

By using these approaches we are not only able to reduce the training time signi cantly by 54% with negligible decrease in accuracy but also improve on the results of our previous work in regards of the noise prone simulations.

References

[1] W. Guan, G. Perdue, A. Pesah, M. Schuld, K. Terashi, S. Vallecorsa, J.R. Vlimant (2020), 2005.08582

[2] C. Tüysüz, C. Rieger, K. Novotny, B. Demirköz, D. Dobos, K. Potamianos, S. Vallecorsa, J.R. Vlimant, R. Forster, Tech. Rep. University, (2021)

[3] A. Di Meglio, M. Doser, B. Frisch, D. Grabowska, M. Pierini, S. Vallecorsa, Tech. rep. (2022), https://zenodo.org/record/5846455

[4] A. Di Meglio, K. Jansen, I. Tavernelli, C. Alexandrou, S. Arunachalam, C.W. Bauer, K. Borras, S. Carrazza, A. Crippa, V. Croft et al., Tech. rep. (2023), arXiv:2307.03236 [hep-ex, physics:hep-lat, physics:hep-th, physics:quant-ph] type: article, http://arxiv.org/abs/2307.03236

[5] J.M. Gambetta, IBM Quantum roadmap to build quantum-centric supercomputers (2021), https://research.ibm.com/blog/ibm-quantum-roadmap-2025

[6] T. Keck, F. Abudinén, F.U. Bernlochner, R. Cheaib, S. Cunli e, M. Feindt, T. Ferber, M. Gelb, J. Gemmler, P. Goldenzweig et al., Computing and Software for Big Science 3, 6 (2019)

[7] J. Kahn, I. Tsaklidis, O. Taubert, L. Reuter, G. Dujany, T. Boeckh, A. Thaller, P. Goldenzweig, F. Bernlochner, A. Streit et al., Machine Learning: Science and Technology 3, 035012 (2022)

[8] T. Kipf, E. Fetaya, K.C. Wang, M. Welling, R. Zemel, Neural Relational Inference for Interacting Systems (2018), 1802.04687

[9] M. Strobl, E. Kuehn, M. Fischer, A. Streit, Journal of Physics: Conference Series (in review)

[10] M. Kobayashi, K. Nakaji, N. Yamamoto, Quantum Machine Intelligence **4**, 30 (2022), arXiv:2205.11446 [quant-ph]
[11] A.N.I.S. MD SAJID, Abby-Mitchell, H. Abraham, AduO ei, R. Agarwal, G. Agliardi, M. Aharoni, V. Ajith, I.Y. Akhalwaya, G. Aleksandrowicz et al., Qiskit: An open-source framework for quantum computing (2021)
[12] M. Claesen, B. De Moor, Tech. rep. (2015), arXiv:1502.02127 [cs, stat] type: article, http://arxiv.org/abs/1502.02127
[13] C. Moussa, J.N. van Rijn, T. Bäck, V. Dunjko, in Discovery Science, edited by P. Pascal, D. Ienco (Springer Nature Switzerland, Cham, 2022), Vol. 13601, pp. 32–46, ISBN 978-3-031-18839-8 978-3-031-18840-4, series Title: Lecture Notes in Computer Science, https://link.springer.com/10.1007/978-3-031-18840-4_3
[14] P. Jain, A.G. Garcia, Tech. rep. (2023), arXiv:2212.04209 [quant-ph, q- n] type: article, http://arxiv.org/abs/2212.04209
[15] A. Matic, M. Monnet, J.M. Lorenz, B. Schachtner, T. Messerer, Tech. rep. (2022), arXiv:2204.12390 [quant-ph] type: article, http://arxiv.org/abs/2204.12390
[16] J. Bergstra, R. Bardenet, Y. Bengio, B. Kégl, Algorithms for Hyper-Parameter Optimization, in Advances in Neural Information Processing Systems, edited by J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, K. Weinberger (Curran Associates, Inc., 2011), Vol. 24, https://proceedings.neurips.cc/paper_files/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf
[17] K. Mitarai, M. Negoro, M. Kitagawa, K. Fujii, Physical Review A **98**, 032309 (2018), arXiv:1803.00745 [quant-ph]
[18] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, N. Killoran, Physical Review A **99**, 032331 (2019), arXiv:1811.11184 [quant-ph]
[19] M. Schuld, R. Sweke, J.J. Meyer, Physical Review A **103**, 032430 (2021), arXiv:2008.08605 [quant-ph, stat]
[20] S. Sim, P.D. Johnson, A. Aspuru-Guzik, Advanced Quantum Technologies **2**, 1900070 (2019)
[21] T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama, Optuna: A Next-generation Hyperparameter Optimization Framework, in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2019)
[22] J.R. McClean, S. Boixo, V.N. Smelyanskiy, R. Babbush, H. Neven, Nature Communications **9**, 4812 (2018)