Quantum Science and Technology

PAPER

Precise image generation on current noisy quantum computing devices

Florian Rehm[1,2,*] , Sofia Vallecorsa[1] , Kerstin Borras[2,3] , Dirk Krücker[3] , Michele Grossi[1] and Valle Varo[3]

1   CERN, Geneva, Switzerland
2   RWTH Aachen University, Aachen, Germany
3   DESY, Hamburg, Germany
*   Author to whom any correspondence should be addressed.

E-mail: florian.matthias.rehm@cern.ch

Abstract

The quantum angle generator (QAG) is a new full quantum machine learning model designed to generate accurate images on current noise intermediate scale quantum devices. Variational quantum circuits form the core of the QAG model, and various circuit architectures are evaluated. In combination with the so-called MERA-upsampling architecture, the QAG model achieves excellent results, which are analyzed and evaluated in detail. To our knowledge, this is the first time that a quantum model has achieved such accurate results. To explore the robustness of the model to noise, an extensive quantum noise study is performed. In this paper, it is demonstrated that the model trained on a physical quantum device learns the noise characteristics of the hardware and generates outstanding results. It is verified that even a quantum hardware machine calibration change during training of up to 8% can be well tolerated. For demonstration, the model is employed in indispensable simulations in high energy physics required to measure particle energies and, ultimately, to discover unknown particles at the large Hadron Collider at CERN.

1. Introduction

Quantum computing has the potential for a new paradigm in future computing to accelerate tasks or even handle classically unsolvable problems [1]. In the current noise intermediate scale quantum (NISQ) era, quantum devices suffer from non-negligible hardware errors, limited connectivity and a limited number of qubits [2]. While practical quantum advantage is currently extremely difficult to accomplish, finding the best suited algorithms to effectively combat the problems of NISQ-era devices remains a widely researched topic. Quantum machine learning (QML) is a domain which achieves acceptable results on NISQ devices due to the observed robustness against noise [3].

High Energy Physics (HEP) experiments, such as those at the Large Hadron Collider (LHC) at CERN require enormous amounts of simulated data for deriving high precision physics results [4]. To handle this demand, gigantic quantities of computing hardware resources are necessary, which has led to the creation of the world's largest computing grid operated by CERN [5]. To alleviate this strain on computational resources, machine learning (ML) models have been developed that exhibit remarkable speed-ups over current Monte Carlo-based simulations while maintaining the required level of accuracy. In general, QML simulations represent a promising approach to address the only further increasing simulation demands in the future [6, 7]. QML employs quantum circuits which exploit the quantum properties of superposition and entanglement, which possess the potential to outperform neural networks, their classical analogue [8]. In addition, QML might have the advantage of learning more complex distributions with fewer parameters than classical ML due to their wider accessible phase space.

Encoding the classical data into qubit states on quantum computers is a non-trivial task [9]. Currently, many encoding techniques exist, each exhibiting specific advantages and disadvantages, and in practice,

identifying 'the best' encoding technique remains application dependent [9]. To achieve a potential quantum advantage over classical computing, theoretical studies suggest that at least linear scaling from qubits to features is required [10]. On the other side, models which employ better than linear scaling encoding techniques have drawbacks, making them unsuitable for generating precise images on NISQ devices. For example, amplitude encoding can only generate probability distributions and not absolute pixel entries, i.e. energy values.

At present, there exist several quantum generative models, for example, the quantum circuit born machine (QCBM) [11], quantum variational autoencoders [12] or variations of quantum generative adversarial networks [13–15]. They all face limitations. Some models either do not scale well in terms of qubits required relative to the number of encoded features, or they do not achieve a satisfying level of fidelity. The quantum angle generator (QAG) presented in this paper and first introduced in [16], aims to overcome these problems. Employing angle encoding with linear scaling of qubits to features, it achieves extremely accurate results for a real-world problem on current physical noisy quantum devices.

The paper content is structured as follows. First, the HEP use case is motivated and the training data is defined. Next, the QAG model and the employed angle encoding technique are presented. Then, multiple circuit architectures are compared and the advantages of the best ones are highlighted. An in-depth accuracy analysis of the model follows to highlight its excellent precision. The quantum hardware noise behavior is evaluated, including training and inference executed on real quantum devices. Lastly, conclusions are drawn and summarized.

2. High energy physics simulations

Simulations remain a crucial component of HEP analysis to evaluate the results obtained by the processed data of the experiments. Currently, simulations are performed predominantly with Monte Carlo methods such as the Geant4 toolkit [17]. However, Monte Carlo simulations are very hardware resource demanding and occupy half of the worldwide LHC Computing Grid [18]. Future LHC experiments will require more simulations due to more energetic particles, simultaneous collisions and detectors constructed with higher granularity. However, the projected budget for hardware development and computing resources cannot keep pace with these increasing demands [19, 20]. As a result, ML alternatives to Monte Carlo methods are being actively researched. Initial prototypes predict significant reductions in simulation time and hardware resources while retaining acceptable levels of accuracy [21–23]. This research goes one step beyond classical ML. Since HEP data sets are generally created by underlying quantum mechanical effects, performing the computations on quantum devices which likewise make use of quantum effects has the potential to substantially enhance the simulations in accuracy and in terms of sustainable computing.
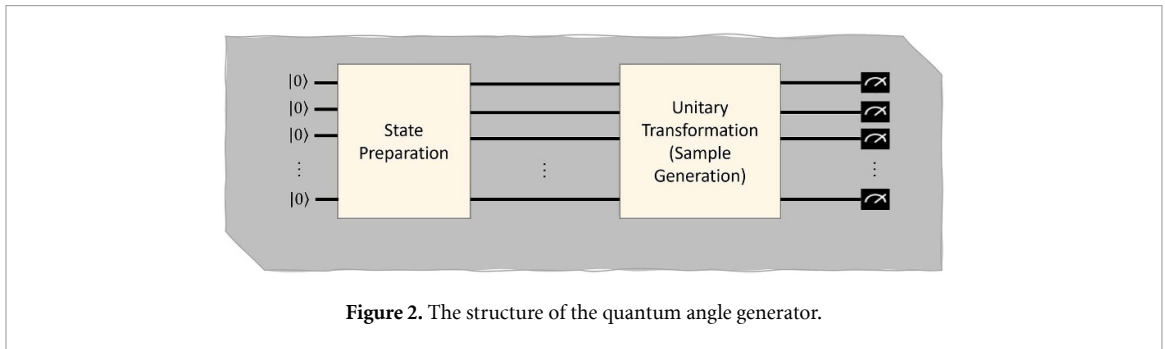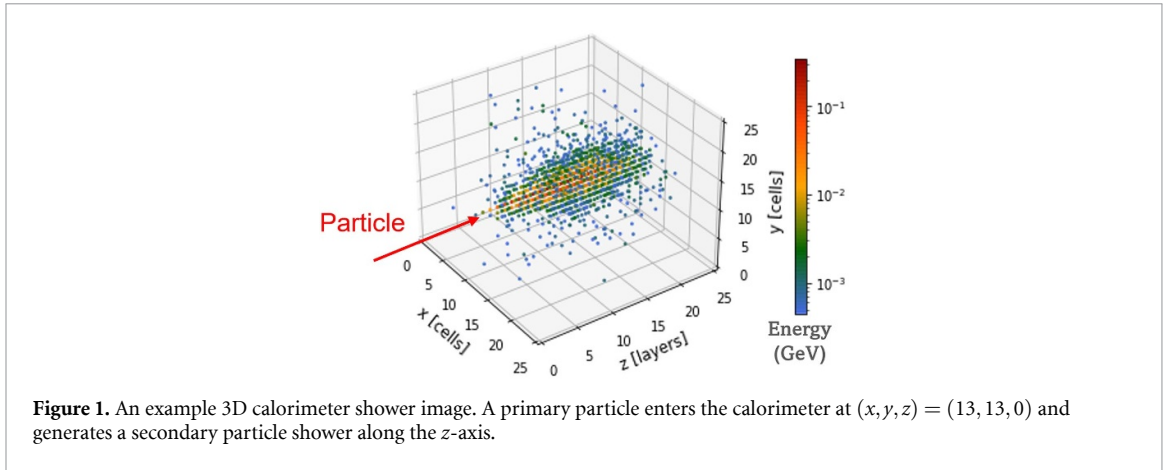
Electromagnetic calorimeters are constructed as high granularity sensor grids to measure the energy of photons, electrons, and positrons through complex particle shower generation processes in space and time [24]. They constitute a key component of HEP detectors to measure the energy of the particles produced in the interaction process and occupy most of the simulation time [18]. Calorimeter outputs can be interpreted at lowest order as static and spatial 3D images, which we call 'shower images': the value of each pixel corresponds to the energy measured in a specific calorimeter cell. The initial data from [25] consists of $25 \times 25 \times 25$ pixel images. An example of a 3D shower image is visualized in figure 1. To reduce the dimensionality, the images are averaged along two spatial axes (x- and y-direction), resulting in a one-dimensional representation that is further downsampled to eight pixels by averaging three contiguous pixels along the z-direction. Although the initial data set provides many different energies, for simplicity this study focuses on images recorded by particles in the energy range of [225, 275] GeV. The data set is split into a training and test set, each consisting of approximately 1000 samples. The downsampled data is available in [26], and an example image is illustrated later in this paper in figure 6(a).

3. The QAG

The QAG represents a QML model which employs the well established technique of angle encoding [27, 28] to generate extremely precise images. It scales linearly with the number of encoded features. Thus, the generation of n features requires n qubits. In this study, the number of features corresponds to the number of pixels. A comprehensive description of the QAG model, its objective training function, and an evaluation of several quantum circuits are provided below.

3.1. Model description
The QAG model consists of variational quantum circuits trained by an objective function. The model structure is visualized in figure 2. All qubits are initialized in the basis state $|0\rangle$. The state preparation

Figure 1. An example 3D calorimeter shower image. A primary particle enters the calorimeter at $(x, y, z) = (13, 13, 0)$ and generates a secondary particle shower along the z-axis.

Figure 2. The structure of the quantum angle generator.

function implements a Hadamard (H) gate to constitute superposition, followed by a y-rotational (Ry) gate to introduce randomness so that the model can draw new samples at each execution. For this, the Ry gate angles Ω are randomly drawn from a $[-1, 1]$ uniform distribution and pixel-wise multiplied by the pixel standard deviations present in the training data to obtain correct pixel energy variations. To account for all various primary particle energies, all angles Ω are multiplied by a random value between $[-0.25, 0.25]$.

The unitary transformation consists of quantum circuits and constitutes the trainable part of the QAG model. Various circuit architectures were tested as documented in section 3.3.

To convert the quantum states back into classical energy values via angle encoding, the model must be executed multiple times and the quantum states measured. The number of executions is commonly denoted as the number of shots nb_shots. It is counted how often state $|0\rangle$ is measured. The scalar intersection I of the vertical axis on the Bloch sphere (z-axis) and the angle θ is calculated with:

$$I = 2 \cdot \frac{\text{counts}\,(|0\rangle)}{nb_\text{shots}} - 1,$$
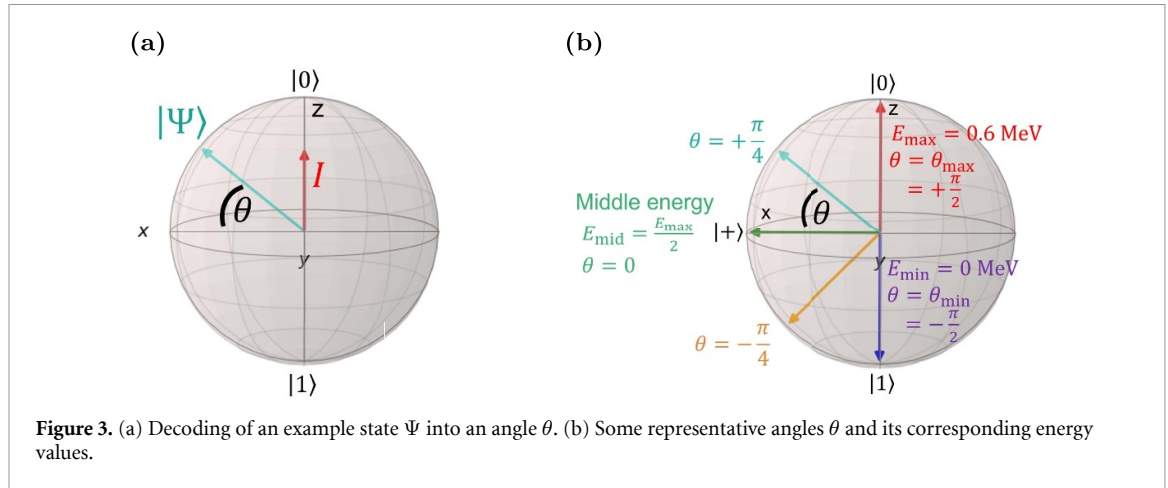$$\theta = \arcsin\,(I)\,.$$

(1)

The angle θ operates in the x-z-plane of the Bloch sphere and is defined as zero in the $|+\rangle$ state. Rotating θ clockwise leads to positive angles. The decoding process is visually illustrated in figure 3(a) with an example state $|\Psi\rangle$, its intersection I and the corresponding angle θ for a single qubit example. The angle θ can then be transformed into a pixel energy E by the linear change of ranges equation:

$$E = \frac{(E_\text{max} - E_\text{min})\,(\theta - \theta_\text{min})}{\theta_\text{max} - \theta_\text{min}}\,.$$

(2)

E_max, E_min, θ_max and θ_min are defined in figure 3(b): the minimum energy $E_\text{min} = 0\,\text{MeV}$ is set to $\theta = -\pi/2$ and the maximum energy $E_\text{max} = 0.6\,\text{MeV}$ is set to $\theta = +\pi/2$. With $E_\text{min} = 0$ and $\theta_\text{max} = -\theta_\text{min}$ equation (2) can be simplified into:

$$E = \left(\frac{E_\text{max}}{2 \cdot \theta_\text{max}}\right) \cdot (\theta + \theta_\text{max})\,.$$

(3)

For multi-qubit quantum circuits: the final quantum states of all qubits are measured, and the outcomes are decoded independently. Although the qubit results are individually decoded, the state of each qubit is entangled with others due to the gates applied within the variational quantum circuit, as lined out later.

Figure 3. (a) Decoding of an example state Ψ into an angle θ. (b) Some representative angles θ and its corresponding energy values.

It is worth to be noted that the angle θ and, therefore, the corresponding decoded energy E remain in discrete values. Since, θ depends in value and accuracy on the number of shots nb_{shots}: the larger nb_{shots}, the better the achievable energy precision and resolution. Fortunately, with present quantum devices, the nb_{shots} can be easily chosen to be large. Currently, on IBMQ devices the maximum possible number of shots is $nb_{shots} = 100\,000$. For the simplified calorimeter use case, this is more than sufficient. For comparison, in a previous classical reduced precision ML research in [29], it is demonstrated that already 256 discrete energy levels are sufficient for correctly reproducing the full-size calorimeter shower image. In this reduced precision research, the parameters of the neural network are quantized from a larger format (floating point 32) down to a smaller number format (integer 8). This study will show that 512 shots provide a sufficient resolution. The detailed image generation process for the QAG model, as described in this subsection, is provided as an algorithm in the appendix A.

3.2. Training objective function

The QAG model is trained with two losses employed as objective functions. The first one is the mean maximum discrepancy (MMD) loss [30, 31], already successfully applied by other quantum models, e.g. the QCBM [11]. Training exclusively with the MMD loss resulted in good average shower distributions. However, when exploring the generated images in more detail, for example in the pixel correlation, the model did not perform satisfactorily. Therefore, a second correlation (Corr) loss is added to help learn the patterns present in the training data (e.g. image pixel correlations). The Corr loss is calculated by the pixel-wise mean squared error (MSE) between the pixel correlation values present in the training data and the ones inside the generated data. The pixel-wise correlations are illustrated in figure 6(b).

To train the QAG model, the simultaneous perturbation stochastic approximation (SPSA) optimizer [32] is employed, which only requires two optimization steps per epoch. The hyperparameters for training were found by extensive hyperparameter searches employing the Optuna [33] library. All tests in this study are executed in Qiskit version 0.26.2. The models are trained for 500 epochs, containing one batch. The dynamic MMD loss weight starts at a value of one and decays linearly with $-0.001 \cdot$ epoch, starting from epoch 100. Opposite, the Corr loss weight increases by the same value starting at zero. The dynamical training batch size is set to generate one image in the first hundred epochs and afterward to 20 images to calculate the Corr loss between multiple images. Each quantum job contains 512 shots for training and inference. The generator SPSA optimizer learning rate is set to $c_0 = 1$ with an exponential learning rate decay of 0.006 starting from epoch 50. All these settings showed the best performance in the tests.

3.3. Quantum circuit study

The ideal circuit should contain a certain, optimized to be minimal, number of parameters to achieve a sufficient level of accuracy. Different circuit architectures were compared to each other based on characteristic numbers expressing the power of the circuit. The circuits employ trainable rotational gates and two-qubit entanglement gates. As the angle encoding primarily uses the qubits y-axis component, we predominantly employ Ry gates. For some circuit architectures, we test if additional z-rotational gates (Rz gates) or deeper circuits with depth 2 (denoted as d2) can further improve the results. We use two-qubit controlled-not gates (cx gates) native on IBM Quantum (IBMQ) [34] devices; while other entanglement gates are compositions of multiple native gates of the hardware. Keeping an eye on the goal of executing the
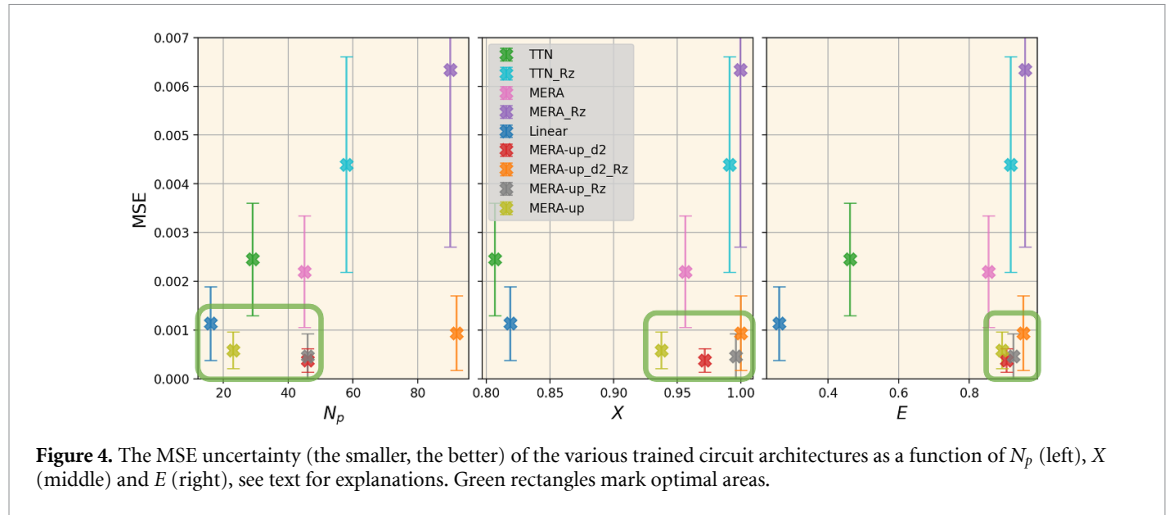
Figure 4. The MSE uncertainty (the smaller, the better) of the various trained circuit architectures as a function of N_p (left), X (middle) and E (right), see text for explanations. Green rectangles mark optimal areas.

training on a real quantum device, the absolute number of decomposed gates should be kept as small as possible.

The characteristic circuit numbers used in this study are the number of trainable parameters N_p, the expressibility X and the entanglement capability E. Larger circuits with more trainable parameters are potentially capable of achieving more accurate results. However, it might be that a plateau is reached at some point where the same task can be solved with similar accuracy by a smaller circuit, which is being investigated here. The definitions for X and E are from [35]: X describes how well the circuit can represent the pure states of the representative Hilbert space. For a single qubit the expressibility exhibits how many states of the Bloch sphere can be represented. In this paper, we measure $1 - X$: the closer to 1 the better the expressibility of the model, while the closer to 0 the worse. The entanglement capability E is a measure that expresses the ability of a circuit to generate entangled states between the qubits. Likewise, E ranges from 0 to 1, where 1 represents the best achievable value. The circuit architectures under study are introduced in the appendix B. Their corresponding characteristic numbers and theoretical potential are provided in the appendix C. In the following, the circuits are evaluated for the calorimeter use case.

We start by interpreting the results displayed in figure 4. The MSE accuracy metric is calculated by taking the pixel-wise MSE between the average Geant4 and QAG images. The training is repeated 25 times for each circuit and the mean and standard deviations are plotted. To prevent the influence of outliers, the best and worst two trials are discarded in this analysis. The MSE is given as a function of: N_p on the left, X in the middle, and E on the right. By inspecting the plots, it can be recognized that the MSE does not correlate with any of the characteristic circuit values in the plots, neither do the characteristic values correlate among themselves, as shown in the appendix C.

The MERA-up, MERA-up_d2, and MERA-up_Rz architecture perform best with the lowest MSE. This is consistent with the observation that they maintain a high X and E, as provided in the appendix B. The error bars provide a hint about the training stability. It can be observed that the better the average MSE of a model, the smaller its standard deviation.

All in all, the MERA-up_Rz circuit clearly performs best considering the characteristic circuit values and the achieved accuracy in training. However, with the emphasis on a low number of N_p, the plain MERA-up circuit performs almost as well, while needing only half the number of parameters. Therefore, the following studies employ the MERA-up circuit for training the QAG model.

4. In-depth accuracy analysis

In this section, we analyze the results of the QAG model operating the MERA-up circuit architecture. We showcase typical accuracy metrics for the calorimeter simulation in HEP.

4.1. Training evaluation

In a first step, the statistical trends of the objective functions during training are investigated. In figure 5, the unweighted loss functions (excluding loss weights) are plotted as a function of the training epochs. The mean of twenty training repetitions is visualized as a thick solid line and the standard deviation (STD) as a colored band. The Corr loss starts influencing the training only at epoch 100 because its weight is set to zero before.

The MMD loss of all training repetitions converges stable, and the STD band narrows towards the end of the training. Overall, the MMD and Corr loss converge smoothly without strong oscillations, which is a
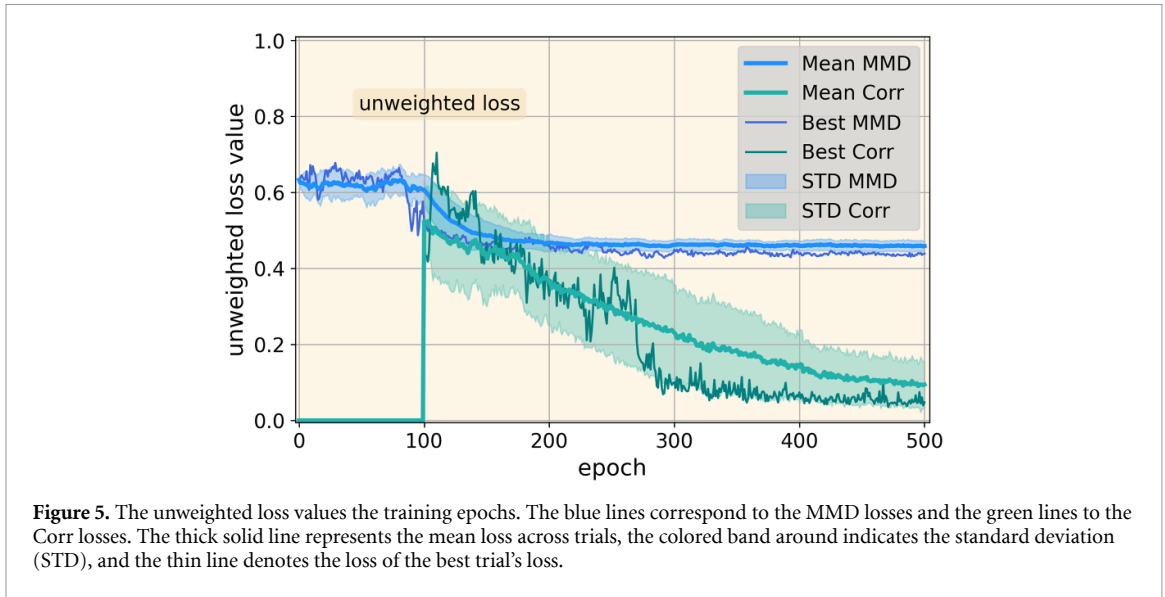
Figure 5. The unweighted loss values the training epochs. The blue lines correspond to the MMD losses and the green lines to the Corr losses. The thick solid line represents the mean loss across trials, the colored band around indicates the standard deviation (STD), and the thin line denotes the loss of the best trial's loss.
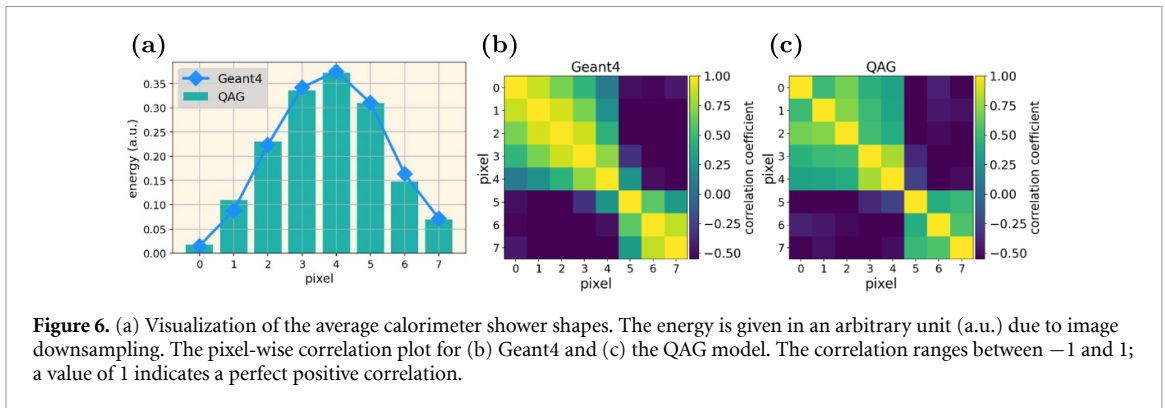
Figure 6. (a) Visualization of the average calorimeter shower shapes. The energy is given in an arbitrary unit (a.u.) due to image downsampling. The pixel-wise correlation plot for (b) Geant4 and (c) the QAG model. The correlation ranges between -1 and 1; a value of 1 indicates a perfect positive correlation.

desirable characteristic for stable (Q) ML training. The MMD loss contributes far more than the Corr loss throughout the training. However, the Corr loss plays a significant role in achieving good physics accuracy in the generated shower images.

4.2. Inference evaluation

In the following, the accuracy in inference is evaluated. The generated images of the best trained model are compared to the Geant4 test data, which consists of 980 images. Likewise, there are 980 images generated by the QAG model to create the following accuracy metrics. The details about how the accuracy metrics are calculated are provided in the appendix D.

    1. Average calorimeter shower shape: the first metric represents the calorimeter shower shape displayed in figure 6(a). The shower shape is perfectly reproduced by the QAG model. The MSE corresponds to $0.000\,59 \pm 0.000\,37$, which is extremely close to zero, indicating a very good accuracy.

    2. Pixel-wise correlation: the second metric corresponds to the pixel-wise image correlation. The positive or negative correlation patterns between all the pixels are determined. The baseline represents the correlation from Geant4 in figure 6(b). The correlation for the generated data of the QAG model is presented in figure 6(c). It can be derived that the overall correlation pattern is accurately reproduced by the QAG model. Like the Geant4 data, it consists of a larger and more compact positively correlated group of pixels. The other pixels are negatively correlated. Inspecting the particular details, different color shades indicate some minor deviations. However, the achieved correlation precision by QAG is astonishing. Therefore, it can be concluded, that the quantum circuits are capable to reproduce complex correlation patterns through substantial entanglement strategies, as present in the MERA-up architecture.

    3. Energy sum: the energies contained in all pixels calculated for the individual images represent the third accuracy metric. In figure 7(a), the energy sum histogram of the Geant4 images reveals a Gaussian shape and is correctly reproduced visually by the QAG model, which is also confirmed by the mean μ and the standard deviation σ.
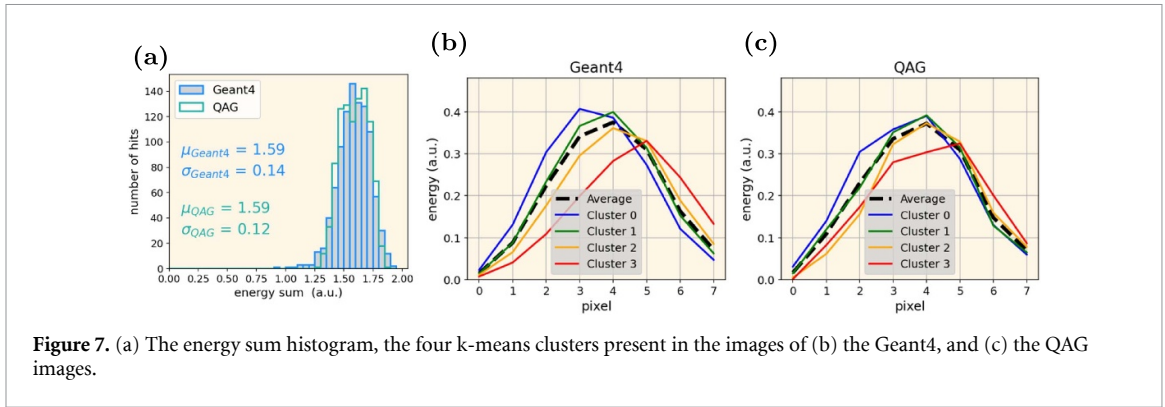
Figure 7. (a) The energy sum histogram, the four k-means clusters present in the images of (b) the Geant4, and (c) the QAG images.
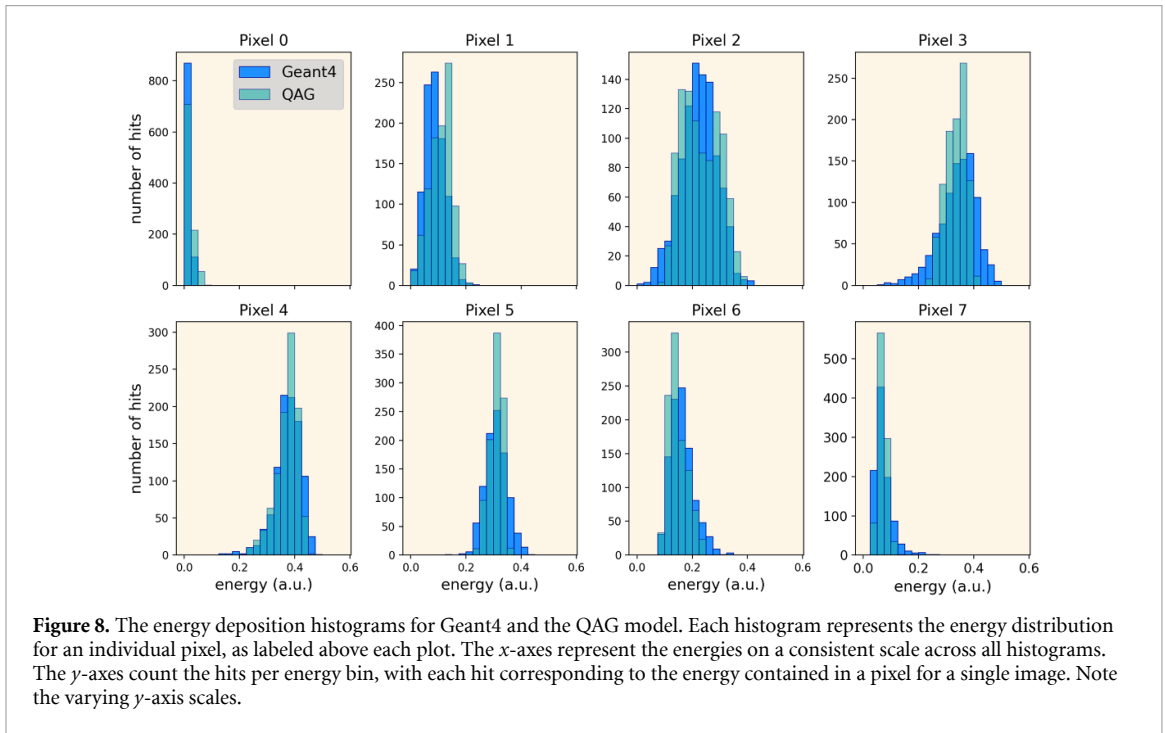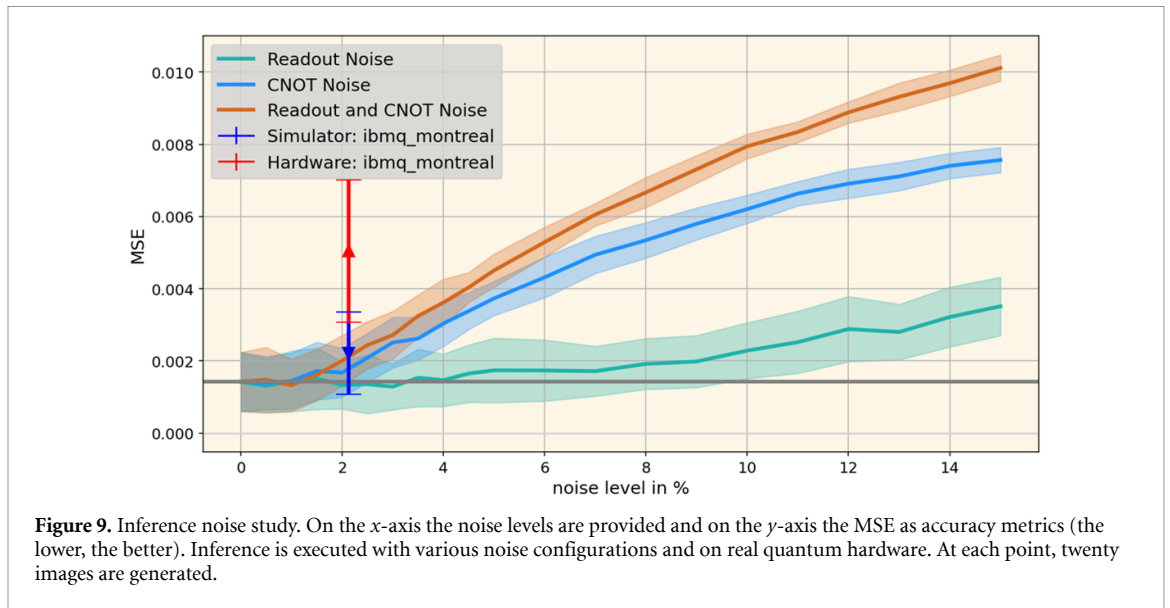

Figure 8. The energy deposition histograms for Geant4 and the QAG model. Each histogram represents the energy distribution for an individual pixel, as labeled above each plot. The *x*-axes represent the energies on a consistent scale across all histograms. The *y*-axes count the hits per energy bin, with each hit corresponding to the energy contained in a pixel for a single image. Note the varying *y*-axis scales.

4. k-means Clusters: this metric evaluates if the QAG model can correctly represent specific image modes. The Geant4 data is clustered with the k-means algorithm [36] to find four clusters or image modes as illustrated in figure 7(b). Cluster 0 deposits substantially larger fractions of energy in earlier calorimeter cells than in higher cluster numbers. Further, the particles from clusters 0 and 1 contain a larger energy fraction, estimated by the integral area below the curves. Here we are interested in whether the QAG model can reproduce this behavior. The four clusters of the QAG images are provided in figure 7(c). A similar structure can be observed, which indicates a good accuracy in reproducing the energy contents and image modes on average.

5. Pixel-wise energy distribution: the last metric is used to examine the distributions of the energy content of each pixel, as illustrated in figure 8. Overall, the histograms of the QAG model match those of the Geant4 model. Even pixels with non-Gaussian energy distributions in the Geant4 model are correctly reproduced by the QAG model. For example, the longer tail towards smaller energies on the left side of pixel 4 is equally present in the histogram for the QAG model. The large histogram overlaps indicate that not only the averages are reproduced with high accuracy, but also the energy distributions returned for each individual pixel are correct.

5. Quantum noise study

In the current NISQ era, relatively high hardware error levels are one of the primary limitations to effectively employing algorithms on real quantum devices. Similar as in the classical case, QML models appear to be noise resilient to some degree of hardware errors [37–40]. In the following, the robustness of the QAG model

Figure 9. Inference noise study. On the *x*-axis the noise levels are provided and on the *y*-axis the MSE as accuracy metrics (the lower, the better). Inference is executed with various noise configurations and on real quantum hardware. At each point, twenty images are generated.

to simulated noise is tested in inference and training. Furthermore, training and inference are executed on real quantum devices with measured noise levels and compared to the results with simulated noise.

5.1. Inference

In a first step, quantum noise is applied only to the inference of a model trained without noise. Inference is performed using three different noise configurations: simulated noise at varying levels, simulated noise derived from the later used hardware, and finally, with the real quantum hardware. In the simulated noise configurations, each qubit noise is modeled with the same readout measurement error, and each inter-qubit connection with the same two-gate (CNOT) error. In the combined noise model both are on the same level. In contrast, the hardware noise levels can vary widely for individual qubits as well as for gates.

Multiple noise configurations and error levels from zero up to 15% are tested. The MSE is utilized as the accuracy measure. The results are illustrated in figure 9. For each configuration, the average value of 20 generated images is plotted in dependence of the noise level as a solid line and the standard deviation as a colored band around the mean. The gray horizontal line serves as accuracy reference for the noise-free configuration. All configurations maintain sufficient accuracy up to approximately 1.5% of noise. The configuration with readout noise only (green) is most robust and maintains a stable accuracy of up to 8% of noise. CNOT noise only (blue) and both readout noise and CNOT noise combined (orange) experience a stronger impact. As expected, the combined configuration (orange) performs worst. As a side note, current quantum devices have average noise levels below 5%, which is expected to gradually decrease further. But as discussed in the following, the noise levels are unstable and can sometimes spike up. Therefore, wider noise ranges were investigated.

Next, the inference is run by loading the real hardware noise model from `ibmq_montreal` into the simulator. The `ibmq_montreal` device consists of a Falcon r4 processor with 27 qubits. The average readout noise over the qubits employed at the time of the test is 2.51% and the average gate noise level is 0.97%. The explicit noise model, containing the entries for each qubit, is provided in the appendix in figure 11(b). The result is included in figure 9 (blue triangle). The noise level position (*x*-axis) is determined as the average of readout and CNOT noise. Although the noise levels of the qubits vary strongly, the measured accuracy of the hardware noise simulation agrees well with the simulated noise in mean and standard deviation within the uncertainties. This suggests that a model trained without any noise would theoretically be able to run inference on noisy hardware without a significant drop in accuracy.

Finally, the inference is executed on the real `ibmq_montreal` device. The result is plotted as a red triangle in figure 9. The accuracy on the real hardware is worse than predicted by the simulation, as indicated by a larger MSE value. The decomposition of the circuit to the real hardware includes swap operations, which imply additional two-qubit entanglement gates for the quantum circuit. It is possible that these are not included in the hardware noise simulation and lead to higher noise influence on real hardware and thus to worse results.
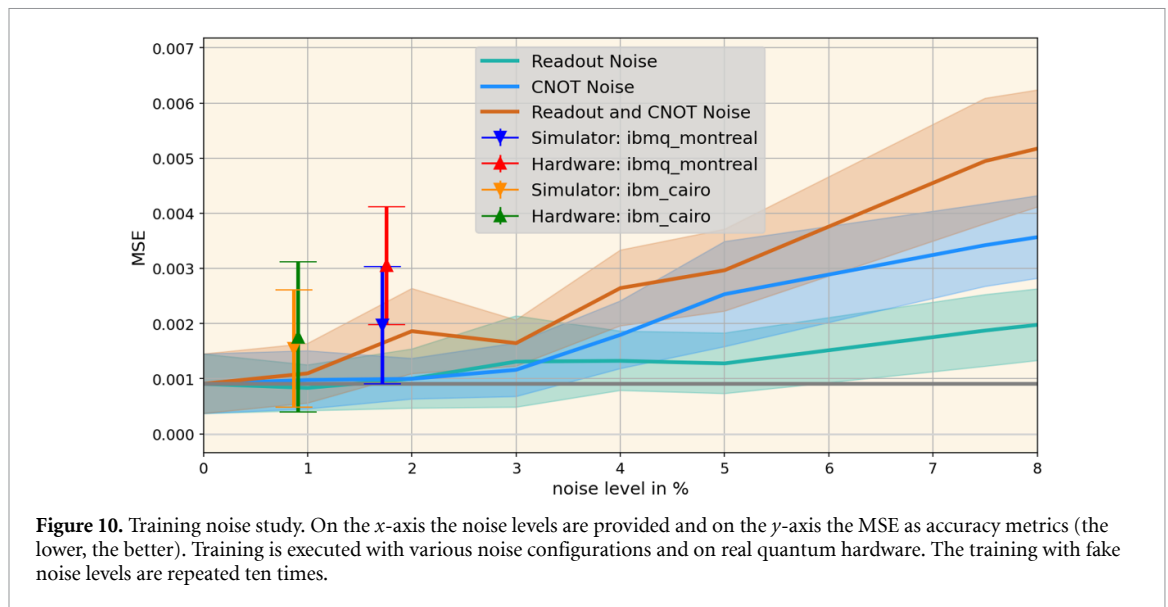
Figure 10. Training noise study. On the *x*-axis the noise levels are provided and on the *y*-axis the MSE as accuracy metrics (the lower, the better). Training is executed with various noise configurations and on real quantum hardware. The training with fake noise levels are repeated ten times.
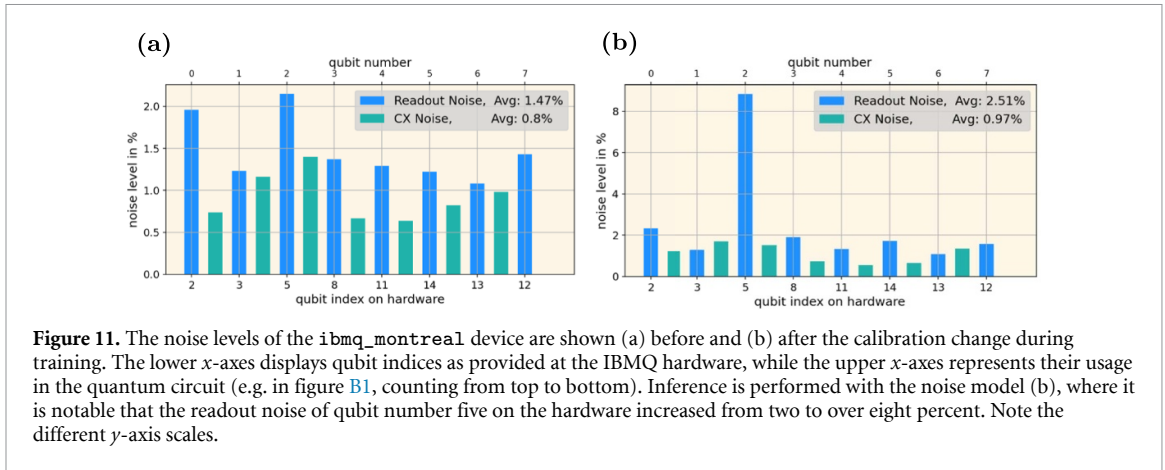
5.2. Training

The noise study is repeated to include noise also during training. We study if the QAG model can learn to compensate for noise in training, especially when running on the real quantum device. Besides, we investigate with which noise values the model can still maintain a reasonable accuracy.

The results are provided in figure 10. The configurations with readout noise (green) and CNOT noise (blue) maintain a similar level of accuracy until approximately 3%. The accuracy of the combination of readout and CNOT noise (orange) decreases marginally from 1% of noise and further with larger noise levels. However, at 3% of noise, its accuracy is still close to the noiseless case, staying within one standard deviation. This indicates that training the model with noise makes the QAG model more robust than only applying noise to a trained model in inference only.

Next, two quantum devices are simulated. `ibmq_montreal` and `ibm_cairo`, which are both 27 qubit devices, but `ibm_cairo` has the more advanced Falcon r5.11 processor. Likewise, the training is repeated ten times and the results are added as blue and orange triangles in figure 10. It can be noted that the average accuracy of the training with hardware noise performs slightly worse than the simulated combined noise model (orange line). The strong overlap between the error from the noise simulation (orange band) and the hardware noise simulation (orange and blue error bar) indicates that the accuracy difference is statistically not significant.

Finally, the entire training was executed on the real quantum device. First, the training was carried out on `ibmq_montreal`. During training, around epoch 280, an unpredicted significant noise change occurred and the readout noise of one qubit increased to 8%, as shown in figure 11. As a result, the MMD loss spikes up, as shown in figure 12(a). This negatively influenced the training. However, after the noise change during training, the model recovered and adapted to the new noisy environment. In the still remaining number of epochs the loss decreased to a modest level. The training was repeated on the best performing `ibm_cairo` device without having a hardware calibration change during training, and the training losses are shown in figure 12(b).

The results of both hardware training are included as red and green triangles in figure 10. The error bars correspond to the accuracy deviations within 50 generated validation images. It can be observed that the average accuracy of the `ibmq_montreal` hardware training (red) is visibly worse than the noise simulation (blue). This is most likely due to the large noise increase and the fact that the model did not have enough remaining epochs to fully recover. This is also indicated by the still decreasing losses towards the end of the training. As mentioned above, the training was repeated on the more stable machine `ibm_cairo`. This time, the training was completed without calibration changes and with about 1% lower hardware noise level (readout noise 0.86% and CNOT noise 0.89%). The accuracy of the hardware training on `ibm_cairo` (green) is only marginally worse than the one from the simulator (orange). This indicates that the simulated and real hardware results behave similarly for low hardware noise levels. This fulfills the expectations derived from the pure simulation that exhibit only statistically insignificant variations in the very good accuracy at these low noise levels.

Figure 11. The noise levels of the `ibmq_montreal` device are shown (a) before and (b) after the calibration change during training. The lower *x*-axes displays qubit indices as provided at the IBMQ hardware, while the upper *x*-axes represents their usage in the quantum circuit (e.g. in figure B1, counting from top to bottom). Inference is performed with the noise model (b), where it is notable that the readout noise of qubit number five on the hardware increased from two to over eight percent. Note the different *y*-axis scales.
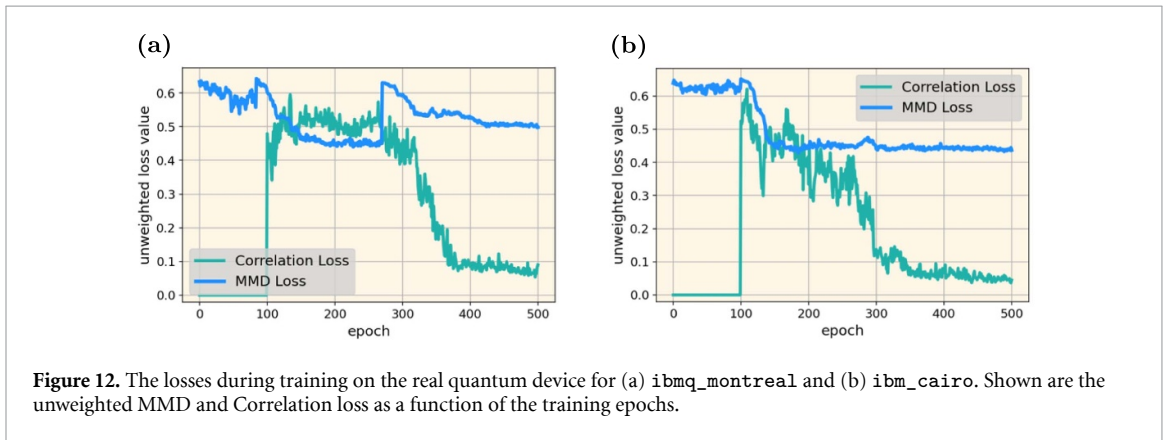
Figure 12. The losses during training on the real quantum device for (a) `ibmq_montreal` and (b) `ibm_cairo`. Shown are the unweighted MMD and Correlation loss as a function of the training epochs.

Comparing the absolute MSE magnitude of the real hardware training ($\approx$0.002 and $\approx$0.003) with the one from inference from the previous section ($\approx$0.005), the accuracy improved, suggesting that the QAG model is able to adapt its parameters to the noisy hardware to improve its precision. This is also confirmed by the `ibmq_montreal` training, where the accuracy worsened entirely after the calibration change, but then recovered. In the appendix in figure E3(a), the average shower image created by the `ibmq_montreal` device is visualized, and in figure E3(b) by `ibm_cairo`. The shower image of `ibm_cairo` agrees well with Geant4, whereas `ibmq_montreal` exhibits some deficiencies because of the missing remaining training epochs after the calibration.

6. Conclusion

The results of this study clearly demonstrate that the newly developed QAG model is capable of generating images with good precision, as measured with a variety of validation metrics. This includes correctly reproducing average values, but most importantly, also complex pixel-wise correlations with the chosen optimal MERA-up quantum architecture. These results reveal that the QAG model with a good entangled circuit is capable of learning intrinsic correlation patterns from the training data.

Our study exhibit the significant impact that quantum hardware noise can have on the accuracy of quantum machine learning models. The results evidence that training the models with noise leads to better performance (stable until 3% noise) because the QAG model adapts to the underlying noise behavior and converges faster in contrast to the situation of applying noise in inference only (stable until 1.5% noise). This was also verified on the real hardware by the `ibm_cairo` training. Furthermore, our study shows that the QAG model is robust and can produce accurate results even with significant hardware calibration changes with up to 8% noise, as demonstrated by the training on `ibmq_montreal`. Overall, the newly developed QAG model demonstrates that training quantum machine learning models with realistic quantum hardware noise can lead to robust models and accurate results, which is of great importance for the future development of real world quantum machine learning applications.

Data availability statement

Acknowledgment

Appendix A. QAG inference algorithm

Algorithm 1 shows the pseudo code for generating one image using the QAG model from section 3.1. Here the MERA-up architecture (appendix B) is used as an example. The standard deviation σ_{pixel} for the energy distribution of each pixel is obtained from training data.

Algorithm 1. Quantum angle generator inference.

Input: nb_{shots} ▷ Number of measurements
σ_{pixel} ▷ Array of the pixel standard deviations

Output: E ▷ Array of pixel energies forming the image

Begin
    Initialize all 8 qubits to state $|0\rangle$
    Randomly draw value v from uniform distribution $[-0.25, 0.25]$
    for all index, qubits **do**
        Apply Hadamard gate H to qubit
        Randomly draw Ω from uniform distribution $[-1, 1]$
        $\Omega_{pixel} = \Omega \cdot v \cdot \sigma_{pixel}[index]$
        Apply rotation $Ry(\Omega_{pixel})$ to qubit
    end for
    Apply the MERA-up unitary transformation to all qubits
    Measure the qubits nb_{shots} times
    Count number of times state $|0\rangle$ is measured $\rightarrow$ counts($|0\rangle$)
    Compute θ according to equation (1)
   Compute pixel energies E according to transformation 3
    Return E
End

Appendix B. Quantum circuit architectures under study

The quantum circuit architectures investigated in this paper are summarized in figure B1. In general, it was observed that hierarchical architectures perform best while maintaining a reasonable number of quantum gates and parameters. Specifically, we examine the tree tensor network (TTN) architecture and multi-scale entanglement renormalization ansatz (MERA) introduced in [41]. Multiple variations of these circuits are tested: (1) circuits with a depth two (naming scheme 'd2'). These circuits contain two layers, with all circuit gates placed twice, to evaluate if deeper circuits perform better. (2) Circuits additional with Rz-gates are employed after each Ry-gate (naming scheme 'Rz') to assess if rotations around an additional axis can improve the accuracy. Both architecture variants double the number of parameters of the initial circuit.

For the MERA architecture, further variations are analyzed where only the right circuit half from the original architecture is implemented, denoted as the MERA upsampling (MERA-up) circuit. In the
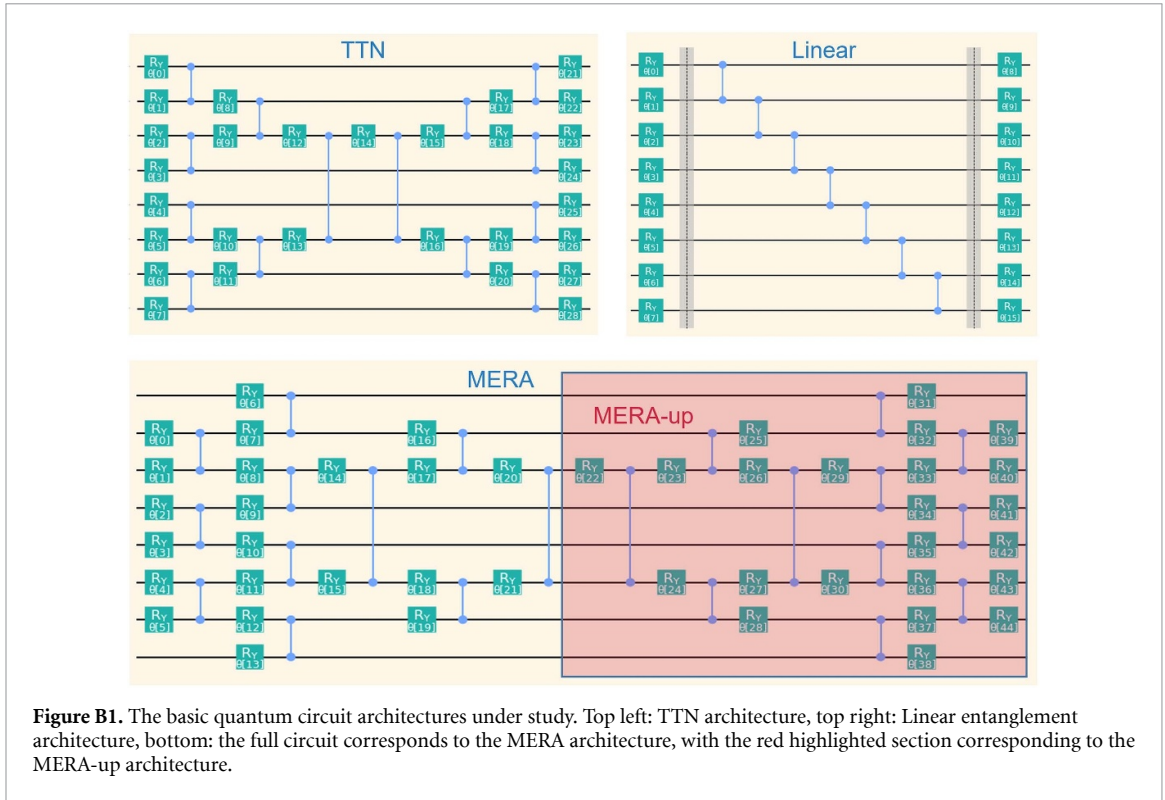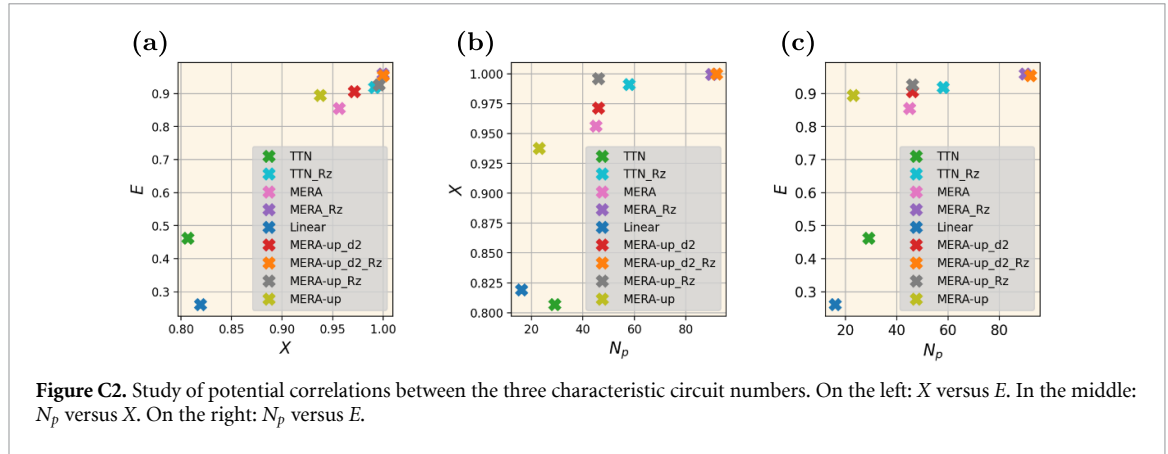
Figure B1. The basic quantum circuit architectures under study. Top left: TTN architecture, top right: Linear entanglement architecture, bottom: the full circuit corresponds to the MERA architecture, with the red highlighted section corresponding to the MERA-up architecture.

MERA-up circuit, the information is upsampled from the central qubit and spread to all the other qubits, similar to what happens in classical ML generative models, for example, transpose convolutional neural networks. The left half of the MERA circuit, the MERA downsampling (MERA-down) architecture, is not tested because it would rather compress the information as required in classification tasks that are not used in this paper. The last architecture contains a simple linear entanglement strategy.

In [35], multiple complex four-qubit architectures are compared. However, these architectures are not exploited in our study. The reason is most of them contain many more gates and parameters, and they do not scale well for more than four qubit circuits. Also, many architectures employ parameterized two-qubit rotational gates which we found, taking the decomposition into account, not as effective as the combination of separate rotation and entanglement gates.

Appendix C. Characteristic circuit numbers

The results of evaluating the characteristic circuit numbers (number of parameters N_p, expressibility X and entanglement capability E) are shown in figure C2. In figure C2(a) E is plotted as a function of X. The MERA_Rz and MERA-up_d2_Rz architecture perform best and lie almost on top of each other in the top right corner, directly followed by MERA-up_Rz and TTN_Rz. The Linear architecture and the basic TTN perform worst. It can be clearly noted that circuits with more gates (Rz, d2) perform better. N_p is studied in figures C2(b) and (c). It can be seen that the best two circuits—MERA_Rz and MERA-up_d2_Rz—contain by far the largest number of parameters. However, a more limited number of parameters is desirable to address NISQ hardware limitations. The MERA-up_Rz and TTN_Rz architectures contain approximately half the number of parameters compared to the MERA_Rz and MERA-up_d2_Rz circuits but perform almost similarly accurately. Therefore, they may be preferred in practice. Further reducing the number of parameters, the MERA-up architecture with only $N_p = 23$ parameters maintains adequate values for E and X and is the baseline circuit for the more detailed studies. All characteristic circuit numbers measured in the circuit study are summarized in table C1. In the table, it is evident that the MERA-up circuit maintains a considerably lower training time on the classical hardware compared to models with a higher number of parameters. The training was executed on a cluster equipped with 32 Intel Xeon Gold 6130 CPU cores running at 2.10 GHz.

Figure C2. Study of potential correlations between the three characteristic circuit numbers. On the left: X versus E. In the middle: N_p versus X. On the right: N_p versus E.

Table C1. The characteristic circuit values N_p, X and E are listed. Additionally, the statistics of the MSE accuracy metrics and the training times t in minutes are provided for the individual circuits for repeating the training 25 times on the calorimeter data set. The line highlighted in bold corresponds to the MERA-up circuit, which is excessively evaluated in this paper in section 4 and 5.

Circuit Name	N_p	X	E	MSE	t in min
Linear	16	0.8191	0.261	0.00113 ± 0.0008	35.0 ± 0.2
TTN	29	0.8068	0.462	0.00245 ± 0.0012	41.5 ± 0.6
TTN_Rz	58	0.9912	0.918	0.00440 ± 0.0022	85.9 ± 1.5
MERA	45	0.9564	0.855	0.00220 ± 0.0011	51.3 ± 0.5
MERA_Rz	90	0.9997	0.959	0.00634 ± 0.0036	121.5 ± 9.6
MERA-up	**23**	**0.9377**	**0.894**	$\mathbf{0.00059 \pm 0.0004}$	$\mathbf{39.4 \pm 2.9}$
MERA-up_d2	46	0.9715	0.906	0.00038 ± 0.0002	50.5 ± 0.2
MERA-up_Rz	46	0.9961	0.926	0.00047 ± 0.0005	75.2 ± 7.5
MERA-up_d2_Rz	92	0.9999	0.954	0.00094 ± 0.0008	116.2 ± 4.3

Appendix D. Definitions of accuracy metrics

In this section, we provide detailed formulas for the inference accuracy metrics discussed in section 4.2.

D.1. MSE of the average shower shape

The MSE for the average calorimeter shower shape is derived from the difference between the average images of Geant4 and the QAG model, calculated over 980 generated inference images and the corresponding 980 Geant4 images. In the subsequent equations, k iterates over the $N = 980$ images and i denotes the ith pixel of each image. Hence, $E_{k,i}^{\text{G4}}$ (Geant4) and $E_{k,i}^{\text{QAG}}$ (QAG model) correspond to the energy of pixel i from image k.

The average pixel energies $\overline{E}_i^{\text{G4}}$ and $\overline{E}_i^{\text{QAG}}$ are computed as follows:

$$\overline{E}_i^{\text{G4}} = \frac{1}{N}\sum_{k=1}^{N} E_{k,i}^{\text{G4}} \quad \text{and} \quad \overline{E}_i^{\text{QAG}} = \frac{1}{N}\sum_{k=1}^{N} E_{k,i}^{\text{QAG}}. \tag{D1}$$

The MSE is then calculated by iterating over the $M = 8$ individual pixels by the index i:

$$\text{MSE} = \frac{1}{M}\sum_{i=1}^{M}\left(\overline{E}_i^{\text{G4}} - \overline{E}_i^{\text{QAG}}\right)^2. \tag{D2}$$

As the MSE corresponds to an error, the smaller the value, the better the accuracy of the QAG model.

D.2. Energy sum histogram

Each histogram entry (or hit) is the total energy $\hat{E}$ of a shower image:

$$\hat{E}^{\text{G4}} = \sum_{i=1}^{M} E_i^{\text{G4}} \quad \text{and} \quad \hat{E}^{\text{QAG}} = \sum_{i=1}^{M} E_i^{\text{QAG}}, \tag{D3}$$

where i is iterating over the $M = 8$ pixels.

The means of the distributions for our data sets, Geant4 and QAG, are denoted as μ^{G4} and μ^{QAG}, respectively. Similarly, the standard deviations are represented by σ^{G4} and σ^{QAG}.
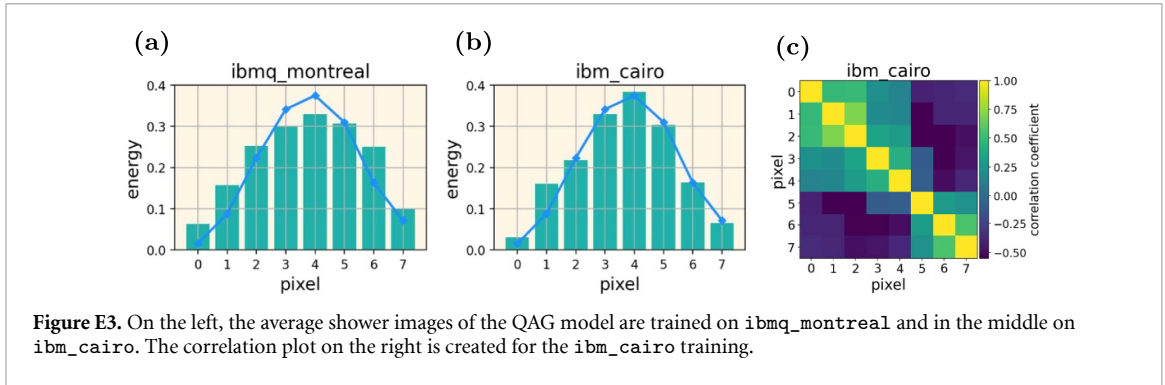
Figure E3. On the left, the average shower images of the QAG model are trained on `ibmq_montreal` and in the middle on `ibm_cairo`. The correlation plot on the right is created for the `ibm_cairo` training.

The means μ of the total energy for the $N = 980$ images are then calculated using the well-known equation:

$$\mu^{\mathrm{G4}} = \frac{1}{N}\sum_{i=1}^{N}\hat{E}_i^{\mathrm{G4}} \quad \text{and} \quad \mu^{\mathrm{QAG}} = \frac{1}{N}\sum_{i=1}^{N}\hat{E}_i^{\mathrm{QAG}}. \tag{D4}$$

And the standard deviations, denoted by σ, are computed as:

$$\sigma^{\mathrm{G4}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{E}_i^{\mathrm{G4}} - \mu^{\mathrm{G4}}\right)^2}$$

and $\hspace{8cm}$ (D5)

$$\sigma^{\mathrm{QAG}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{E}_i^{\mathrm{QAG}} - \mu^{\mathrm{QAG}}\right)^2}$$

A closer agreement between the means and standard deviations of the two distributions indicates a higher accuracy of the QAG model.

Appendix E. Full training on quantum device

The average shower images of the quantum hardware training are correctly reproduced, as provided in figure E3. The correlation plot for the `ibm_cairo` training is provided in figure E3(c). The overall correlation pattern is correctly reproduced. All in all, the model trained on `ibm_cairo` shows a good performance.

ORCID iDs

Florian Rehm ⓘ https://orcid.org/0000-0002-8337-0239
Sofia Vallecorsa ⓘ https://orcid.org/0000-0002-7003-5765
Kerstin Borras ⓘ https://orcid.org/0000-0003-1111-249X
Dirk Krücker ⓘ https://orcid.org/0000-0003-1610-8844
Michele Grossi ⓘ https://orcid.org/0000-0003-1718-1314
Valle Varo ⓘ https://orcid.org/0000-0002-6524-4298

References

[1] Zidan M, Eleuch H and Abdel-Aty M 2021 Non-classical computing problems: toward novel type of quantum computing problems *Results Phys.* **21** 103536
[2] Preskill J 2018 Quantum computing in the NISQ era and beyond *Quantum* **2** 79
[3] Huang H-Y *et al* 2021 Power of data in quantum machine learning *Nat. Commun.* **12** 2631
[4] Clissa L 2022 Survey of big data sizes in 2021 (arXiv:2202.07659)
[5] CERN: Worldwide LHC Computing Grid (available at: https://wlcg-public.web.cern.ch/) (Accessed June 2022)
[6] Zheng H, Li Z, Liu J, Strelchuk S and Kondor R 2022 On the super-exponential quantum speedup of equivariant quantum machine learning algorithms with SU(d) symmetry (arXiv:2207.07250)
[7] Wu S L *et al* 2022 Application of quantum machine learning to high energy physics analysis at LHC using quantum computer simulators and quantum computer hardware *Proc. Sci.* **842**

[8] Khrennikov A 2021 Roots of quantum computing supremacy: superposition, entanglement, or complementarity? *Eur. Phys. J. Spec. Top.* **230** 1053–7

[9] Weigold M, Barzen J, Leymann F and Salm M 2020 Data encoding patterns for quantum computing *Proc. 27th Conf. on Pattern Languages of Programs* (*PLoP '20*) (The Hillside Group) (https://doi.org/10.1109/ICSA-C52384.2021.00025)

[10] Weigold M, Barzen J, Leymann F and Salm M 2021 Expanding data encoding patterns for quantum algorithms *2021 IEEE 18th Int. Conf. on Software Architecture Companion* (*ICSA-C*) pp 95–101

[11] Liu J-G and Wang L 2018 Differentiable learning of quantum circuit born machines *Phys. Rev.* A **98** 062324

[12] Khoshaman A, Vinci W, Denis B, Andriyash E, Sadeghi H and Amin M H 2018 Quantum variational autoencoder *Quantum Sci. Technol.* **4** 014001

[13] Li T, Xia J and Zhang S 2020 Quantum generative adversarial network: a survey *Comput. Mater. Contin.* **64** 401–38

[14] Rehm F, Vallecorsa S, Borras K and Krücker D 2021 Quantum machine learning for HEP detector simulations *Conf. Proc. "Distributed Computing and Grid Technologies in Science and Education"* (*GRID 2021*) (Dubna) (available at: http://ceur-ws.org/Vol-3041/363-368-paper-67.pdf)

[15] Dallaire-Demers P-L and Killoran N 2018 Quantum generative adversarial networks *Phys. Rev.* A **98** 012324

[16] Rehm F, Vallecorsa S, Grossi M, Borras K, Krücker D, Schnake S and Verney-Provatas A-H 2022 Quantum angle generator for image generation *2022 IEEE/ACM 7th Symp. on Edge Computing* (*SEC*) pp 425–9 (available at: https://inspirehep.net/literature/2620987)

[17] Agostinelli S *et al* 2003 GEANT4–a simulation toolkit *Nucl. Instrum. Methods Phys. Res.* A **506** 250–303

[18] Elsen E 2019 A roadmap for HEP software and computing research and development for the 2020s *Comput. Softw. Big Sci.* **3** 16

[19] Apostolakis J 2020 *Detector Simulation* ed C W Fabjan and H Schopper (Springer) pp 485–531

[20] Kadochnikov I *et al* 2018 WLCG data lake prototype for HL-LHC *Advisory Committee* 127 (available at: https://ceur-ws.org/Vol-2267/509-512-paper-97.pdf)

[21] Ju X *et al* 2021 Performance of a geometric deep learning pipeline for HL-LHC particle tracking *Eur. Phys. J.* C **81** 1–14

[22] Biscarat C, Caillou S, Rougier C, Stark J and Zahreddine J 2021 Towards a realistic track reconstruction algorithm based on graph neural networks for the HL-LHC *EPJ Web Conf.* **251** 03047

[23] Sela E, Huang S and Horn D 2022 Deep learning study of an electromagnetic calorimeter *Algorithms* **15** 115

[24] Fabjan C and Gianotti F 2003 Calorimetry for particle physics *Rev. Mod. Phys.* **75** 1243–86

[25] Pierini M and Zhang M 2020 CLIC calorimeter 3D images: electron showers at fixed angle (Zenodo) (https://doi.org/10.5281/zenodo.3603122)

[26] Rehm F 2021 Downsampled calorimeter shower images to 8 pixels (Zenodo) (https://doi.org/10.5281/zenodo.7025233)

[27] LaRose R and Coyle B 2020 Robust data encodings for quantum classifiers *Phys. Rev.* A **102**

[28] Sierra-Sosa D, Telahun M and Elmaghraby A 2020 Tensorflow quantum: impacts of quantum state preparation on quantum machine learning performance *IEEE Access* **8** 215246–55

[29] Rehm F *et al* 2021 Reduced precision strategies for deep learning: a high energy physics generative adversarial network use case *Proc. 10th Int. Conf. on Pattern Recognition Applications and Methods* (https://doi.org/10.5220/0010245002510258)

[30] Gretton A *et al* 2012 A kernel two-sample test *J. Mach. Learn. Res.* **13** 723–73 (available at: https://api.semanticscholar.org/CorpusID:10742222)

[31] Hofmann T, Scholkopf B and Smola A 2007 Kernel methods in machine learning *Ann. Stat.* **36** 1171–220 (available at: https://api.semanticscholar.org/CorpusID:9989562)

[32] Alhabsi A H 2010 Improved SPSA optimization algorithm requiring a single measurement per iteration *10th Int. Conf. on Information Science, Signal Processing and Their Applications* (*ISSPA 2010*) pp 263–5

[33] Akiba T, Sano S, Yanase T, Ohta T and Koyama M 2019 Optuna: a next-generation hyperparameter optimization framework (arXiv:1907.10902)

[34] IBM Quantum Services IBM 2022 (available at: https://quantum-computing.ibm.com/services?services=systems) (Accessed 25 August 2022)

[35] Sim S, Johnson P D and Aspuru-Guzik A 2019 Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms *Adv. Quantum Technol.* **2** 1900070

[36] Hartigan J A and Wong M A 1979 A k-means clustering algorithm *Appl. Stat.* **28** 100–8

[37] Wang H *et al* 2021 QuantumNAS: noise-adaptive search for robust quantum circuits (arXiv:2107.10845)

[38] Bogdanov Y I, Bogdanova N A, Fastovets D V and Lukichev V F 2022 Study of the effect of quantum noise on the accuracy of the Schrodinger equation simulation on a quantum computer using the Zalka-Wiesner method (arXiv:2201.03068)

[39] Borras K *et al* 2023 Impact of quantum noise on the training of quantum generative adversarial networks *J. Phys.: Conf. Ser.* **2438** 012093

[40] Wang H, Gu J, Ding Y, Li Z, Chong F, Pan D Z and Song han: 2022 RoQNN: noise-aware training for robust quantum neural networks (available at: https://openreview.net/forum?id=wwIBobGFj2V)

[41] Grant E, Benedetti M, Cao S, Hallam A, Lockhart J, Stojevic V, Green A G and Severini S 2018 Hierarchical quantum classifiers *npj Quantum Inf.* **4** 65