Finding the Higgs boson with quantum machine learning

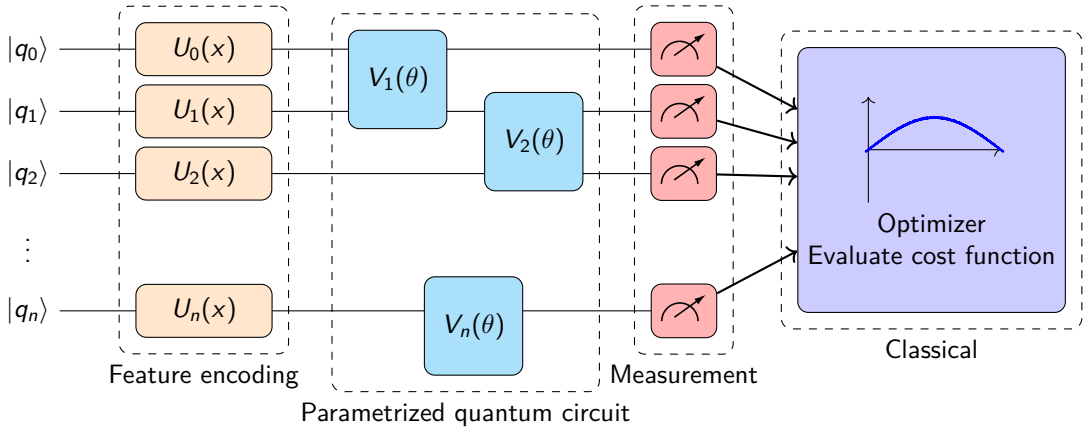
Eric Ballabene

University and INFN, Bologna
Department of Physics and Astronomy "Augusto Righi"

3rd Workshop on Quantum Computing @ INFN
Milan, February 3-6, 2026

Outline

Introduction



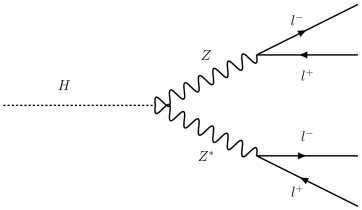
The physics case

Classification task: separation of a signal process from background processes.

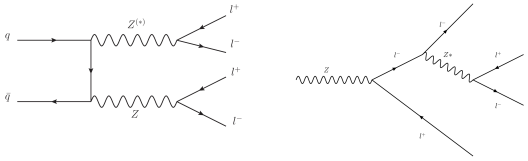
Processes:

- Signal, Higgs boson golden channel, $H \to ZZ^* \to 4\ell$
- Irreducible background, ZZ^*
- Reducible backgrounds, $Z, t\bar{t}, t\bar{t}V, VVV$

Starting from jupyter notebook and using the available ATLAS Open Data [1].



Higgs boson signal



ZZ* and Z boson backgrounds

The physics case

Pre-selection:

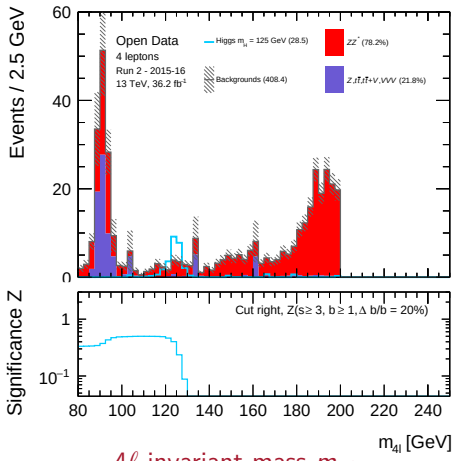
- Presence of 4ℓ
- Single lepton trigger matching
- $m_{4\ell} < 250$ GeV (training)

Input features:

- $m_{4\ell}$,
- $p_{\mathrm{T}}^{\ell_1}$, $p_{\mathrm{T}}^{\ell_2}$, $p_{\mathrm{T}}^{\ell_3}$, $p_{\mathrm{T}}^{\ell_4}$

Selection:

- $m_{4\ell} < 200$ GeV (score distributions)

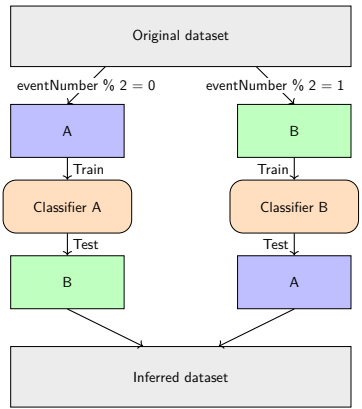


4ℓ invariant mass $m_{4\ell}$

Pre-processing

Training and inference over two independent subsets, so that the events of a subset are never predicted with a classifier trained over the events of the same subset:

- The dataset is split into subsets A and B by event number parity.
- Each subset trains its classifier, which is tested on the opposite subset.
- The resulting tested subsets are merged into a final inferred dataset.



Pre-processing

Physics observables as input variables x_i

$$\left(m_{4\ell}, p_{\mathrm{T}}^{\ell_1}, p_{\mathrm{T}}^{\ell_2}, p_{\mathrm{T}}^{\ell_3}, p_{\mathrm{T}}^{\ell_4}\right) \longmapsto (x_0, x_1, x_2, x_3, x_4) \tag{1}$$

x_i are rescaled to $\tilde{x}_i \in [-\pi, \pi]$

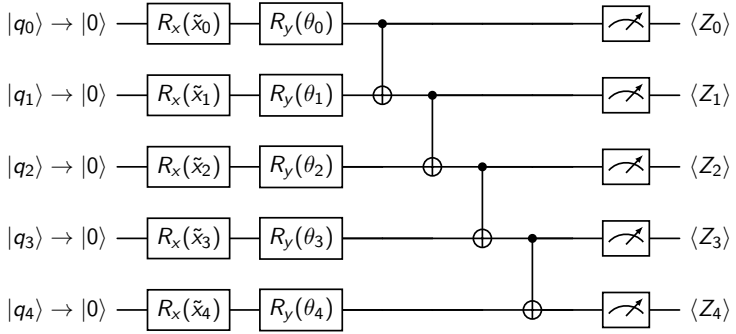
$$\tilde{x}_i = \begin{cases} \left(2\,\dfrac{x_i - x_i^{\min}}{x_i^{\max} - x_i^{\min}} - 1\right)\pi, & x_i^{\max} \neq x_i^{\min}, \\ 0, & x_i^{\max} = x_i^{\min}, \end{cases} \qquad i = 0, \ldots, 4 \tag{2}$$

Qubits are prepared in the $|0\rangle$ state and the features are encoded using R_x rotations

$$|\psi\rangle = \left(\prod_{i=0}^{4} R_x(\tilde{x}_i)\right)|0\rangle^{\otimes 5}, \qquad R_x(\alpha) = e^{-i\alpha X/2}. \tag{3}$$

The linear PQC

The PQC consists of a set of R_x rotations applied to the encoded input features and a linear sequence of CNOT gates that give expressibility to the model [2].



Expectation values of the PQC

We consider the initial state and circuit unitary

$$|\psi_0\rangle = |0\rangle^{\otimes 5}, \qquad U = U_{\mathrm{CNOT}} \left(\bigotimes_{i=0}^{4} R_y(\theta_i) \right), \qquad U_{\mathrm{CNOT}} = \prod_{k=0}^{3} \mathrm{CNOT}_{k \to k+1}, \tag{4}$$

and measure

$$\langle Z_i \rangle = \langle \psi_0 | \, U^\dagger Z_i U \, | \psi_0 \rangle \, . \tag{5}$$

Using the Heisenberg action of a CNOT gate,

$$U^\dagger_{\mathrm{CNOT}} Z_c U_{\mathrm{CNOT}} = Z_c, \qquad U^\dagger_{\mathrm{CNOT}} Z_t U_{\mathrm{CNOT}} = Z_c Z_t, \tag{6}$$

the observable propagates along the chain as

$$U^\dagger_{\mathrm{CNOT}} Z_i U_{\mathrm{CNOT}} = \prod_{k=0}^{i} Z_k. \tag{7}$$

Defining the rotated product state

$$|\phi\rangle = \bigotimes_{j=0}^{4} R_y(\theta_j)\,|0\rangle\,, \tag{8}$$

the expectation value factorizes,

$$\langle Z_i\rangle = \langle\phi|\left(\prod_{k=0}^{i} Z_k\right)|\phi\rangle = \prod_{k=0}^{i}\langle\phi_k|\,Z_k\,|\phi_k\rangle\,, \qquad |\phi_k\rangle = R_y(\theta_k)\,|0\rangle\,. \tag{9}$$

$$\langle Z_i\rangle = \prod_{k=0}^{i}\langle Z_k\rangle, \qquad \langle Z_k\rangle = \langle 0|\,R_y^{\dagger}(\theta_k) Z R_y(\theta_k)\,|0\rangle = \cos\theta_k, \tag{10}$$

which yields

$$\begin{aligned}
\langle Z_0\rangle &= \cos\theta_0,\\
\langle Z_1\rangle &= \cos\theta_0\cos\theta_1,\\
\langle Z_2\rangle &= \cos\theta_0\cos\theta_1\cos\theta_2,\\
\langle Z_3\rangle &= \cos\theta_0\cos\theta_1\cos\theta_2\cos\theta_3,\\
\langle Z_4\rangle &= \cos\theta_0\cos\theta_1\cos\theta_2\cos\theta_3\cos\theta_4.
\end{aligned} \tag{11}$$

Model Configuration

Training performed using Tensorflow Quantum and Cirq [3, 4]

Compilation:

- **Optimizer:** `Adam`, learning rate `1e-3`
- **Loss:** `BinaryCrossentropy`
- **Metrics:** `accuracy`

Callbacks:

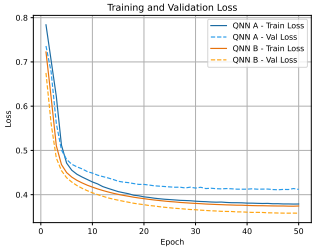
- **EarlyStopping:** Monitor: `val_loss`; Patience: `4`; Restore best weights: `True`
- **ReduceLROnPlateau:** Monitor: `val_loss`; Factor: `0.5`; Patience: `2`

Training:

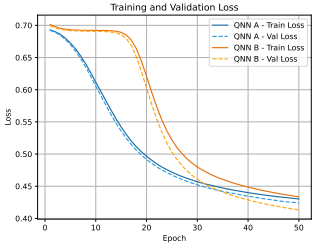
- Epochs: `50`
- Batch size: `128`
- Validation split: `0.2`

Hyper-parameters

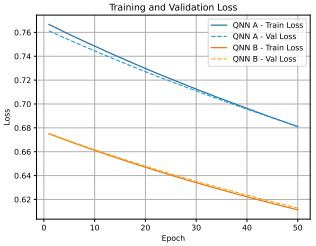
Learning rate = 1e-2



Learning rate = 1e-3

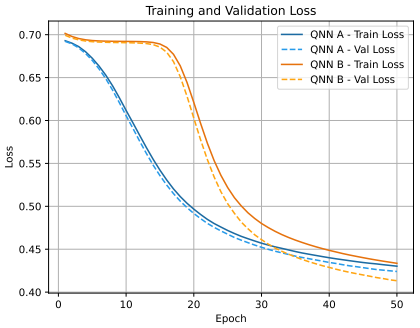


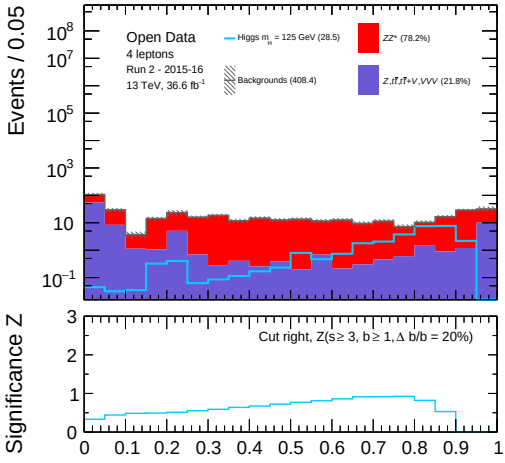
Learning rate = 1e-4



Output score with linear PQC

Learning rate = 1e-3





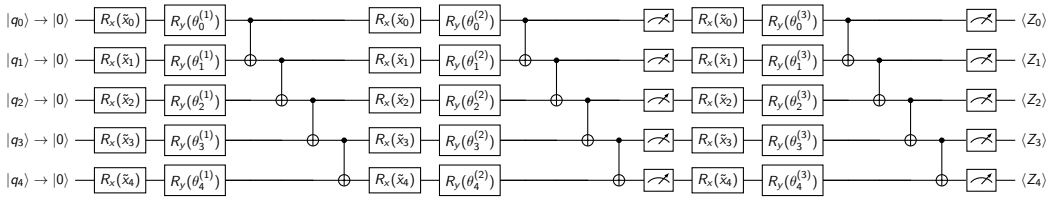
Data Re-Uploading PQC

A single data-encoding layer followed by a simple PQC has very limited expressive power.
Data re-uploading means encoding the same classical input features into the quantum circuit multiple times, interleaved with trainable quantum layers [7]:

- Each repetition is called a re-uploading layer.
- Data re-uploading lets the circuit build nonlinear feature interactions and behave more like a deep neural network.
- Data re-uploading allows for improving expressivity by reusing the same qubits multiple times, rather than increasing qubits, making QML more hardware-friendly [5, 6].

Data Re-Uploading PQC

The same input features $\tilde{x}_i$ are encoded multiple times through $R_x(\tilde{x}_i)$ gates. Each re-uploading layer consists of a data encoding block, followed by trainable rotations $R_y(\theta_i^{(\ell)})$ and an entangling CNOT structure.



Here $L = 3$ re-uploading layers are considered, with layer-dependent trainable parameters $\theta_i^{(\ell)}$.

From Factorized to Non-Factorized Expectation Values

In the single-layer PQC, the expectation values take the form

$$\langle Z_i \rangle = \prod_{k=0}^{i} \cos\theta_k, \tag{12}$$

which depends on multiple parameters but remains *fully factorized*:

$$\langle Z_i \rangle = f_0(\theta_0)\, f_1(\theta_1) \cdots f_i(\theta_i). \tag{13}$$

Key property:

- Cross terms exist only as simple products.
- Each parameter contributes independently.
- No mixing between different qubits or layers.

With data re-uploading, the same features are encoded multiple times using non-commuting rotations and re-entangling layers. The resulting expectation values become

$$\langle Z_i \rangle = \sum_{\alpha} c_\alpha \prod_{(k,\ell)\in\alpha} \{\sin, \cos\}\left(\tilde{x}_k, \theta_k^{(\ell)}\right), \tag{14}$$

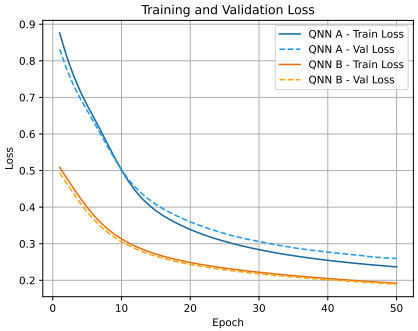
which *cannot* be written as a product of single-qubit functions.

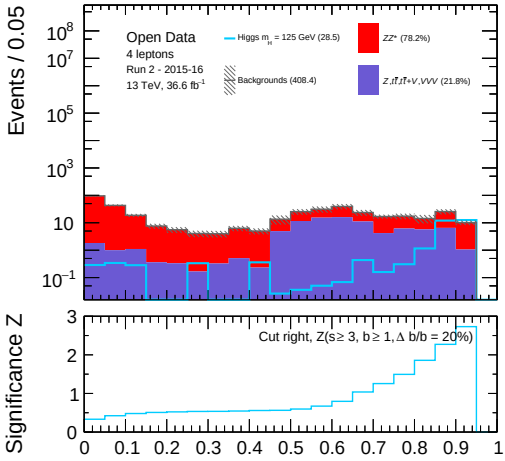
Result:

- Non-separable feature–parameter interactions
- Genuine multi-qubit correlations
- Increased expressive power of the PQC

Output score with data re-uploading

Learning rate = 1e-3





Correlator-Based Readout

While single-qubit observables $\mathcal{O} = \{\langle Z_i \rangle\}$ capture only local responses, entangling layers generate correlations that are only indirectly accessible through individual measurements.

To explicitly probe these correlations, we can replace the local readout by a set of two-qubit correlators aligned with the entanglement structure,

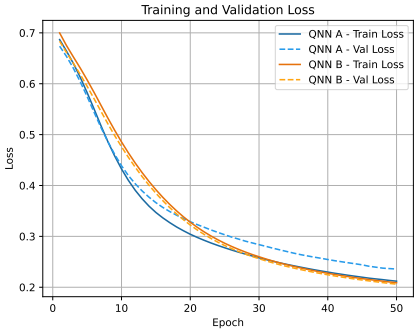
$$\mathcal{O} = \{\langle Z_0 Z_1 \rangle,\ \langle Z_1 Z_2 \rangle,\ \langle Z_2 Z_3 \rangle,\ \langle Z_3 Z_4 \rangle,\ \langle Z_4 \rangle\}. \tag{15}$$

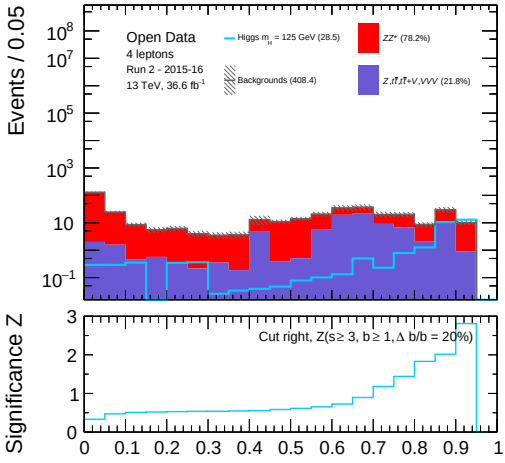
Each correlator $\langle Z_i Z_{i+1} \rangle$ is sensitive to joint feature–parameter interactions mediated by the CNOT gates, providing access to higher-order information without increasing the output dimension.

This readout enhances sensitivity to entangled feature representations while preserving the same classical interface to the downstream neural network.

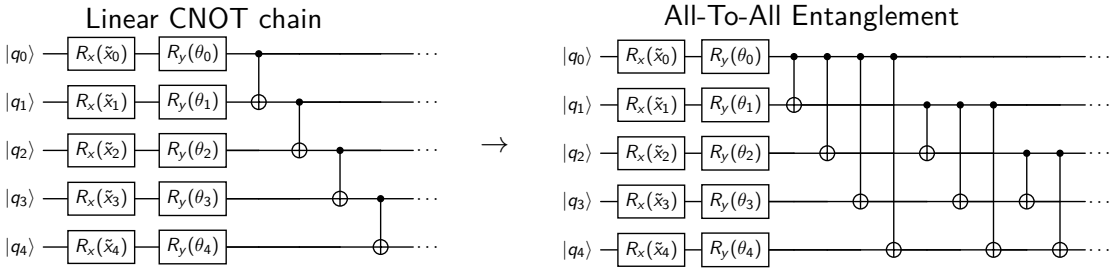
Output score with multi-qubit correlators

Learning rate = 1e-3





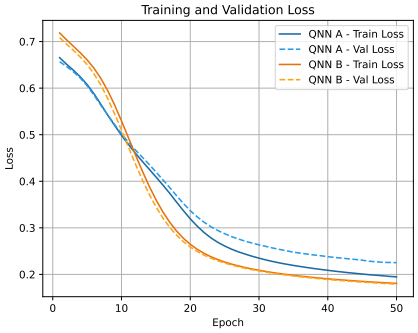
All-To-All Entanglement

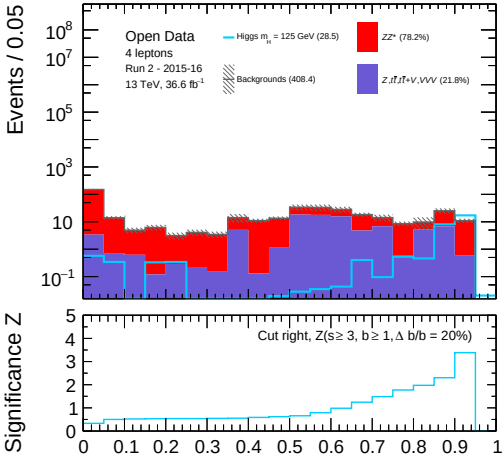


All-To-All Entanglement generates richer Pauli string growth and faster mixing than a linear chain. It approaches the universal random circuits used in expressibility studies, where dense entanglement increases expressivity with fewer layers.

Output score with all-to-all entanglement

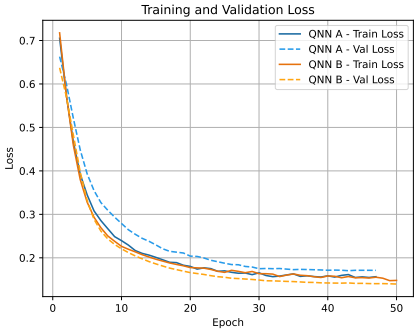
Learning rate = 1e-3



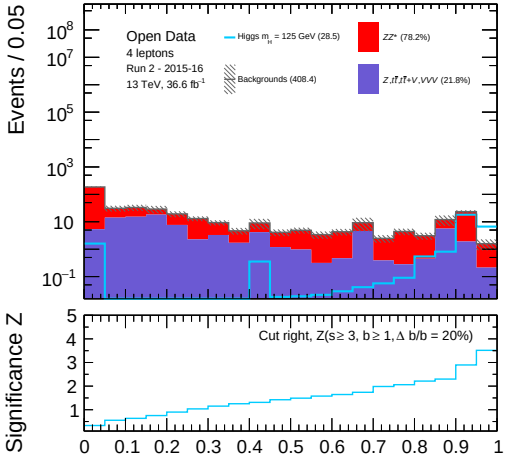


Output score with all-to-all entanglement

Learning rate = 1e-3
Batch size 128 → 64

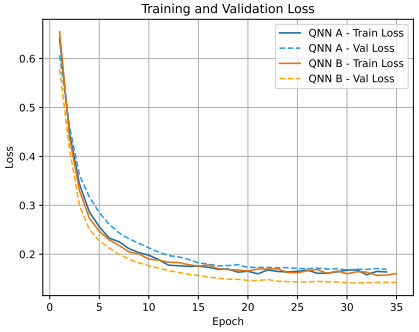


Faster convergence in terms of epochs

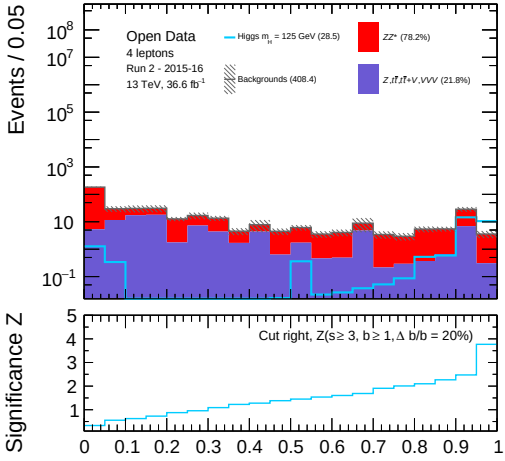


Output score with all-to-all entanglement

Learning rate = 1e-3
Batch size 64 → 32

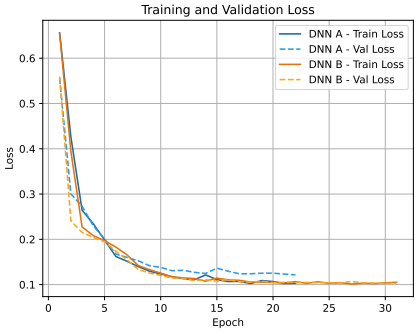


Faster convergence in terms of epochs

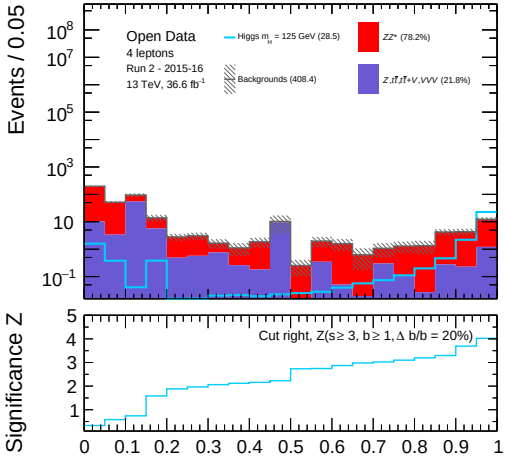


Output score for a classical DNN

Learning rate = 1e-3
Batch size 128



3 fully connected layers, 100 neurons each



Summary

To find the Higgs boson with QML, keep in mind:

- A linear PCQ (rotations + sequence of CNOT gates) allows to learn how to distinguish the signal from the background, but has limited expressivity
- Data re-uploading allows for a much faster decrease of the losses, giving better sensitivity with the same number of epochs
- All-To-All Entanglement improves the expressivity and gives improved results

Similar significance for finding the Higgs boson with QML with respect to classical ML. QML performance also depends on the signal model considered, the chosen set of input features, and the degree of intrinsic correlation among these features.

Thank you!

Questions?

Backup

Backup

References I

[1] ATLAS Collaboration. *ATLAS releases 2025 beta open data for education and outreach*. 2025. DOI: 10.7483/OPENDATA.ATLAS.B5M9.44TN.

[2] Guilherme I. Correr et al. "Optimal Complexity of Parameterized Quantum Circuits". In: *Entropy* 28.1 (2026), p. 73. DOI: 10.3390/e28010073. URL: https://doi.org/10.3390/e28010073.

[3] Michael Broughton et al. "TensorFlow Quantum: A Software Framework for Quantum Machine Learning". In: *arXiv:2003.02989* (2020). URL: https://arxiv.org/abs/2003.02989.

[4] Cirq Developers. *Cirq*. 2020. DOI: 10.5281/zenodo.4062499. URL: https://zenodo.org/doi/10.5281/zenodo.4062499.

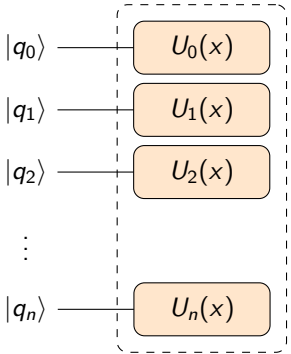
[5] Sukin Sim, Peter D. Johnson, and Alán Aspuru-Guzik. "Expressibility and entangling capability of parameterized quantum circuits". In: *Advanced Quantum Technologies* 2.12 (2019), p. 1900070. DOI: 10.1002/qute.201900070.

References II

[6] Maria Schuld, Ryan Sweke, and Johannes Jakob Meyer. "Effectively trainable quantum neural networks". In: *Physical Review A* 103.3 (2021), p. 032430. DOI: 10.1103/PhysRevA.103.032430.

[7] Adrián Pérez-Salinas et al. "Data re-uploading for a universal quantum classifier". In: *Quantum* 4 (2020), p. 226. DOI: 10.22331/q-2020-02-06-226.

QNN I

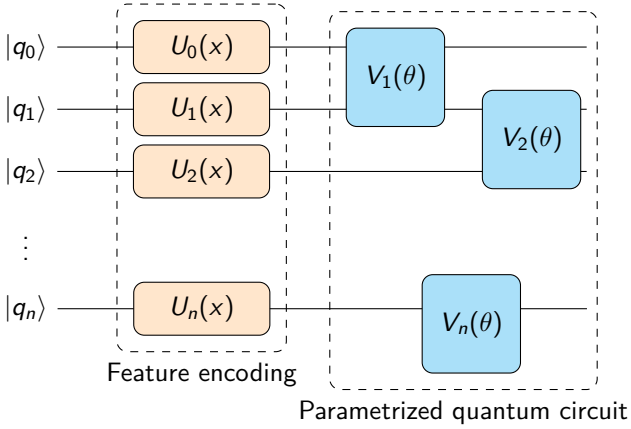


Feature encoding

The QNN is a QML algorithm that leverages Parameterized Quantum Circuits (PQCs) to perform tasks analogous to those in classical neural networks.

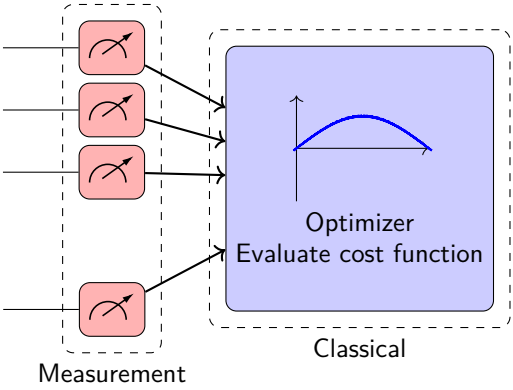
1. Input features are encoded into the quantum states of a set of qubits.

QNN II



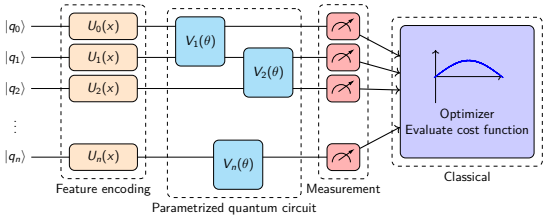
2 A PQC, also known as Variational Quantum Circuit, composed of a sequence of parameterized and trainable quantum gates, transforms the quantum states.

QNN III



3. The final quantum state is measured to extract classical information, which can be used to make predictions or for further optimization steps.

QNN IV



The quantum states exploit unique quantum phenomena

- superposition (which allows qubits to represent multiple states simultaneously);
- entanglement (which enables complex correlations between qubits).

By operating in the Hilbert space and enabling non-classical correlations, QNNs could enable more efficient modelling of complex patterns than their classical counterparts.

Open Data Samples

Following jupyter notebook

```
defs = {
    r'Data': {'dids':['data']},
    r'Background $Z,t\bar{t},t\bar{t}+V,VVV$': {'dids':
        [410470, 410155, 410218, 410219, 412043, 364243,
        364242, 364246,364248, 700320, 700321, 700322,
        700323,700324,700325], 'color': "#6b59d3" }, # purple
    r'Background $ZZ^{*}$': {'dids': [700600],'color': "#
        ff0000" },# red
    r'Signal ($m_H$ = 125 GeV)': {'dids': [345060, 346228,
        346310, 346311, 346312, 346340, 346341, 346342],'color
        ': "#00cdff" },# light blue
}
```

Hyper-parameters setup

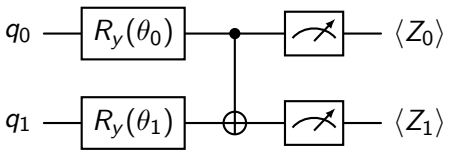
```
model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=1e-3),
    loss=tf.keras.losses.BinaryCrossentropy(),
    metrics=['accuracy']
)

callbacks = [
    tf.keras.callbacks.EarlyStopping(monitor='loss', patience
       =4, restore_best_weights=True),
    tf.keras.callbacks.ReduceLROnPlateau(monitor='loss',
       factor=0.5, patience=2)
]

model.fit(x_train, y_train, batch_size=128, epochs=50,
   verbose=2, validation_split=0.2, callbacks=callbacks)
```

Simplified PQC: 2-Qubit Circuit

Let us consider a simplified PQC with $N = 2$ qubits



The eigenstates of the Pauli-Z operator are

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{16}$$

The initial state is

$$|\psi_0\rangle = |00\rangle = |0\rangle \otimes |0\rangle \tag{17}$$

State After R_y Rotations

Recalling that $Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$, rotations R_y are given by

$$R_y(\theta) = e^{-i\frac{\theta}{2}Y} = \cos\frac{\theta}{2} I - i \sin\frac{\theta}{2} Y = \begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix} \tag{18}$$

After applying single-qubit rotations

$$|\psi_1\rangle = R_y(\theta_0)\,|0\rangle \otimes R_y(\theta_1)\,|0\rangle = \begin{bmatrix} \cos\frac{\theta_0}{2} \\ \sin\frac{\theta_0}{2} \end{bmatrix} \otimes \begin{bmatrix} \cos\frac{\theta_1}{2} \\ \sin\frac{\theta_1}{2} \end{bmatrix} \tag{19}$$

Expanding the tensor product

$$|\psi_1\rangle = \cos\frac{\theta_0}{2}\cos\frac{\theta_1}{2}\,|00\rangle + \cos\frac{\theta_0}{2}\sin\frac{\theta_1}{2}\,|01\rangle + \sin\frac{\theta_0}{2}\cos\frac{\theta_1}{2}\,|10\rangle + \sin\frac{\theta_0}{2}\sin\frac{\theta_1}{2}\,|11\rangle \tag{20}$$

State After CNOT

Applying the CNOT gate

$$|\psi_2\rangle = \cos\frac{\theta_0}{2}\cos\frac{\theta_1}{2}|00\rangle + \cos\frac{\theta_0}{2}\sin\frac{\theta_1}{2}|01\rangle + \sin\frac{\theta_0}{2}\sin\frac{\theta_1}{2}|10\rangle + \sin\frac{\theta_0}{2}\cos\frac{\theta_1}{2}|11\rangle \tag{21}$$

Pauli-Z operator

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad Z|0\rangle = |0\rangle, \quad Z|1\rangle = -|1\rangle \tag{22}$$

$$Z_0 = Z \otimes I, \quad Z_1 = I \otimes Z \tag{23}$$

Expectation Values

For the first qubit

$$\langle Z_0 \rangle = P(00) + P(01) - P(10) - P(11) = \cos^2 \frac{\theta_0}{2} - \sin^2 \frac{\theta_0}{2} = \cos\theta_0 \tag{24}$$

For the second qubit

$$\langle Z_1 \rangle = P(00) - P(01) + P(10) - P(11) = (\cos^2 \frac{\theta_1}{2} - \sin^2 \frac{\theta_1}{2})(\cos^2 \frac{\theta_0}{2} - \sin^2 \frac{\theta_0}{2}) = \cos\theta_0 \cos\theta_1 \tag{25}$$

For a PQC with N qubits

$$\langle Z_i \rangle = \prod_{k=0}^{i} \cos\theta_k, \quad i = 0, \ldots, N-1 \tag{26}$$

Each qubit expectation depends on all previous rotation angles multiplicatively.

Data re-uploading PQC

With data re-uploading, the circuit consists of L repeated layers,

$$U = \prod_{\ell=1}^{L} U_{\mathrm{CNOT}} \left(\bigotimes_{i=0}^{4} R_y(\theta_i^{(\ell)}) \right) \left(\bigotimes_{i=0}^{4} R_x(\tilde{x}_i) \right). \tag{27}$$

The same input features $\tilde{x}_i$ are encoded multiple times, while each layer has independent trainable parameters $\theta_i^{(\ell)}$. The measured observables remain

$$\langle Z_i \rangle = \langle 0|^{\otimes 5} U^\dagger Z_i U |0\rangle^{\otimes 5}. \tag{28}$$

Heisenberg evolution of Z_i (one layer)

We propagate the observable Z_i backwards through one re-uploading layer.
First, through the CNOT chain:

$$U_{\mathrm{CNOT}}^{\dagger} Z_i U_{\mathrm{CNOT}} = \prod_{k=0}^{i} Z_k. \tag{29}$$

Thus, after the entangling block, Z_i becomes a *multi-qubit Pauli string*.

Rotation of Pauli operators

Single-qubit rotations act nontrivially on Pauli operators.
For a data-encoding rotation,

$$R_x^\dagger(\tilde{x}) Z R_x(\tilde{x}) = Z\cos\tilde{x} + Y\sin\tilde{x}. \tag{30}$$

For a trainable rotation,

$$R_y^\dagger(\theta) Z R_y(\theta) = Z\cos\theta - X\sin\theta. \tag{31}$$

Therefore, when a Pauli string such as

$$\prod_{k=0}^{i} Z_k \tag{32}$$

is conjugated by R_x and R_y gates, it becomes a *linear combination of Pauli strings* involving Z, X, and Y operators.

Loss of factorization

After one re-uploading layer, the observable Z_i has evolved into

$$Z_i \longrightarrow \sum_{\alpha} c_\alpha(\tilde{\mathbf{x}}, \boldsymbol{\theta}) P_\alpha, \tag{33}$$

where P_α are multi-qubit Pauli strings.
After multiple layers, this expansion grows in complexity. As a result,

$$\langle Z_i \rangle = \langle 0|^{\otimes 5} \left(\sum_{\alpha} c_\alpha P_\alpha \right) |0\rangle^{\otimes 5}, \tag{34}$$

which no longer factorizes into single-qubit expectation values.

Effect of Correlator Readout

For a general observable O measured after a data re-uploading PQC, the expectation value can be written as

$$\langle O \rangle = \sum_{\alpha} c_{\alpha} \prod_{(k,\ell)\in\alpha} \{\sin, \cos\}\left(\tilde{x}_k, \theta_k^{(\ell)}\right), \tag{35}$$

where each term arises from the Heisenberg evolution of O through non-commuting data-encoding and entangling layers.

When $O = Z_i$, only Pauli strings ending on qubit i contribute, limiting sensitivity to marginal feature responses. In contrast, choosing a two-qubit correlator $O = Z_i Z_j$ generates Pauli strings with simultaneous support on both qubits,

$$U^{\dagger}(Z_i Z_j)U \longrightarrow \sum_{\beta} d_{\beta} P_i^{(\beta)} \otimes P_j^{(\beta)} \otimes \cdots . \tag{36}$$

All-to-All Entanglement and Pauli String Growth

Expectation values can be written in the Heisenberg picture as

$$\langle O \rangle = \langle 0 | \, U^\dagger O U \, | 0 \rangle \, . \tag{37}$$

Single-qubit rotations mix Pauli operators locally,

$$R_x^\dagger Z R_x = \cos x \, Z + \sin x \, Y, \tag{38}$$

while entangling gates spread operators across qubits. For a CNOT,

$$Z_c \mapsto Z_c, \qquad Z_t \mapsto Z_c Z_t. \tag{39}$$

Linear entanglement propagates Pauli operators sequentially along the chain, requiring depth $O(N)$ to generate global correlations.

All-to-all entanglement spreads Pauli operators globally in a single layer, rapidly generating high-weight Pauli strings and faster mixing of observables.

Universal Random Circuits and Expressibility

Universal random circuits consist of alternating layers of

- random single-qubit rotations, and
- dense (often all-to-all) entanglement.

A typical structure is

$$U = \prod_{\ell=1}^{L} \left(U_{\text{ent}}^{(\ell)} \bigotimes_{i=1}^{N} R_i^{(\ell)} \right). \tag{40}$$

Such circuits approximate the statistics of Haar-random unitaries with relatively small depth. Their *expressibility* is measured by how closely the induced state distribution matches Haar-random statistics.

Dense entanglement achieves high expressibility with fewer layers compared to local entanglement patterns.

The Haar Measure and Quantum Correlations

The Haar measure is a uniform probability measure on the unitary group $SU(2^N)$ that is connected to the notion of expressibility. Sampling from the Haar measure implies that

- no unitary is preferred over another,
- all quantum correlations are represented without bias.

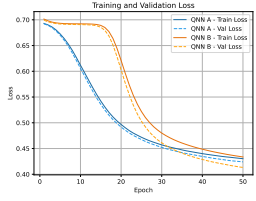
A PQC that approximates Haar-random unitaries can, in principle,

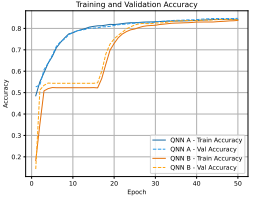
- generates arbitrary entangled states,
- represents highly nonlocal and high-order quantum correlations,
- but gradients concentrate exponentially (barren plateaus).

In general, however, the goal is not full Haar randomness, but structured expressibility, achieved by combining data re-uploading, limited-depth dense entanglement, selective readout observables (e.g. $\langle Z_i Z_j \rangle$).

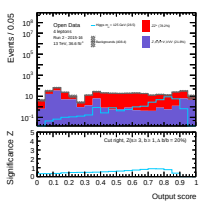
Performance with the linear PQC in the QNN

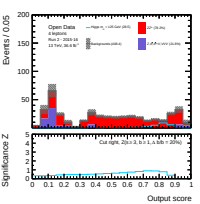
QNN



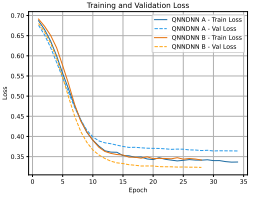


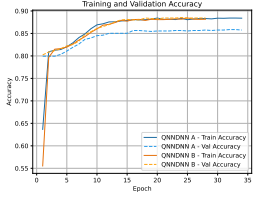
QNN



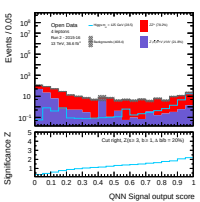


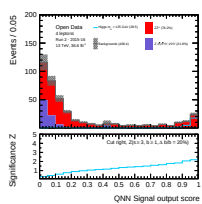
QNN+DNN





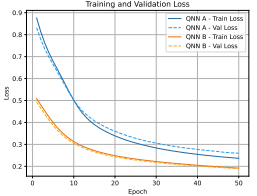
QNN+DNN

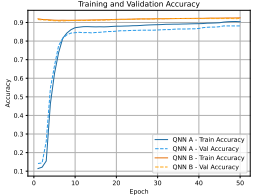




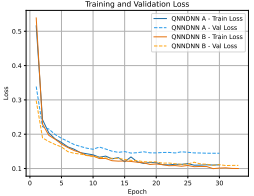
Performance with the data re-uploading PQC in the QNN

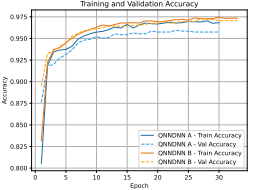
QNN



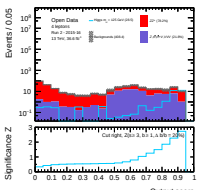


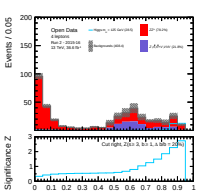
QNN+DNN



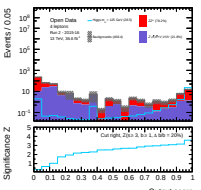


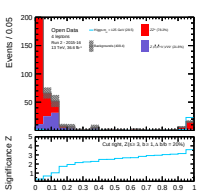
QNN





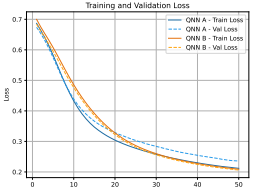
QNN+DNN

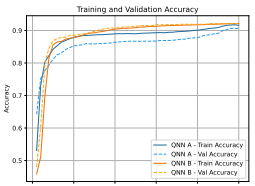




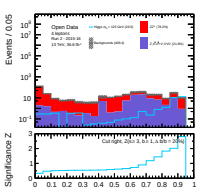
Performance with the correlator readout PQC in the QNN

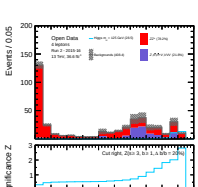
QNN



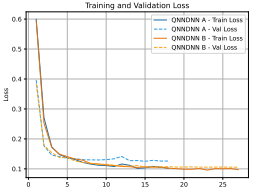


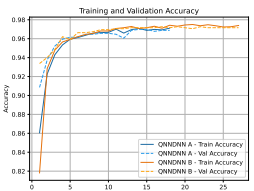
QNN



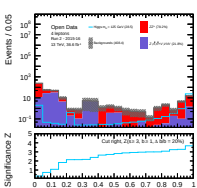


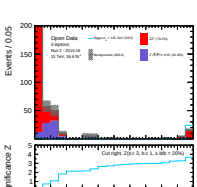
QNN+DNN



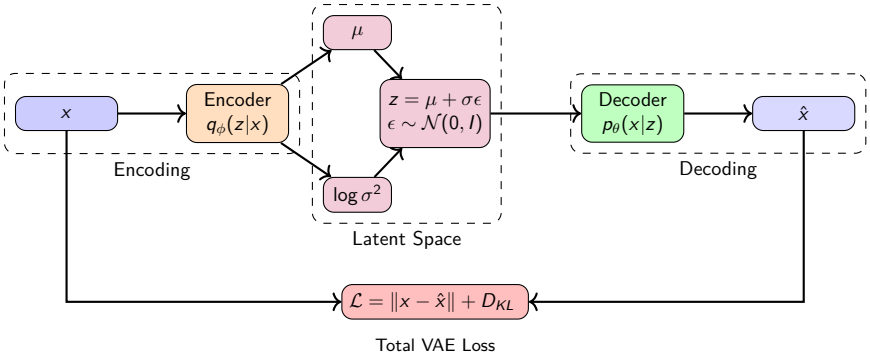


QNN+DNN

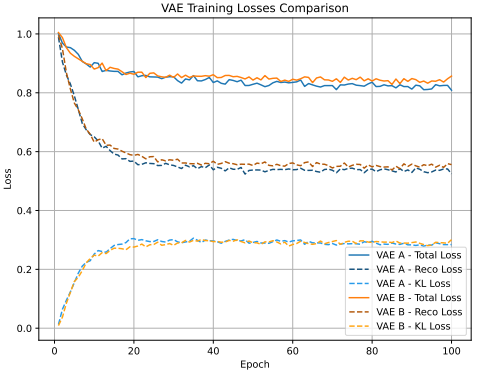


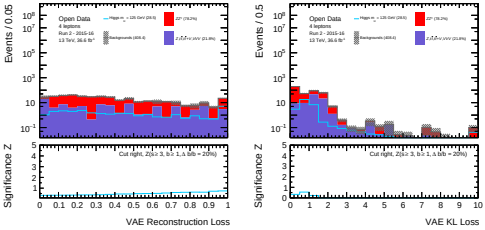


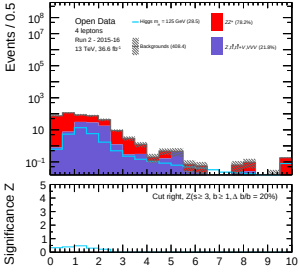
VAE


x
Encoder
$q_\phi(z|x)$
Encoding
μ
$z = \mu + \sigma\epsilon$
$\epsilon \sim \mathcal{N}(0, I)$
$\log \sigma^2$
Latent Space
Decoder
$p_\theta(x|z)$
$\hat{x}$
Decoding
$\mathcal{L} = \|x - \hat{x}\| + D_{KL}$
Total VAE Loss

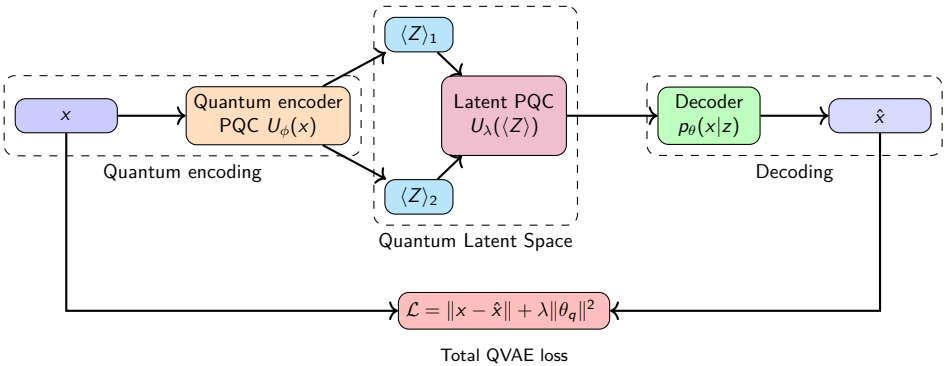
VAE


VAE Training Losses Comparison
Loss
Epoch
VAE A - Total Loss
VAE A - Reco Loss
VAE A - KL Loss
VAE B - Total Loss
VAE B - Reco Loss
VAE B - KL Loss

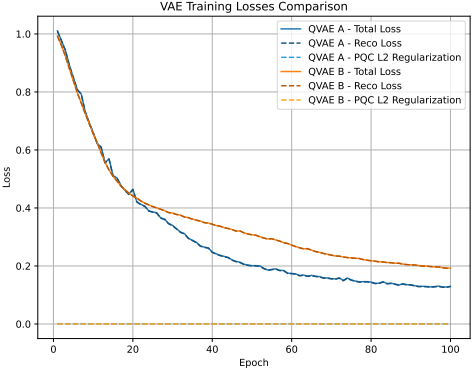

Events / 0.05
Open Data
4 leptons
Run 2 - 2015-16
13 TeV, 36.6 fb⁻¹
Significance Z
VAE Reconstruction Loss
Events / 0.5
VAE KL Loss

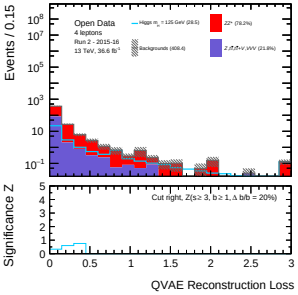

Events / 0.5
Open Data
4 leptons
Run 2 - 2015-16
13 TeV, 36.6 fb⁻¹
Significance Z
Total VAE Loss

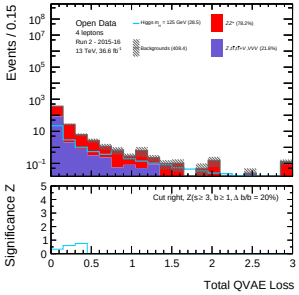
QVAE


x
Quantum encoder
PQC $U_\phi(x)$
Quantum encoding
$\langle Z \rangle_1$
$\langle Z \rangle_2$
Latent PQC
$U_\lambda(\langle Z \rangle)$
Quantum Latent Space
Decoder
$p_\theta(x|z)$
$\hat{x}$
Decoding
$\mathcal{L} = \|x - \hat{x}\| + \lambda \|\theta_q\|^2$
Total QVAE loss

QVAE


VAE Training Losses Comparison
QVAE A - Total Loss
QVAE A - Reco Loss
QVAE A - PQC L2 Regularization
QVAE B - Total Loss
QVAE B - Reco Loss
QVAE B - PQC L2 Regularization
Loss
Epoch
1.0
0.8
0.6
0.4
0.2
0.0
0
20
40
60
80
100


Open Data
4 leptons
Run 2 - 2015-16
13 TeV, 36.6 fb-1
Backgrounds (406.4)
Events / 0.15
Significance Z
Cut right, Z(s≥ 3, b≥ 1, Δ b/b = 20%)
QVAE Reconstruction Loss


Open Data
4 leptons
Run 2 - 2015-16
13 TeV, 36.6 fb-1
Backgrounds (406.4)
Events / 0.15
Significance Z
Cut right, Z(s≥ 3, b≥ 1, Δ b/b = 20%)
Total QVAE Loss