Hybrid Quantum-Classical Graph Neural Networks for Track Reconstruction

Cenk Tüysüz[1], Carla Rieger[2]

[1]Middle East Technical University, Ankara, Turkey

[2]ETH Zürich, Zürich, Switzerland

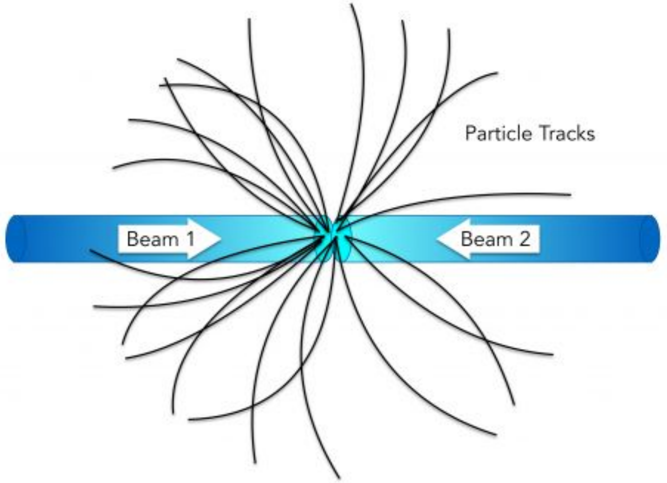
CERN openlab Technical Workshop, 11.03.2021

Outline

- Particle track reconstruction problem
- Quantum Computing and Machine Learning
- Hybrid Embeddings for Particle Tracking
- Hybrid GNN approach
- Comments on future improvements

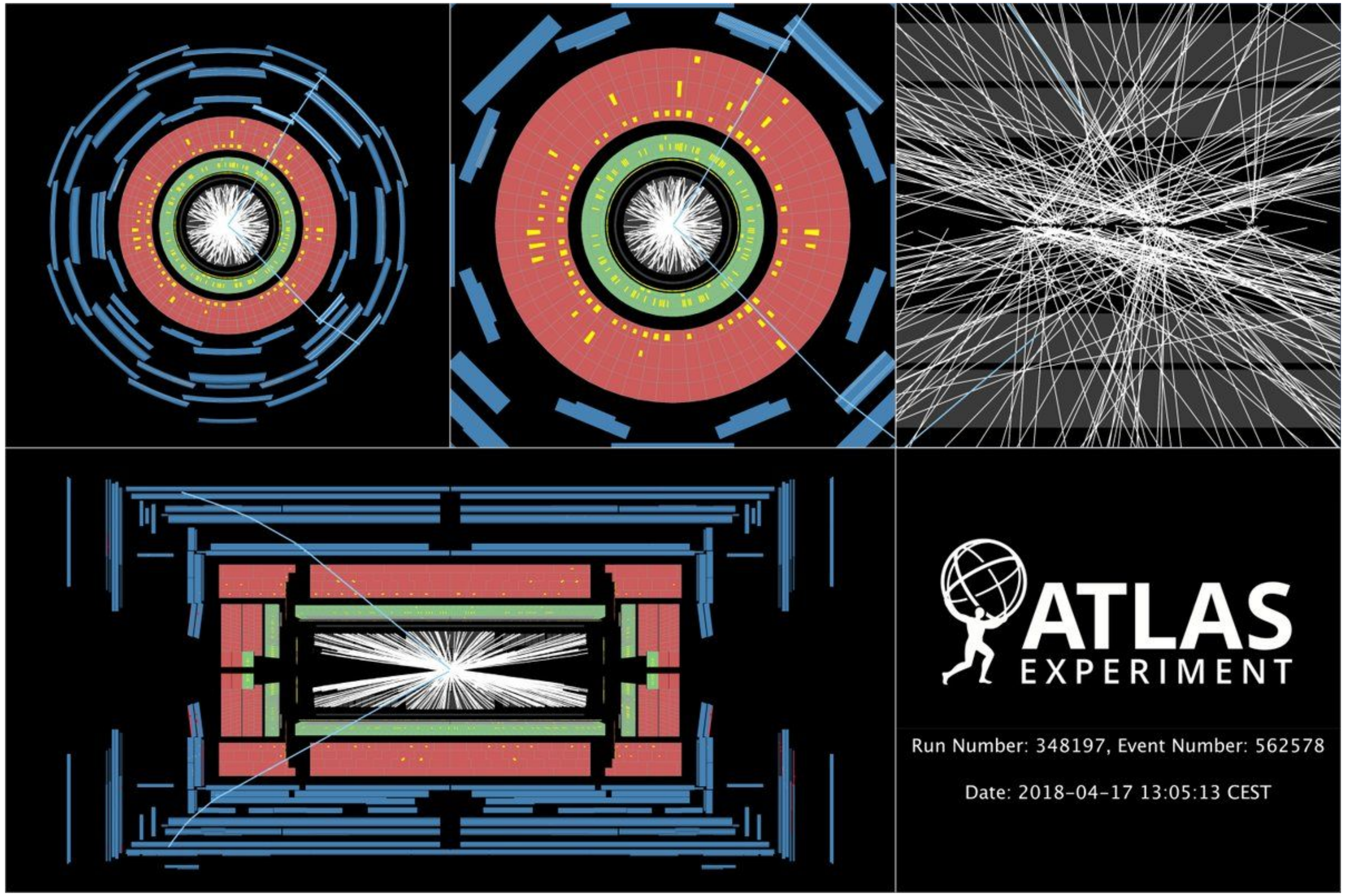
Large Hadron Collider (LHC)

and particle track reconstruction



https://atlas.cern/updates/atlas-news/counting-collisions

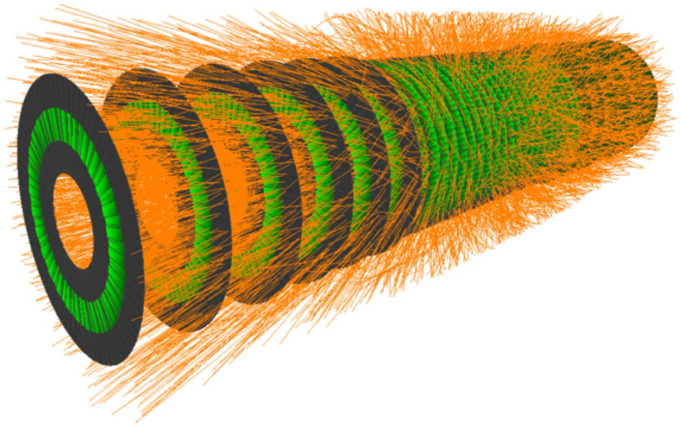
An event view from ATLAS Experiment

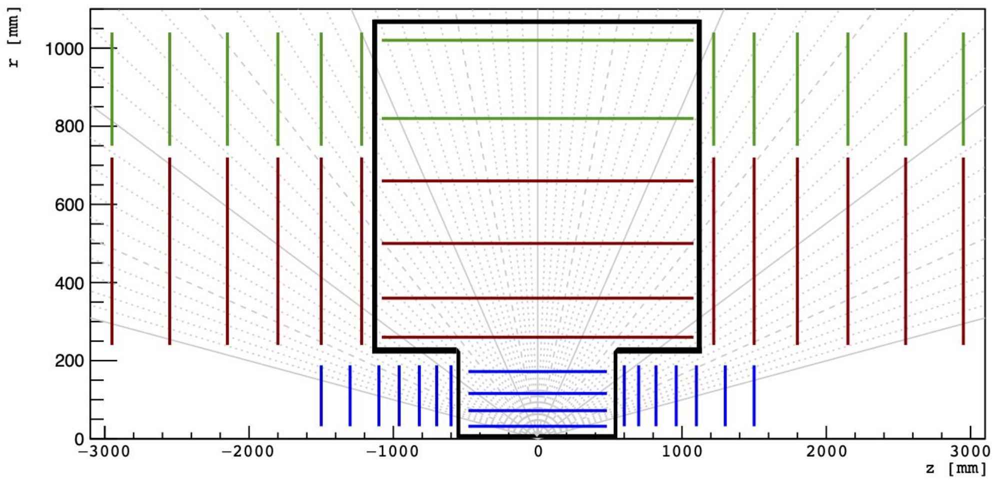


https://cds.cern.ch/record/2315786

TrackML Dataset

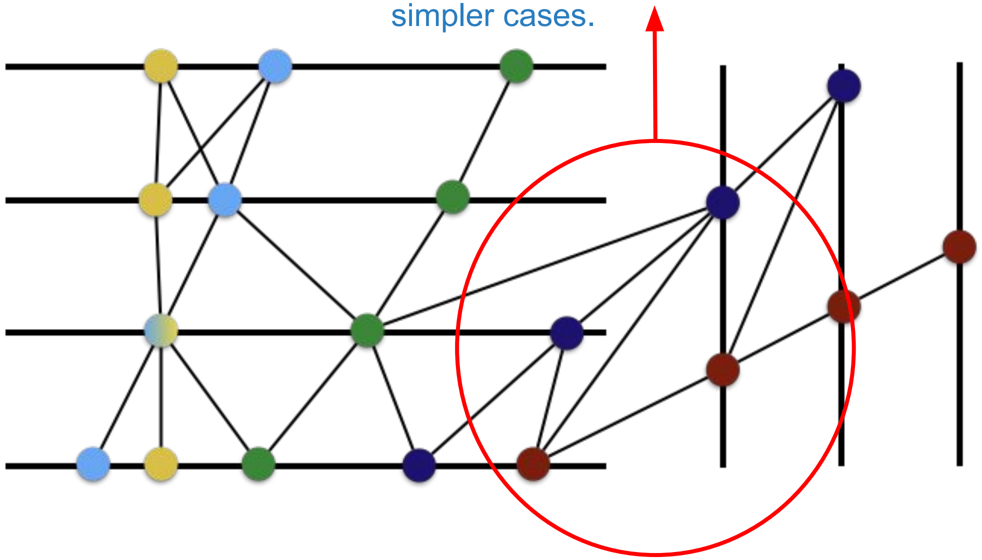
https://www.kaggle.com/c/trackml-particle-identification/overview





Contains: 10k collision events (200 soft QCD interactions)
(arXiv: 1904.06778)

endcaps produce a lot of ambiguity and therefore many track candidates, we omit endcaps as we want to limit our model to simpler cases.

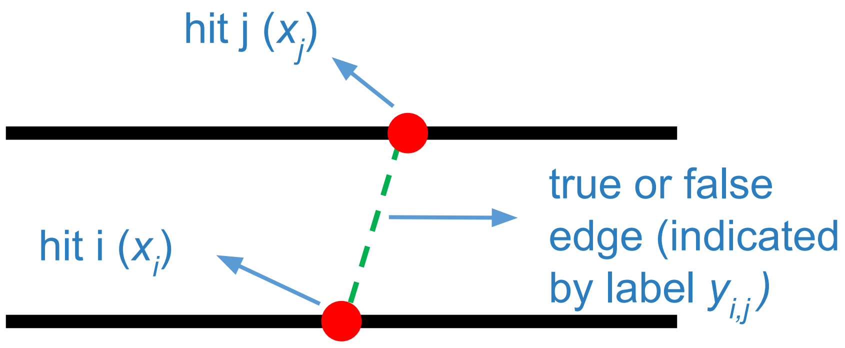


Retrieved from: Farrell et al. 2018 (arXiv: 1810.06111)

Learning the embedding of the hit data set

The data processing pipeline

First Preprocessing Steps of the data set → Embedding of pairs of hits (doublets): $\Phi(x_i, \theta)$ in 4D embedding space → Graph construction and classification with (quantum) GNN



A doublet in original space.

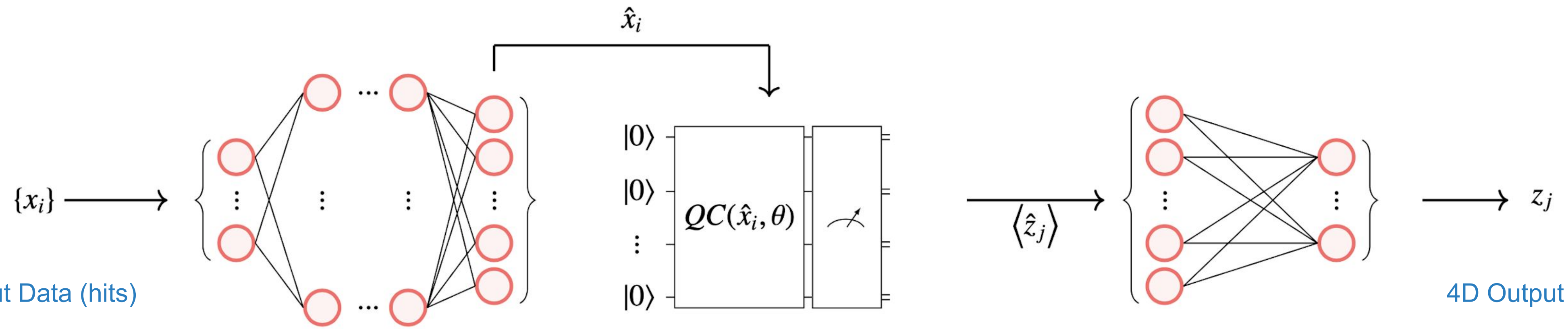
Hinge Embeddig Loss:

$$s_n = (x_i, x_j, y_{i,j})$$

$$loss(s_n) = \begin{cases} \max\{0, \|\Phi(x_i,\theta) - \Phi(x_j,\theta)\|_2\}, & \text{if } (x_i, x_j) \text{ belong to the same trajectory} \\ 0, & \text{if } (x_i, x_j) \text{ belong to different trajectories.} \end{cases}$$

Similar to: Choma et al. 2020(arXiv: 2007.00149)

Hybrid Neural Network architecture



3D Input Data (hits)

Classical MLP
(hidden layers: n_layers x 512)

Quantum Circuit

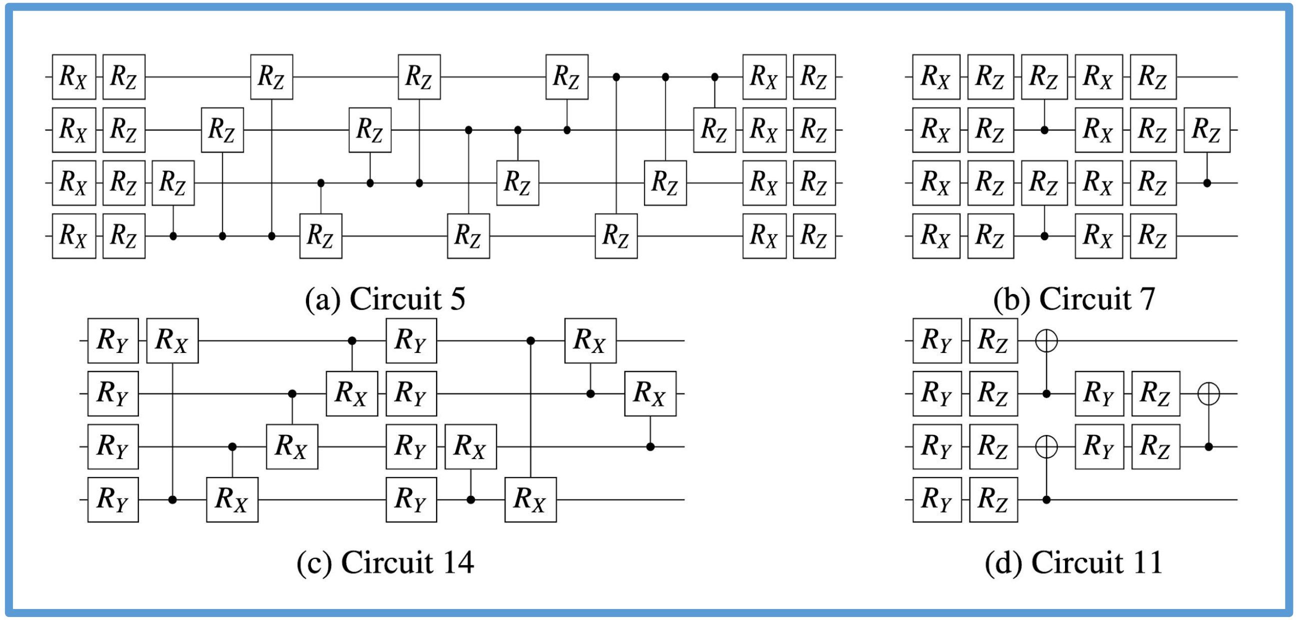
Fully Connected Layer

4D Output Data in Embedded Space

General model of the hybrid architecture

MLP adapted from: Choma et al. 2020(arXiv: 2007.00149)

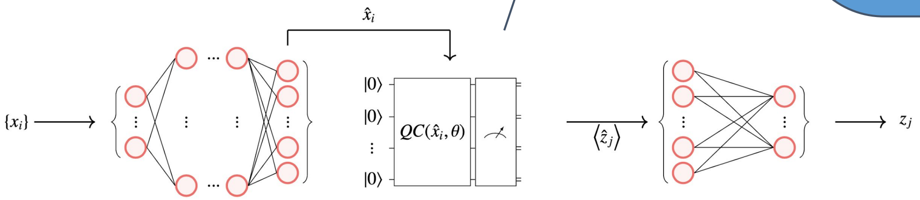
Quantum Circuit Approach



(a) Circuit 5 (b) Circuit 7 (c) Circuit 14 (d) Circuit 11

- those quantum circuits were chosen due to their different values regarding entanglement and expressibility
- they act as an encoding function, each of the rotational gates exhibits a free parameter

$$QC_{id} : \mathbb{R}^{n_{parameters}} \longrightarrow \mathbb{R}^{n_{measurements}}$$

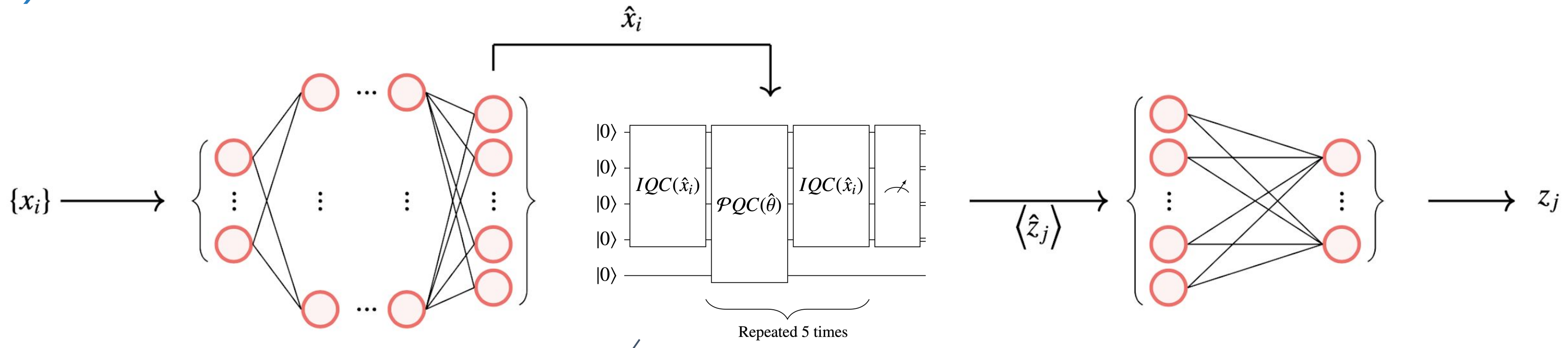


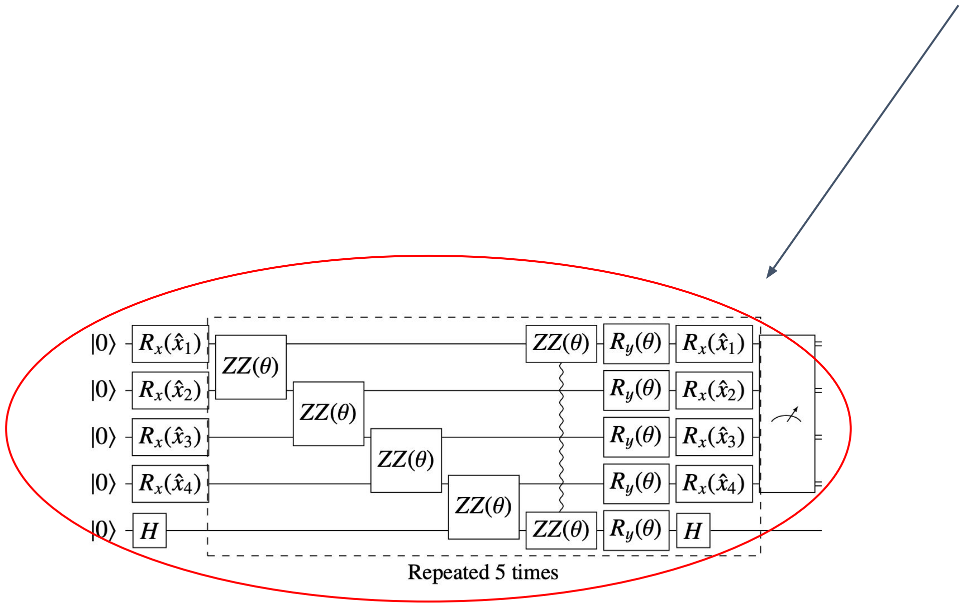
Circuit	Parameters $n_{parameters}$	Entanglement (the higher the better)	Expressibility (the lower the better)
5	28	0.290	0.051
7	19	0.212	0.104
11	12	0.538	0.139
14	16	0.545	0.011

Circuits adapted from and values calculated as in: Sim et al. 2019 (arXiv:1905.10876)

Quantum Feature Map Approach

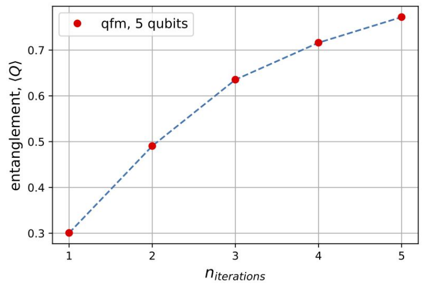
(QFM)

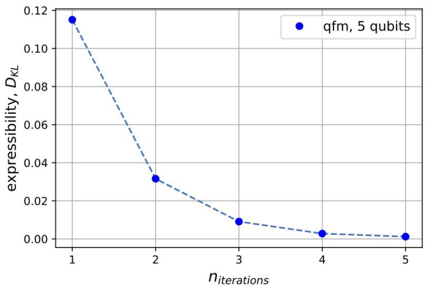




Adapted from: Lloyd et al. 2020 (arXiv:2001.03622)

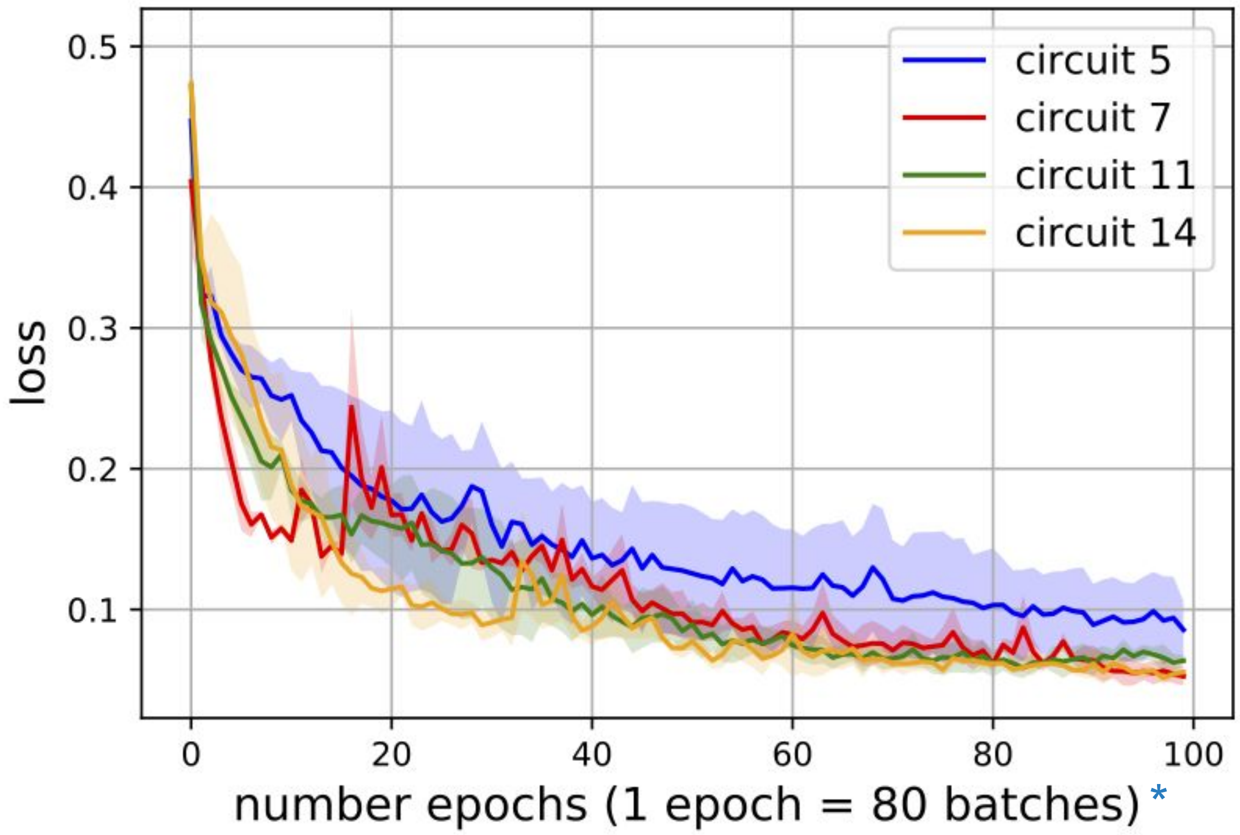
- QFM iteratively encodes the input x_i
- additionally, there are optimizable parameters θ included within the quantum circuit
- repeating blocks of the circuit as indicated on the left increases the entanglement and decreases the expressibility value, which is preferable



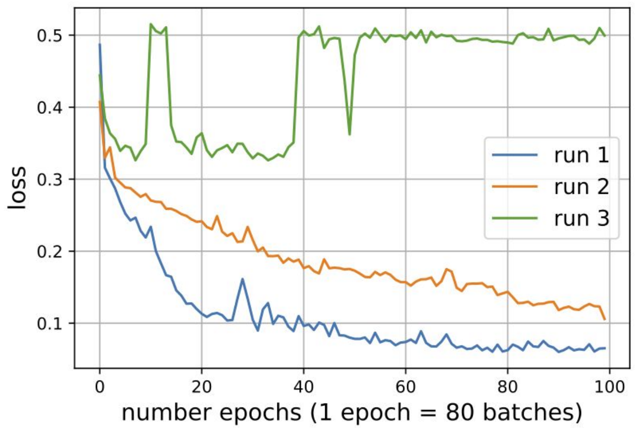


Training results

Quantum Circuit Approach



*



- training time increases with increasing number of gates in the quantum circuit
- observation of plateaus in training/validation loss for circuit 5, which includes the highest number of QC parameters in this test

Circuit	Parameters $n_{parameters}$	Entanglement (the higher the better)	Expressibility (the lower the better)	Training time (average per batch)
5	28	0.290	0.051	$37 \pm 8s$
7	19	0.212	0.104	$20 \pm 4s$
11	12	0.538	0.139	$14 \pm 4s$
14	16	0.545	0.011	$16 \pm 4s$

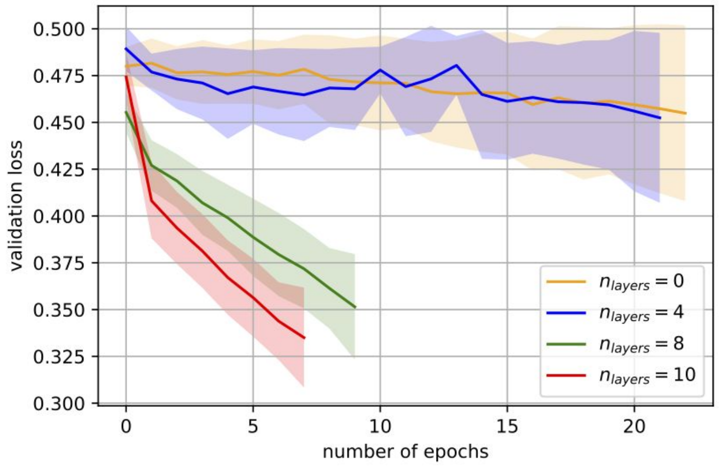
Training data set: 8k hits, validation data set: 2k hits, using ADAMAX optimizer, n_layers = 10, hinge embedding loss, lr = 1e-2.

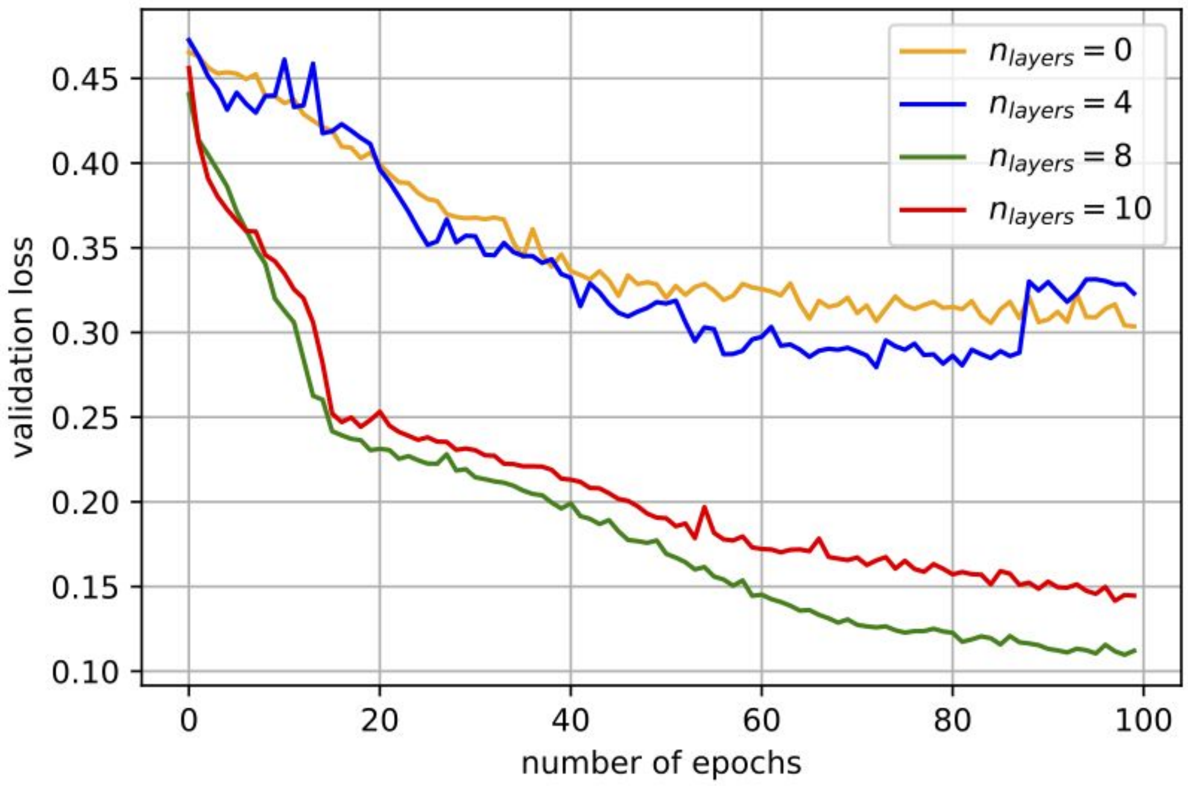
* plot without circuit 5 run 3 for better visualization

Ent./Expr. values calculated as in: Sim et al. 2019 (arXiv:1905.10876)

Training results

QFM Approach





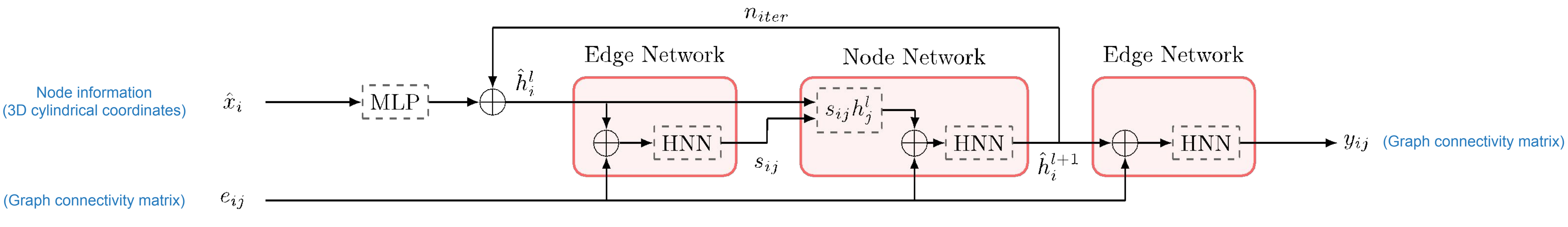
- learning rate has to be lowered using this architecture by factor 0.1
- similar performance of 0 and 4 layer version, as well as for 8 and 10 classical layers
- validation loss converges to high validation loss for low number of layers
- std much higher when training with less classical layers
- possibility for better convergence when training for more than 100 epochs (especially for 8 and 10 layers)

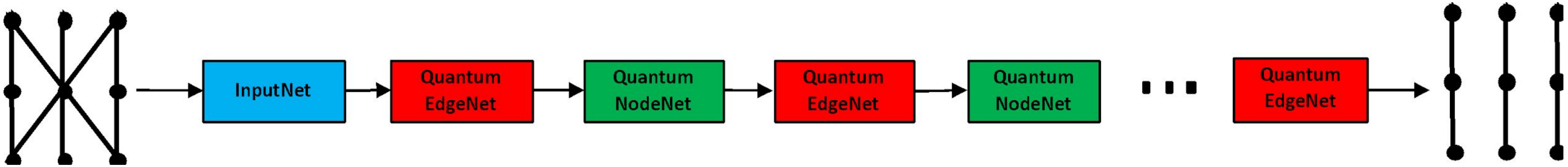
Circuit	Parameters $n_{parameters}$	Entanglement (higher value preferred)	Expressibility (lower value preferred)	Training time (average per batch)
QFM (5 qubits) ($n_{iteration} = 5$)	74	0.772	0.001	$5min38 \pm 8s$
14 (4 qubits) ($n_{iteration} = 1$)	16	0.545	0.011	$16 \pm 4s$

Training data set: 8k hits, validation data set: 2k hits, using ADAMAX optimizer, hinge embedding loss, lr = 1e-3.

Ent./Expr. values calculated as in: Sim et al. 2019 (arXiv:1905.10876)

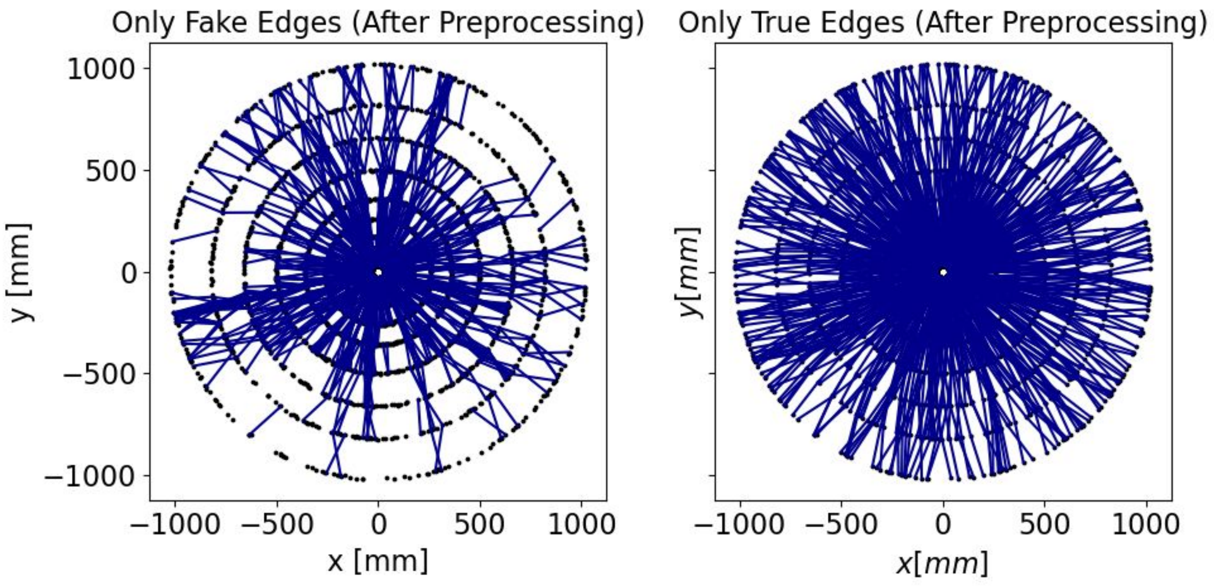
Quantum Graph Neural Network



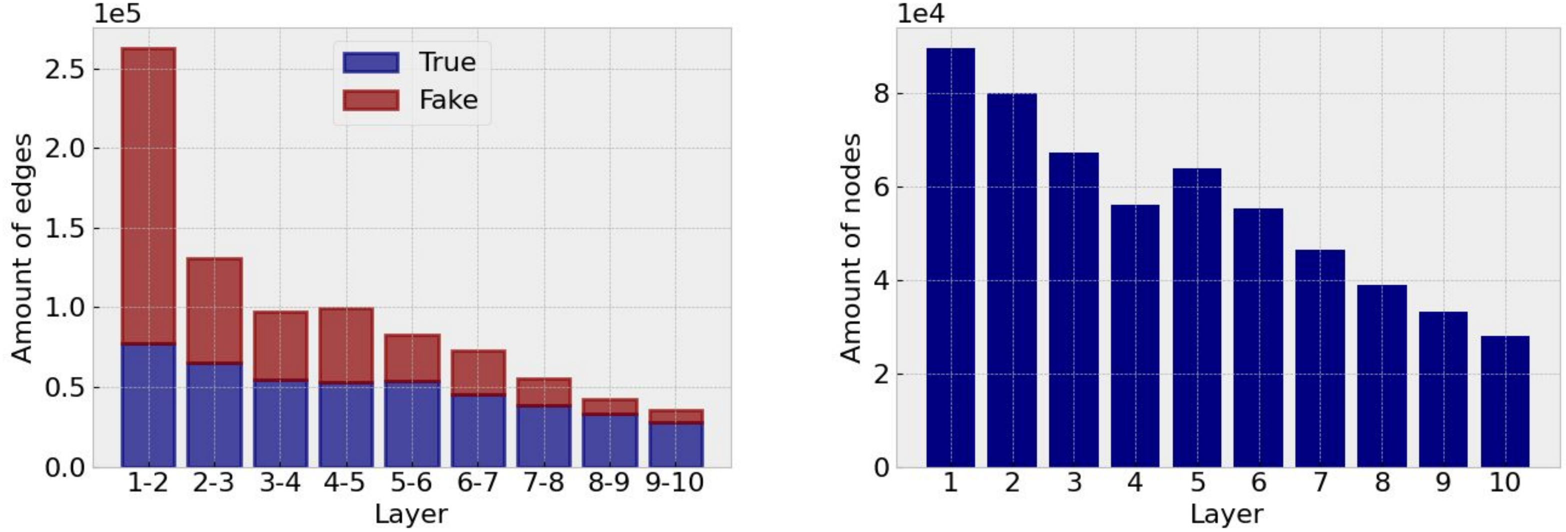


Preprocessing

After preprocessing (100 events):
Total edges: 880k (true: 450k, fake: 430k)
Total nodes: 560k
edges per graph: 8783.7 +/- 1877.3
nodes per graph: 5583.1 +/- 804.4



Distribution of 100 graphs

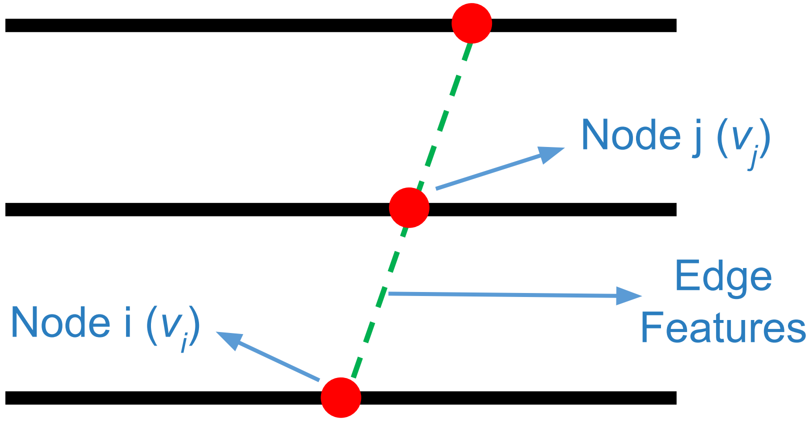


Edge Network

Edge Network:

Input: Node information of each edge

Learn edge features



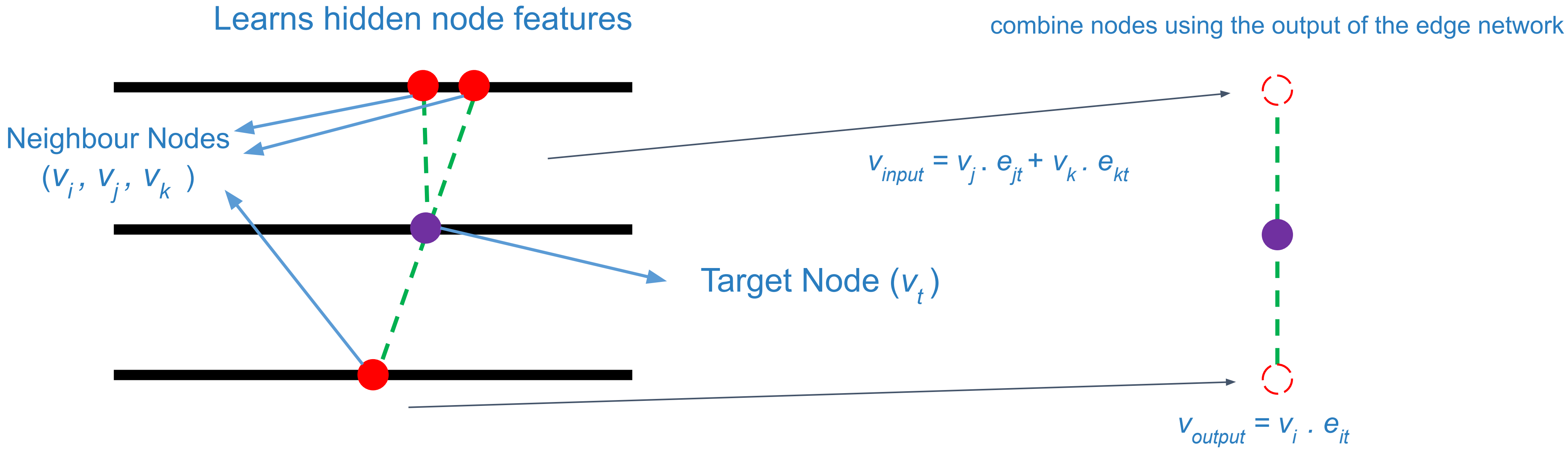
$v_i = [r_i, \phi_i, z_i, h_i^1, \ldots, h_i^N]$

$v_j = [r_j, \phi_j, z_j, h_j^1, \ldots, h_j^N]$

N = hidden dimension size

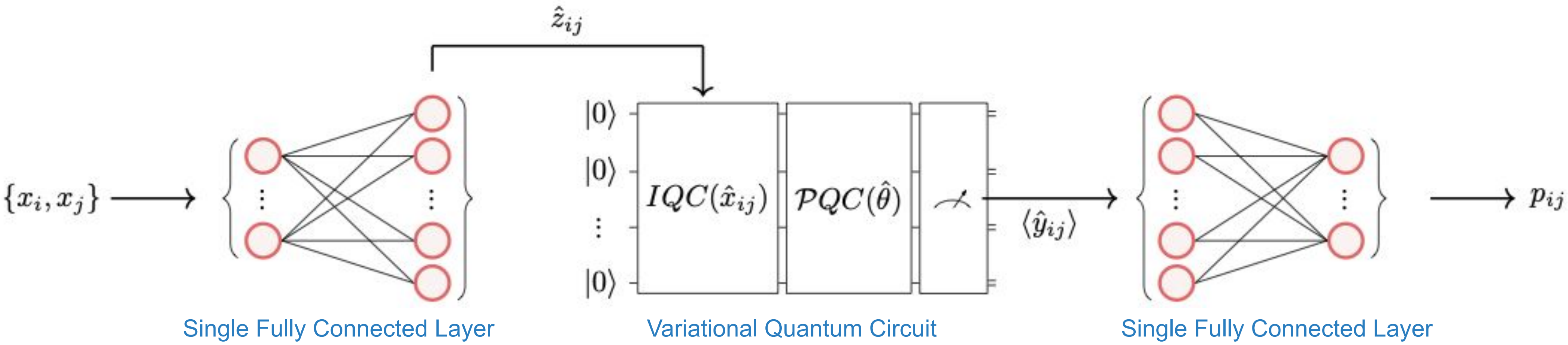
Node Network

Node Network:
Input: Triplet node information
Learns hidden node features



$[v_{input}, v_{output}, v_{target}]$ → Node Network → h_{target} *(update hidden node features)*

Hybrid Neural Network



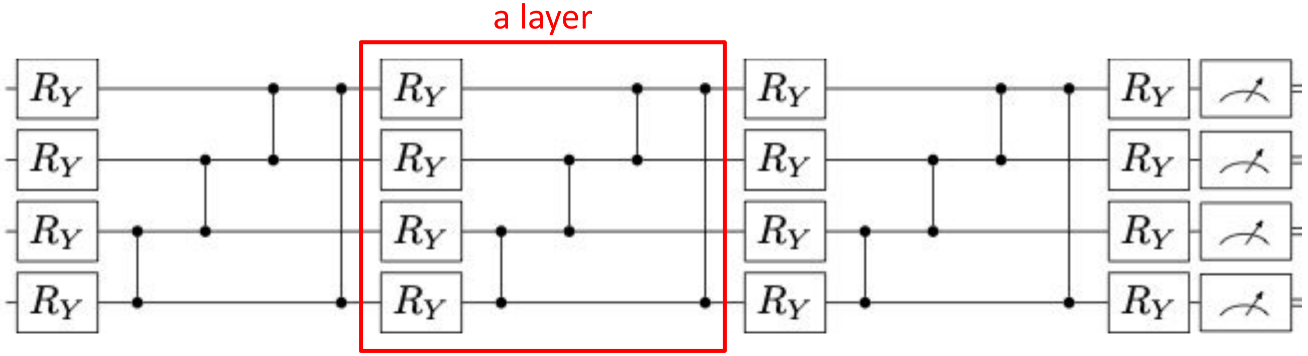
IQC (Information Encoding Quantum Circuit): Encodes the Classical Information to Quantum States

PQC (Parametrized Quantum Circuit): Contains trainable parameters that does operations to the Quantum States on the Hilbert Space

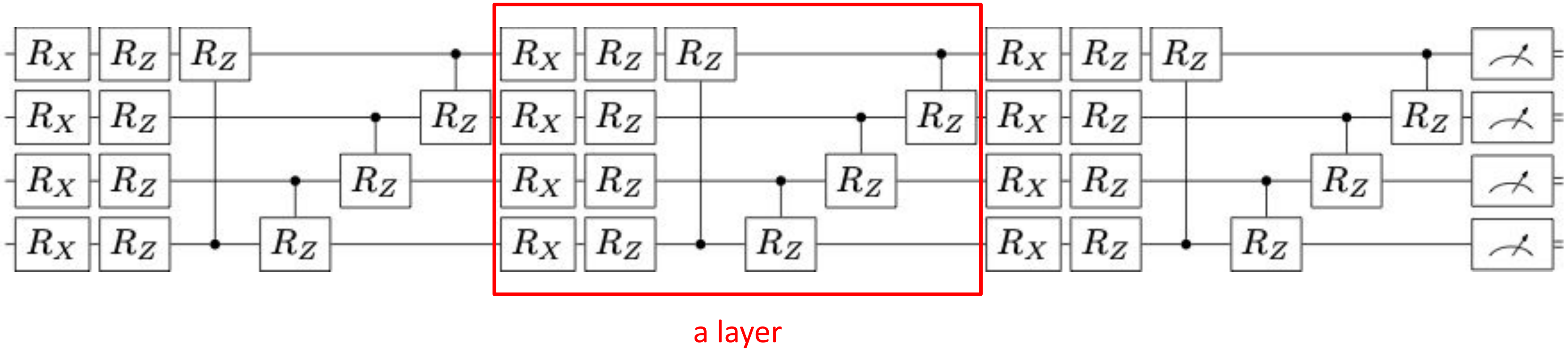
Parametrized Quantum Circuits

Circuits are taken from (Sim et al. 2019, arXiv:1905.10876)

Circuit 10:



Circuit 19:

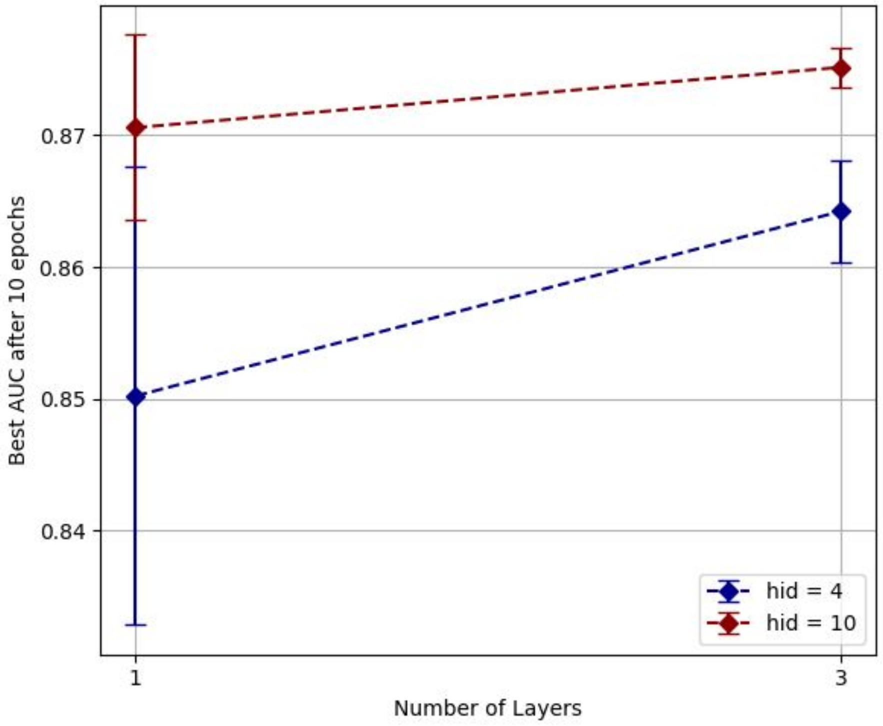


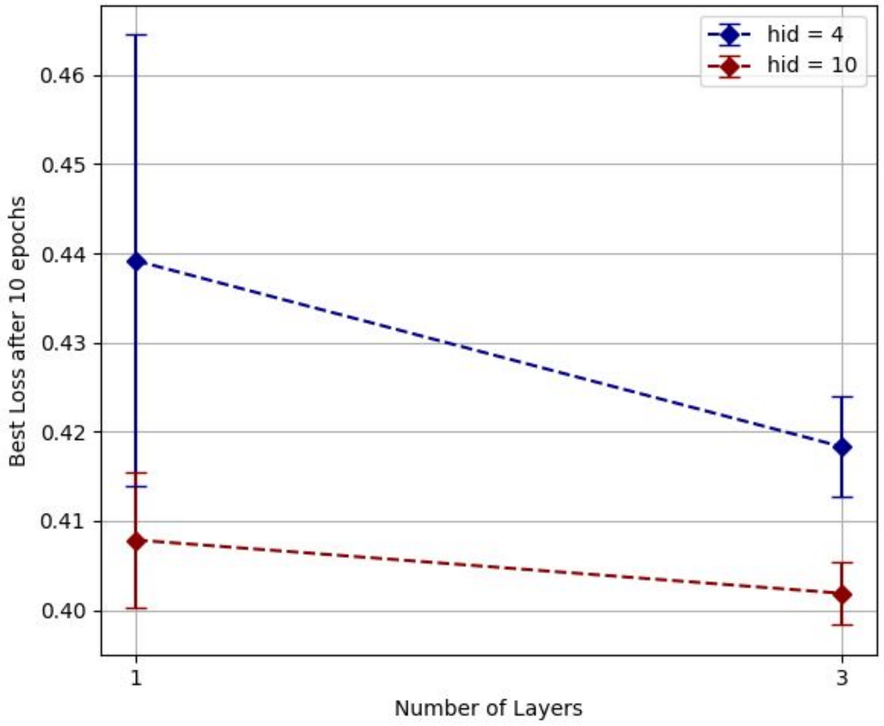
Layers are repeated blocks of Quantum Circuits.
(They can have the same or different parameters)

AUC: Area Under ROC, a measure of accuracy for different thresholds.
AUC = 1.0 means perfect score.

Training Results

Number of Layers ($N_{iteration}$ = 3, qc = 19, N_{hid} = 4, N_{qubits} = 4)





N_{Layers} has a positive effect on the performance as expected!

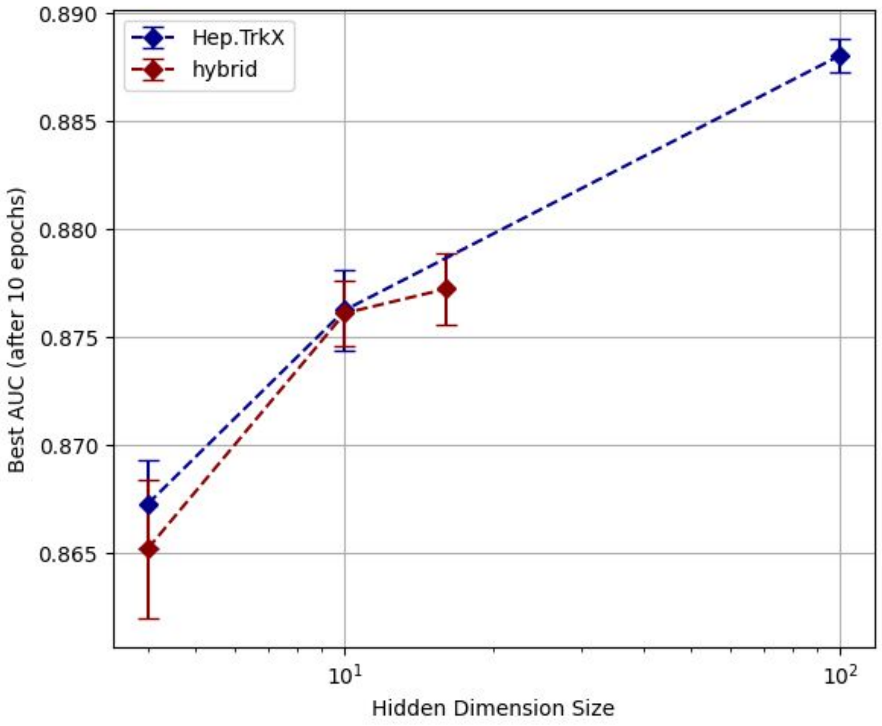
Training set: 50 graphs, Test set: 50 graphs, using ADAM, binary cross entropy, lr = 0.01, analytic results.

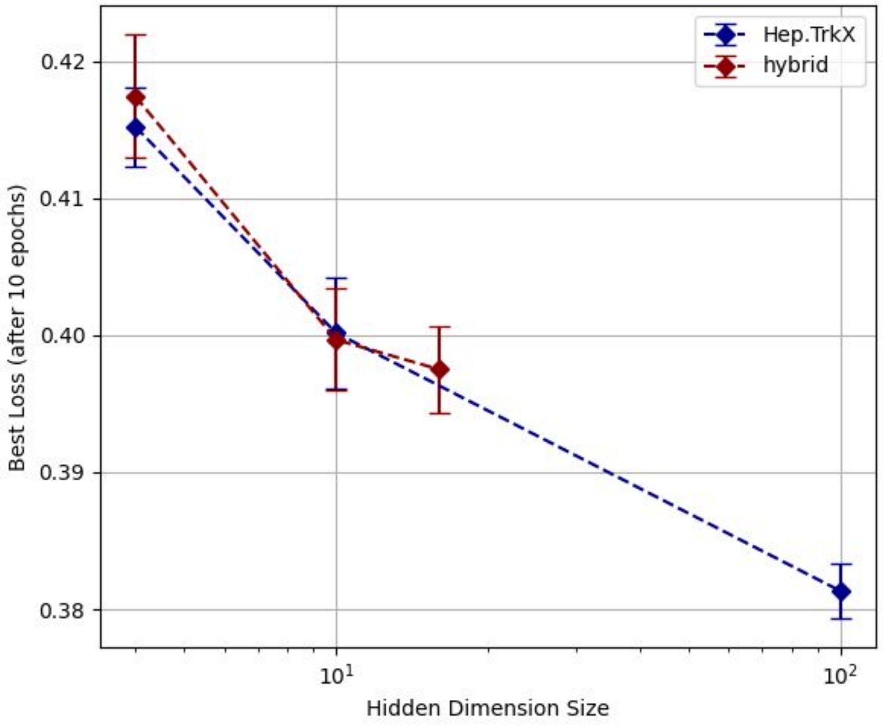
AUC: Area Under ROC, a measure of accuracy for different thresholds.
AUC = 1.0 means perfect score.

Training Results

Comparing Results with Hep.TrkX ($N_{iteration}$ = 3, qc = 10 with 1 layer)

Farrell et al. 2018 (arXiv: 1810.06111)





Our approach shows similar characteristics.
But, it can achieve better AUC and loss with better circuits and more layers!

Training set: 50 graphs, Test set: 50 graphs, using ADAM, binary cross entropy, lr = 0.01, analytic results.

Conclusion

QGNN results are promising. They can achieve similar performance compared to a novel classical model. However, there are still challenges to use this algorithm on a Quantum Computer.

How to improve?

- Use more layers.
- Explore different circuits.
- Explore different architectures.
- Use more events.

Challenges

- Simulation times are long. Quantum models are hard to simulate. (Training takes 1-2 days depending on model complexity)

Things to explore

- Effects of hardware and shot noise.
- Complete overview with more layers and iterations.

Contributors

C. Tüysüz[1,2], C. Rieger[8], K. Novotny[4], B. Demirköz[1], D. Dobos[4,6],
K. Potamianos[4,5], S. Vallecorsa[3], J.R. Vlimant[7]

[1]Middle East Technical University, Ankara, Turkey, [2]STB Research, Ankara, Turkey, [3]CERN, Geneva, Switzerland, [4]gluoNNet, Geneva, Switzerland, [5]Oxford University, Oxford, UK, [6]Lancaster University, Lancaster, UK, [7]California Institute of Technology, Pasadena, California, USA, [8]ETH Zurich, Zurich, Switzerland

Thank you.

Email: ctuysuz@cern.ch, carieger@ethz.ch

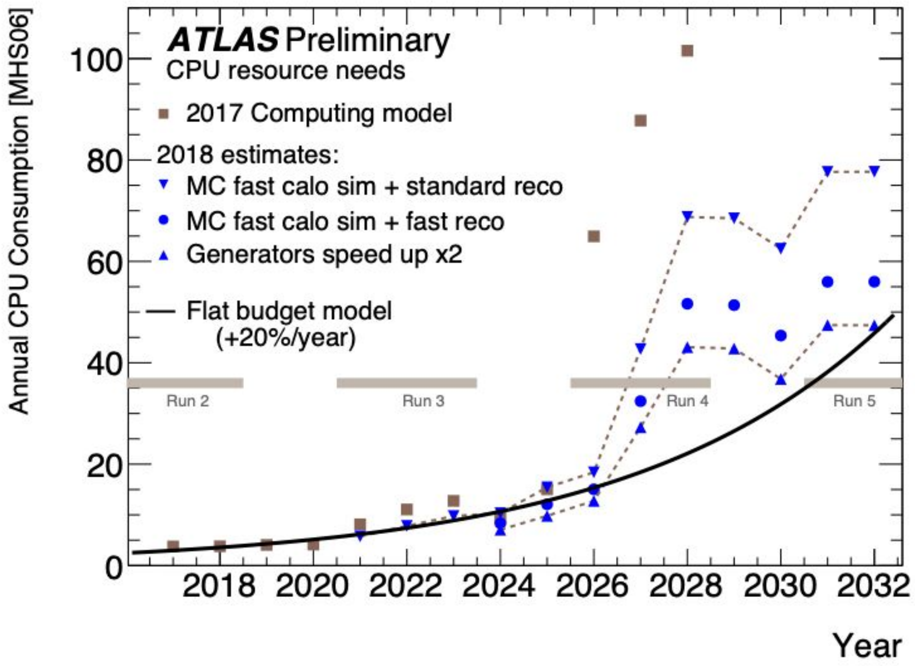
Twitter: @cenk_tuysuz

Results shown here will be published soon, with a complete overview.
You can refer to our recent conference paper for previous results: arXiv:2012.01379
The current code base will be public with the release of the paper.
You can refer to our old codebase: https://github.com/cnktysz/HepTrkX-quantum

Backup Slides

High Luminosity LHC

High Luminosity upgrade of LHC brings many computational challenges.



ATLAS computing model projections for Phase-2

Number of tracks is expected to be increase by 12-15 times

	Run 1	Run 2	Run 3
μ	21	40	150-200?
Tracks	~280	~600	~7-10k

μ: Average number of interactions per bunch crossing

H. Gray, Track reconstruction in the ATLAS experiment, 2016.

Hep.TrkX GNN

Segment Classification

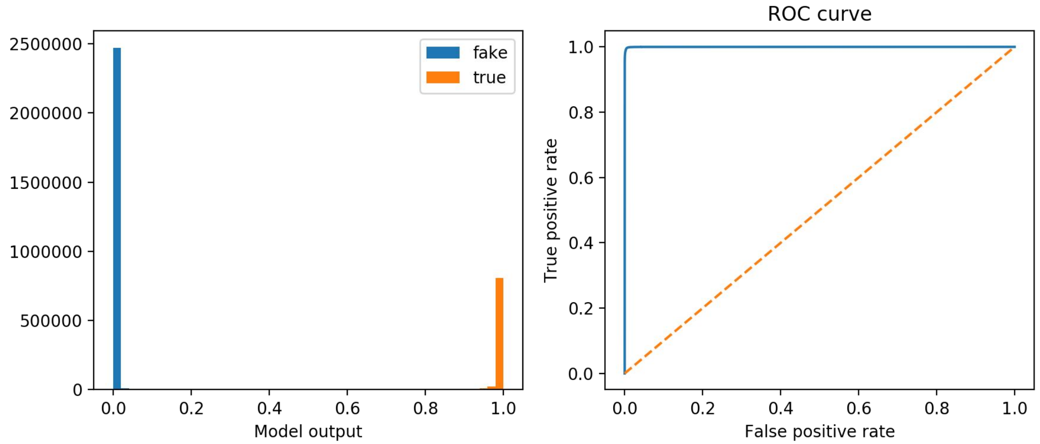
Novel deep learning methods for track reconstruction

Steven Farrell[1,*], *Paolo* Calafiura[1], *Mayur* Mudigonda[1], Prabhat[1], *Dustin* Anderson[2], *Jean-Roch* Vlimant[2], *Stephan* Zheng[2], *Josh* Bendavid[2], *Maria* Spiropulu[2], *Giuseppe* Cerati[3], *Lindsey* Gray[3], *Jim* Kowalkowski[3], *Panagiotis* Spentzouris[3], and *Aristeidis* Tsaris[3]

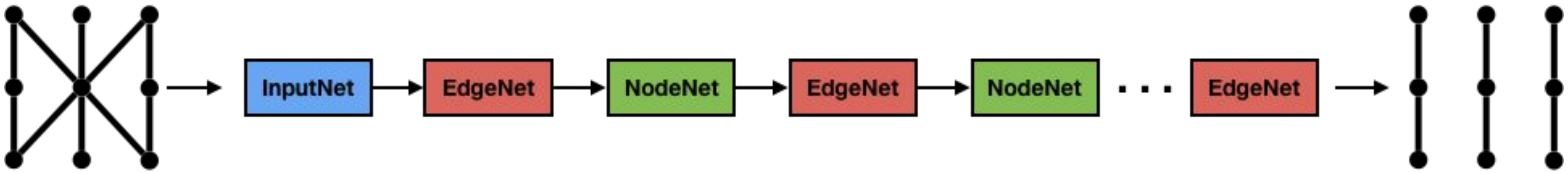
[1]Lawrence Berkeley National Laboratory
[2]California Institute of Technology
[3]Fermi National Accelerator Laboratory

arXiv: 1810.06111

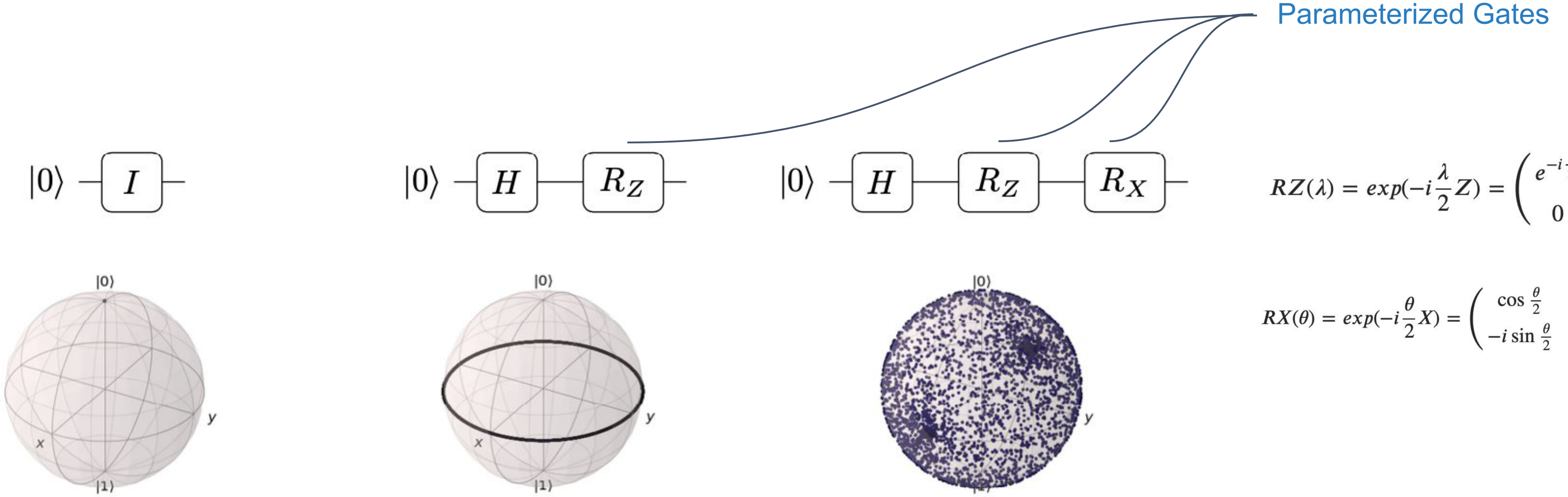
Model Scores (with 0.5 threshold):
Purity: 99.5%, Efficiency: 98.7%
Overall Accuracy: 99.5%



The project is extended with the name Exa.TrkX to continue investigating use of GNNs in track reconstruction.
https://exatrkx.github.io
arXiv:2007.00149



Q.C. for Machine Learning

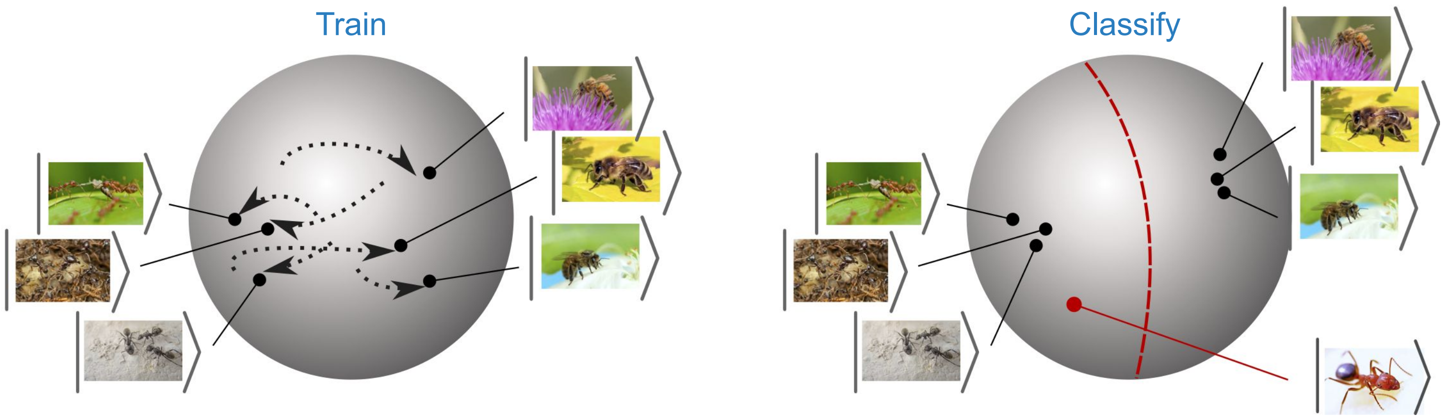


$$RZ(\lambda) = exp(-i\frac{\lambda}{2}Z) = \begin{pmatrix} e^{-i\frac{\lambda}{2}} & 0 \\ 0 & e^{i\frac{\lambda}{2}} \end{pmatrix}$$

$$RX(\theta) = exp(-i\frac{\theta}{2}X) = \begin{pmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{pmatrix}$$

We can use parameterized gates to embed data in the Hilbert Space.

Adapted from: Sim et al. 2019 (arXiv:1905.10876)



Then, we can use other parametrized gates that we can optimize to do tasks such as classification.

Adapted from: Lloyd et al. 2020 (arXiv:2001.03622)

Quantum Classification

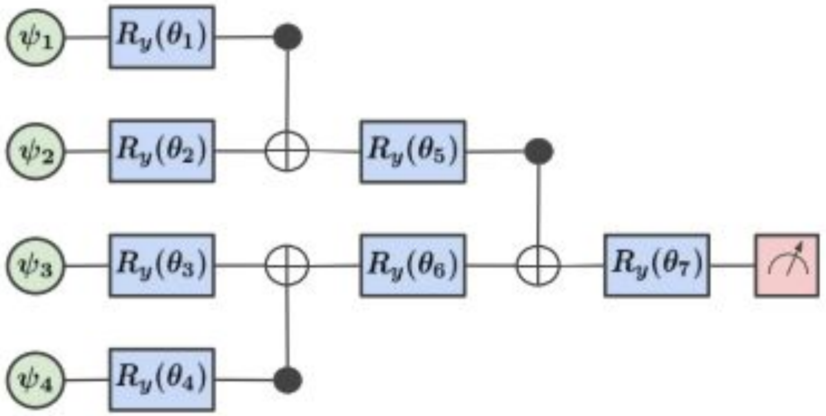
Hierarchical quantum classifiers

Edward Grant[1,2], Marcello Benedetti[1,3], Shuxiang Cao[4,5], Andrew Hallam[6,7], Joshua Lockhart[1], Vid Stojevic[8], Andrew G. Green[6] and Simone Severini[1]

Table 3. Binary classification accuracy on the MNIST dataset

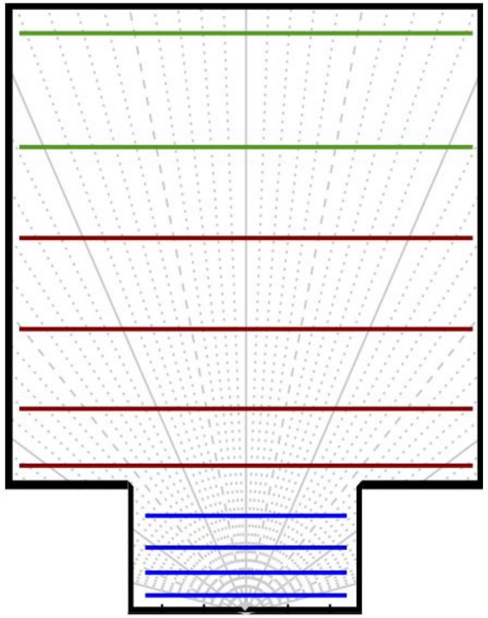
Classifier	Unitaries	Rotations	Is > 4	Is even	0 or 1	2 or 7
TTN	Simple	Real	65.59 ± 0.57	72.17 ± 0.89	92.12 ± 2.17	68.07 ± 2.42
TTN	General	Real	74.89 ± 0.95	83.13 ± 1.08	99.79 ± 0.02	97.64 ± 1.60
MERA	General	Real	75.20 ± 1.51	82.83 ± 1.19	99.84 ± 0.06	98.02 ± 1.40
Hybrid	General	Real	76.30 ± 1.04	83.53 ± 0.21	**99.87** ± 0.02	98.07 ± 1.46
TTN	Simple	Complex	70.90 ± 0.73	80.12 ± 0.64	99.37 ± 0.12	94.09 ± 3.37
TTN	General	Complex	77.56 ± 0.45	83.53 ± 0.69	99.77 ± 0.02	97.63 ± 1.48
MERA	General	Complex	**79.10** ± 0.90	**84.85** ± 0.20	99.74 ± 0.02	**98.86** ± 0.07
Hybrid	General	Complex	78.36 ± 0.45	84.38 ± 0.28	99.78 ± 0.02	98.46 ± 0.19
Logistic	N/A	N/A	70.70 ± 0.01	81.72 ± 0.01	99.53 ± 0.01	96.17 ± 0.01

Mean test accuracy and one standard deviation are reported for TTN, MERA, and hybrid classifiers with five different random initial parameter settings using two different types of unitary parametrization. Hybrid classifiers consist of pre-training a TTN classifier and that transforming it into a MERA classifier by training additional unitaries. Bold values indicate the best result for each classificaiton task



arXiv: 1804.03680

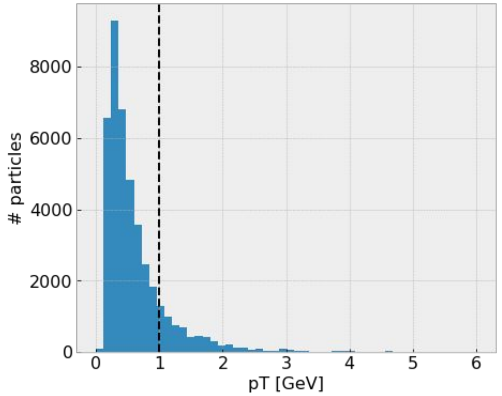
Preprocessing



~15% of hits survive

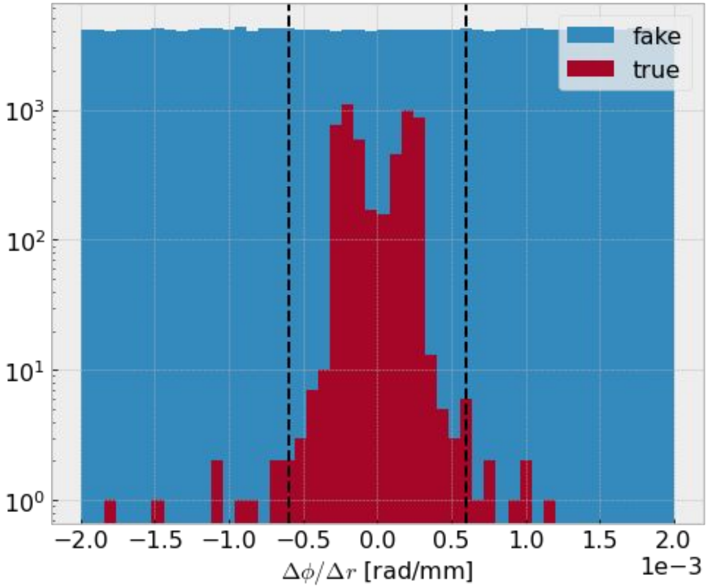
Use hits of particles with pT > 1 GeV

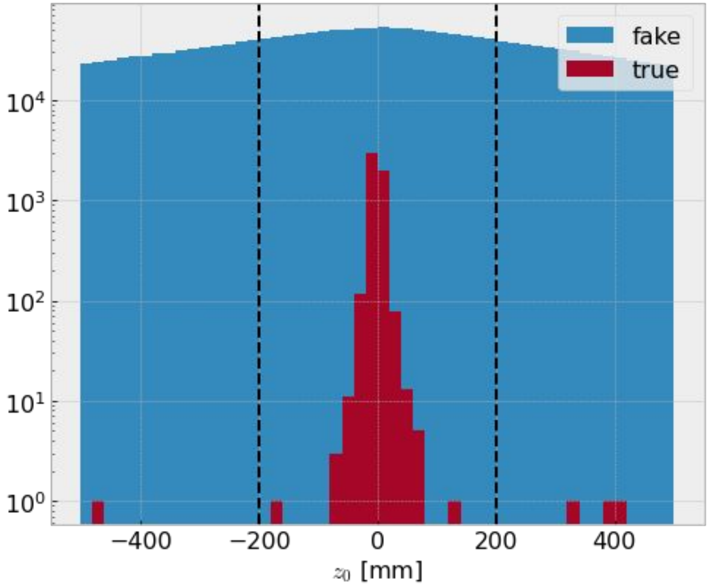
pT distribution of an event



apply cuts to segments

$\|\Delta r/\Delta\phi\|$	< 0.0006
$\|z_0\|$	< 200 mm
$\|\eta\|$	< 5





Use only the barrel region to avoid track ambiguity.

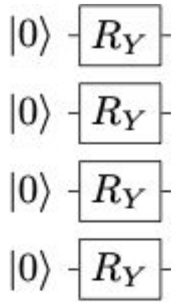
Only 100 events are used!

The preprocessing is used to reduce both the track ambiguity and the size of the dataset. Quantum Machine Learning **simulations** can not handle large datasets at the moment!

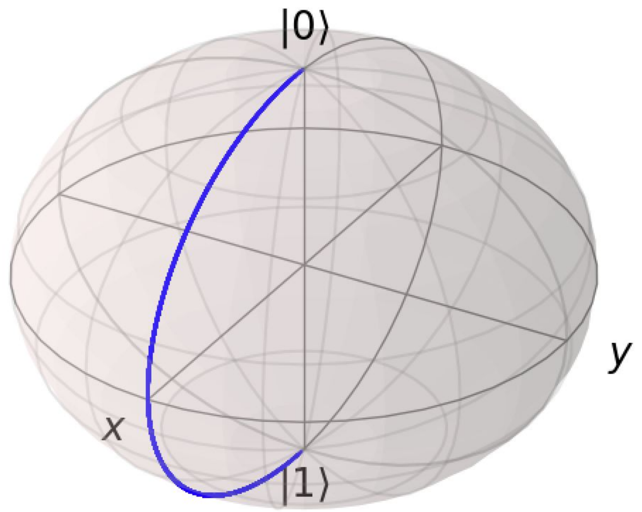
Information Encoding Quantum Circuit

We limit the use of full bloch sphere for two reasons:

- Full circle prevents a 1-1 relation between data and measurements
- Use of full sphere requires complex PQCs (on our radar for future improvements)



$x_i \in [0, \pi]$

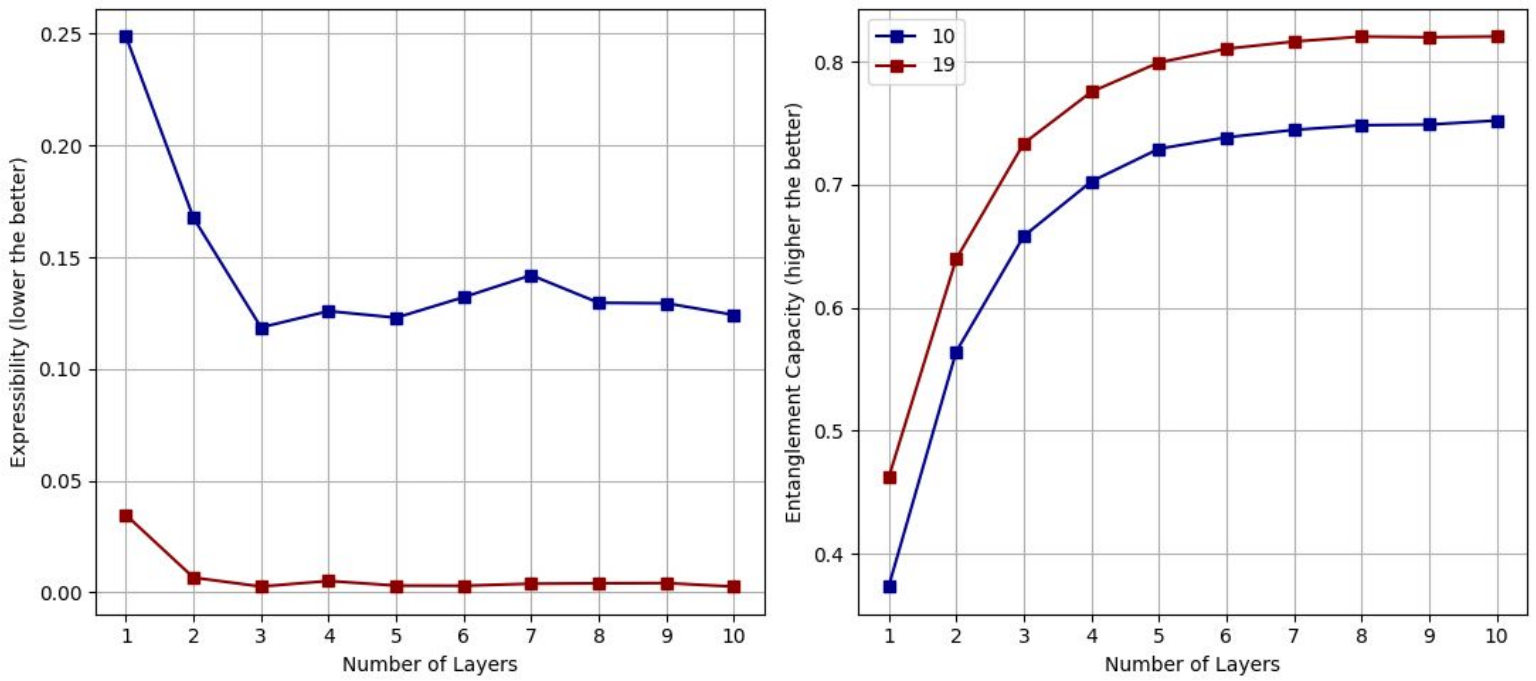


Simple Angle Encoding Circuit:
Requires N_{qubits} = Size of the input

Single Qubit Bloch Sphere Representation

Parametrized Quantum Circuits

How do we choose a Quantum Circuit?

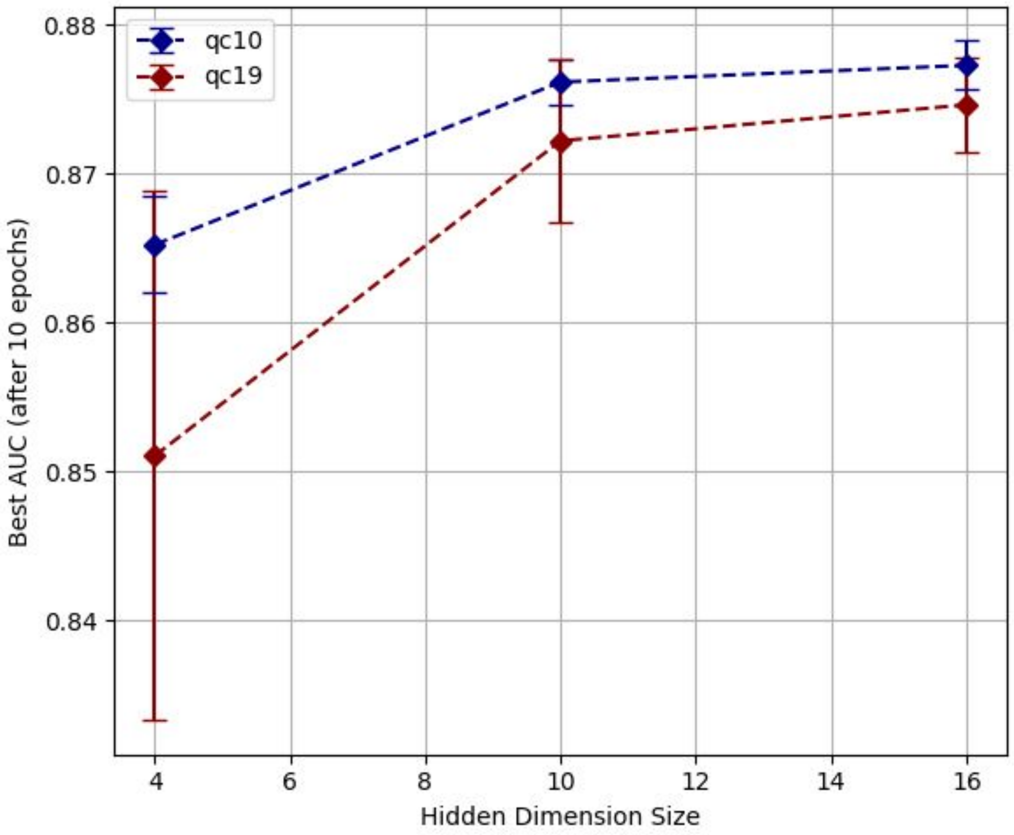


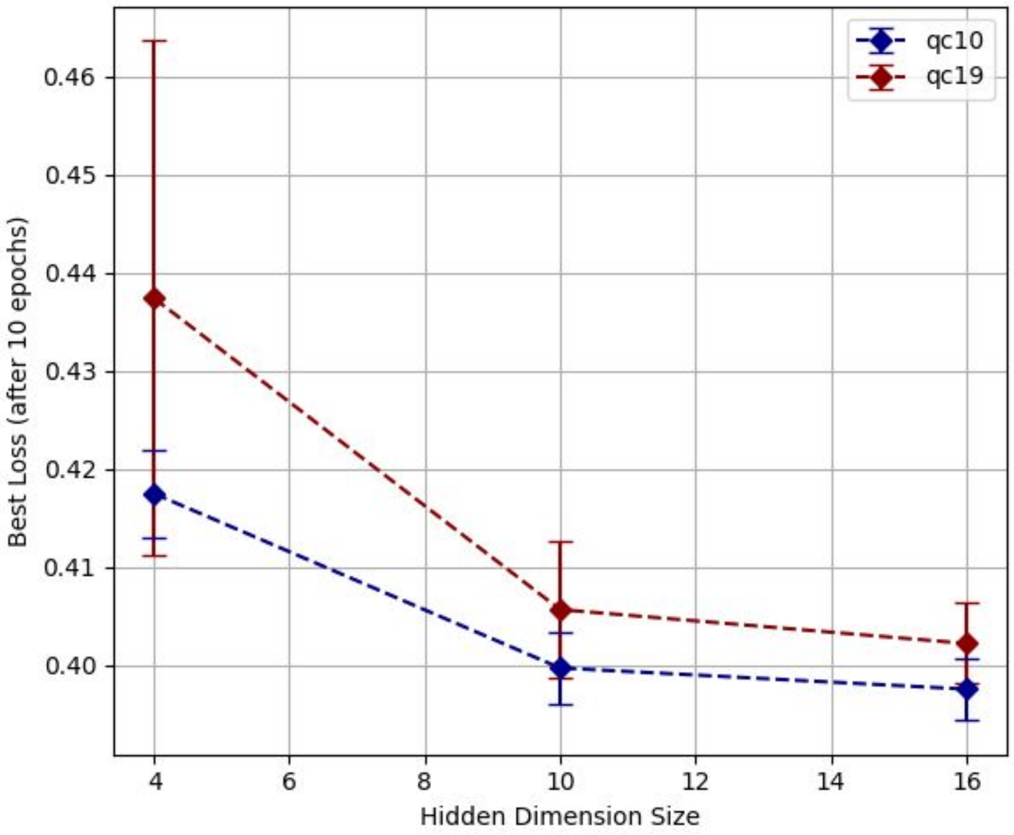
There are metrics in the literature to assess the capacity of Quantum Circuits.
However, they haven't been shown to have correlation with their learning capacity (yet)!
(Sim et al. 2019 (arXiv:1905.10876))

AUC: Area Under ROC, a measure of accuracy for different thresholds.
AUC = 1.0 means perfect score.

Training Results

Hidden Dimension Size ($N_{qubits} = N_{hid}$, $N_{layers} = 1$, $N_{iteration} = 3$)





Hidden Dimension size has a positive effect on the performance as expected.

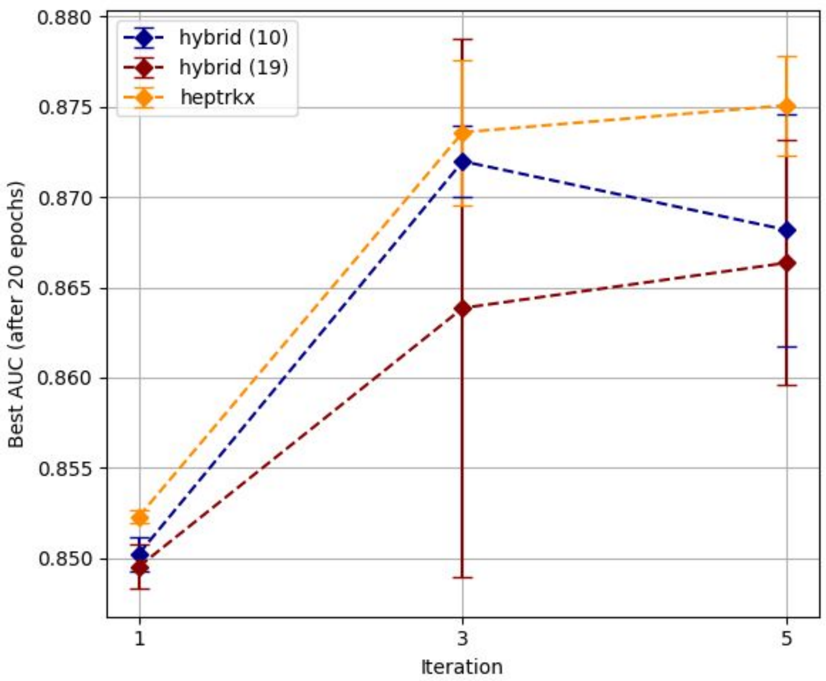
Training set: 50 graphs, Test set: 50 graphs, using ADAM, binary cross entropy, lr = 0.01, analytic results.

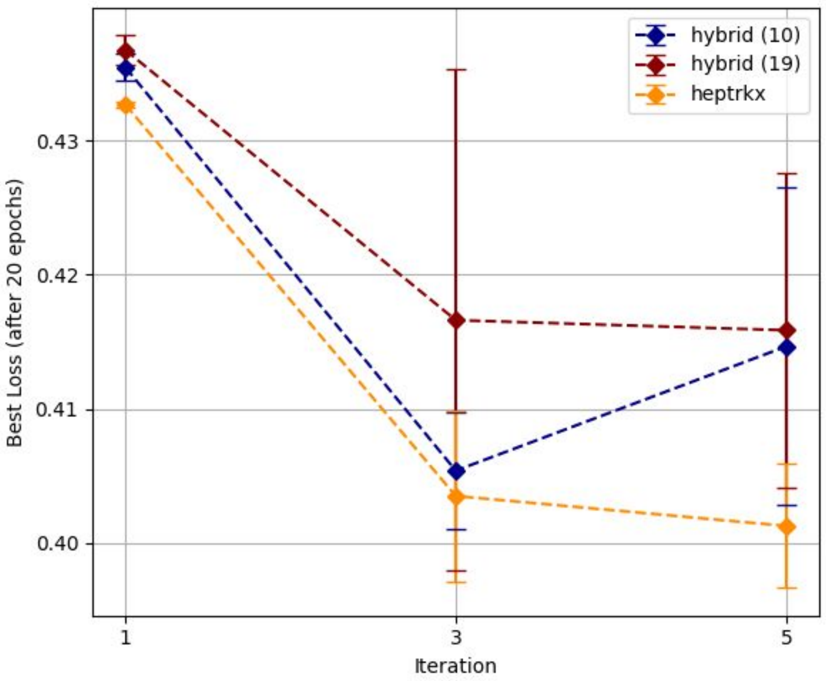
AUC: Area Under ROC, a measure of accuracy for different thresholds.
AUC = 1.0 means perfect score.

Training Results

very small

Number of Iterations (N_{qubits} = 4, N_{hid} = 4, N_{layers} = 1)





$N_{iterations}$ has a positive effect on the performance as expected!
(Exa.TrkX team reported 8 as the optimum amount.)
https://exatrkx.github.io

Training set: 50 graphs, Test set: 50 graphs, using ADAM, binary cross entropy, lr = 0.01, analytic results.