The prospect and frontiers of the application of quantum machine learning in HEP

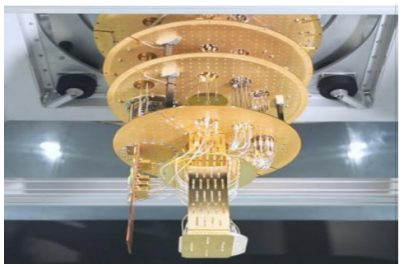
Abdualazem Fadol, Yaquan Fang, Qiyu Sha, Chen Zhou

Sebtember 13, 2022

Contents

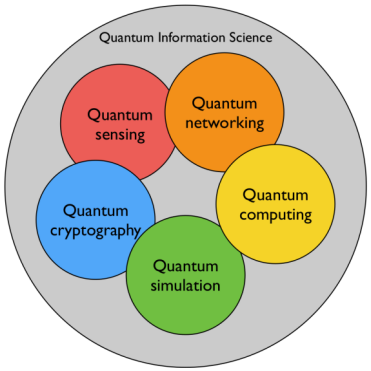
Overview

- Quantum computing systems:
  - Circuit model
  - Turing machine
  - Adiabatic quantum computation
  - One-way quantum computer
  - Cellular automata
- The main goal is to break the limitations:
  - handling more complex data
  - speed and storage problem
- However, quantum computers are at their early stage.
- Is it possible to use machine learning in quantum computing?
  - locating more computationally complex feature spaces
  - better data classification
  - smarter algorithms which give accurate prediction.

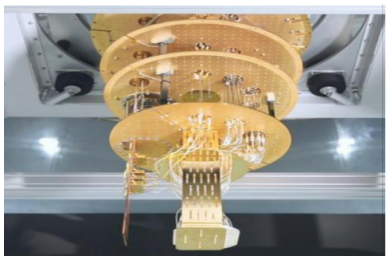


Overview

- We explore machine learning—support-vector machines:
  - In quantum computing systems
  - Conventional system
- Also, the performance in quantum simulator is studied.



- See Introduction to Quantum Computing: Maria Arsuaga here



- Supervised machine learning algorithms for classifications;

$$(\vec{x}_i, y_i) \dots (\vec{x}_n, y_n)$$

- If the data is not linearly separable $\Rightarrow$ move to Kernel

$$k_{ij}(\vec{x}_i, \vec{x}_j) = \langle f(\vec{x}_i), f(\vec{x}_j) \rangle$$

- The function $f(\vec{x})$ could be:
  - Radial basis function

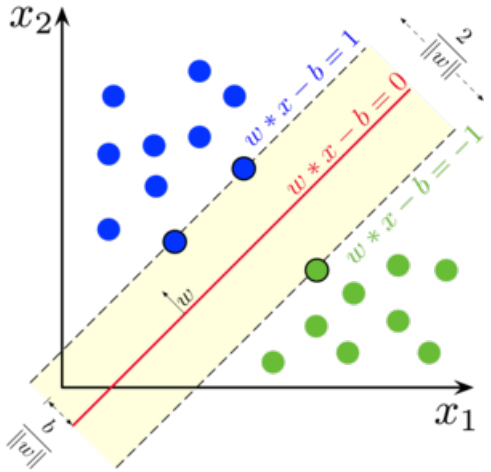
$$f(\vec{x}_i) = e^{-\frac{\vec{x}_i^2}{2\sigma^2}}$$

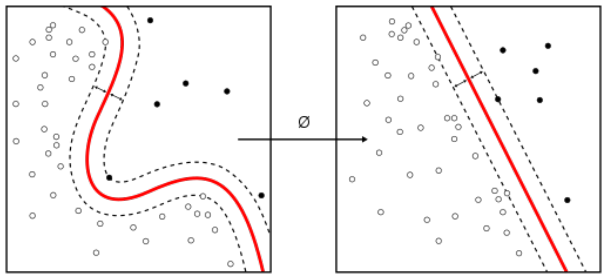
  - polynomial

$$f(\vec{x}_i) = (\gamma \cdot \vec{x}_i^T + r)^d; \quad \gamma > 0$$

  - sigmoid

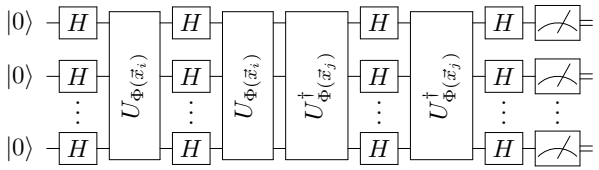
$$f(\vec{x}_i) = \tanh(\gamma \cdot \vec{x}_i^T + r)$$

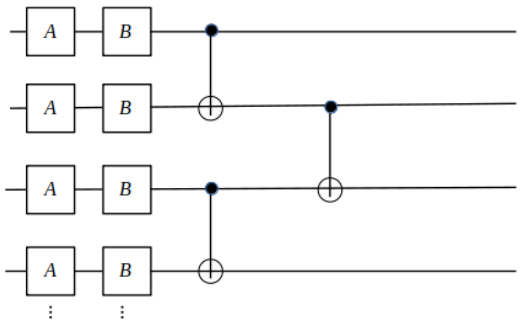




- In a quantum kernel, a classical feature $\vec{x}$ is mapped to higher dimension Hilbert space like $|\phi(\vec{x})\rangle\langle\phi(\vec{x})|$ in such a way that:

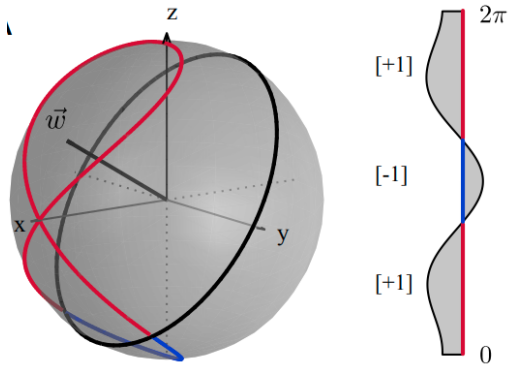
$$k_{ij}(\vec{x}_i, \vec{x}_j) = |\langle\phi(\vec{x}_i)|\phi(\vec{x}_j)\rangle|^2$$





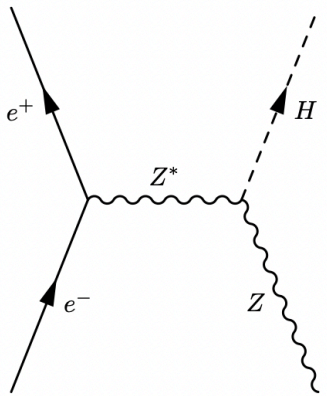
- The circuit is used to evaluate the kernel;
- Hadamard H and controlled not gates;
- The classical data is encoded using the unitary;
- A: rotates the qubit around the z-axis by $2 \cdot x_i$;
- B: rotates the qubit around the y-axis by x_i;
- To match the qubit structure:

$$\vec{x}_i : \vec{x}_i \to \vec{x}_i, \text{ where } \vec{x}_i \in [1, -1]$$



CEPC signatures

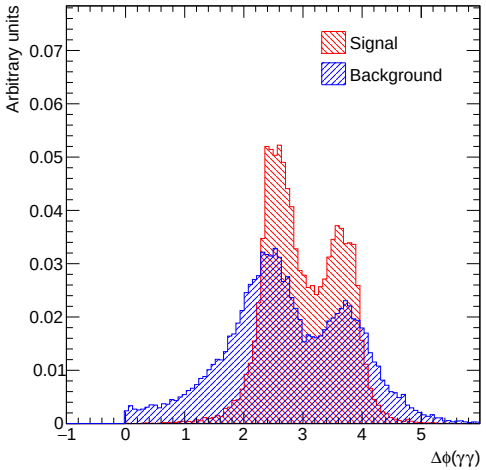
- Use Monte Carlo simulation for the signal and backgrounds:
  - Signal process $e^+e^- \to ZH \to \gamma\gamma q\bar{q}$
  - Backgrounds $e^+e^- \to (Z/\gamma^*)\gamma\gamma$
- The samples are generated with the CEPC configurations;
- at a centre-of-mass energy of 240 GeV with an integrated luminosity of 5.6 ab^{-1}.
- Up to 25k signal and background events are used.
- Six variables are used in both SVM and QSVM (6 qubits).

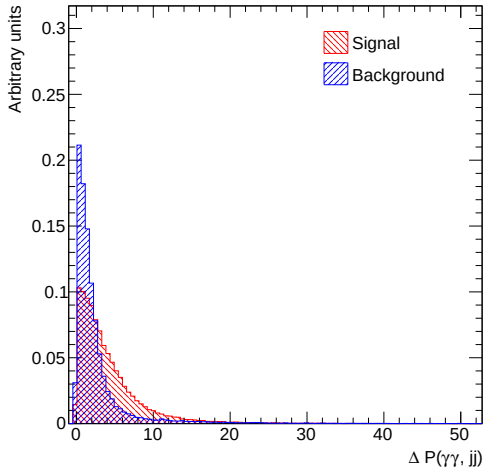


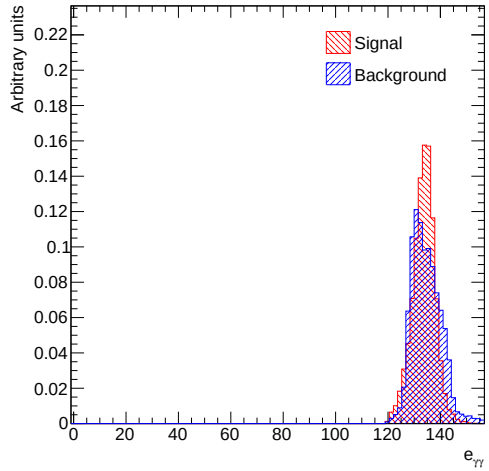
$$\vec{x}_i \to 2 \cdot \frac{\vec{x}_i - \vec{x}_{i,\,\min}}{\vec{x}_{i,\,\max} - \vec{x}_{i,\,\min}}$$

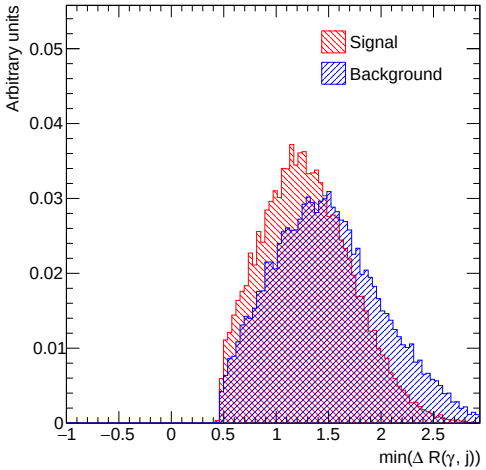
CEPC signatures

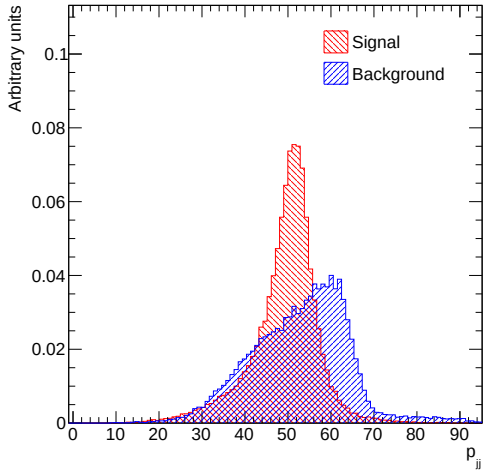
Variables

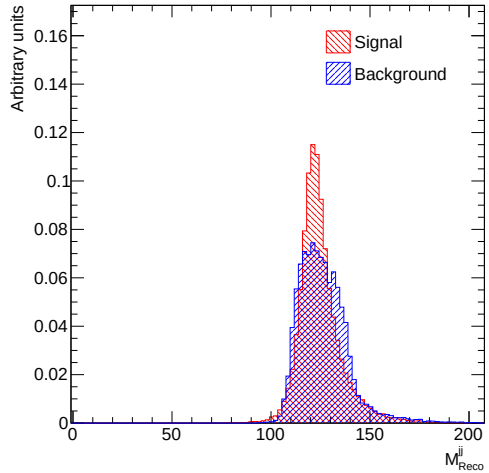




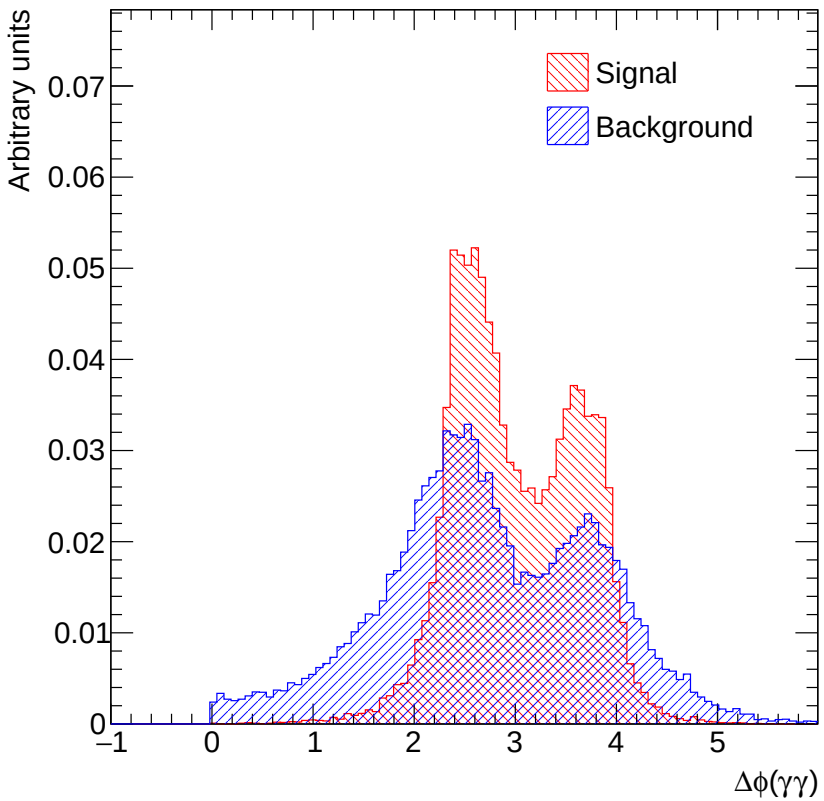


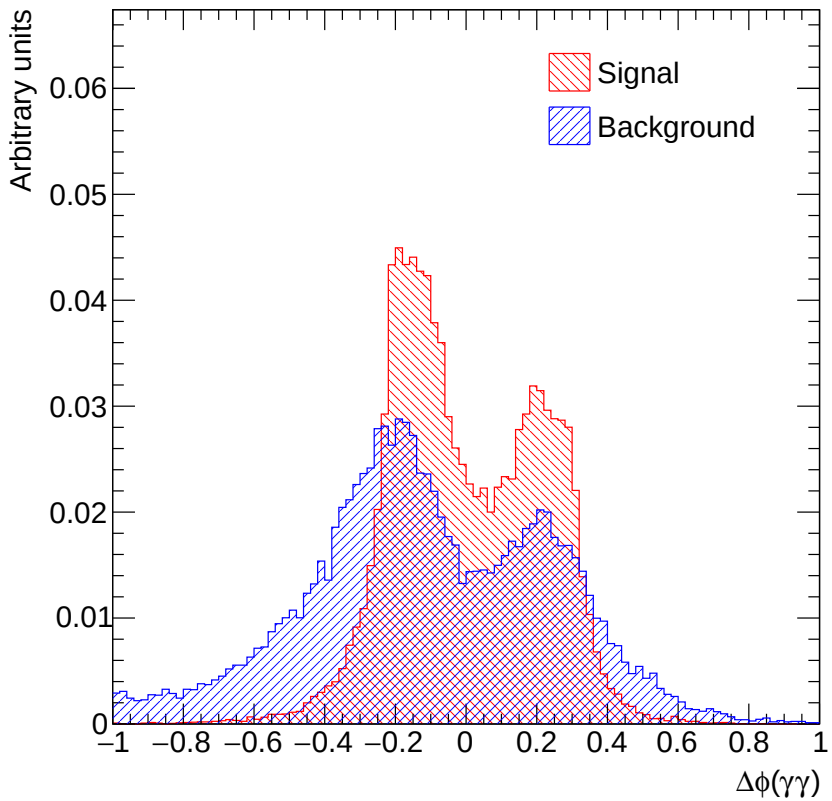






CEPC signatures
Variables



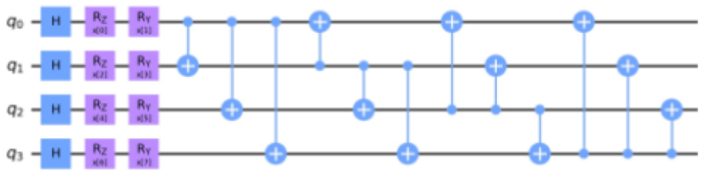
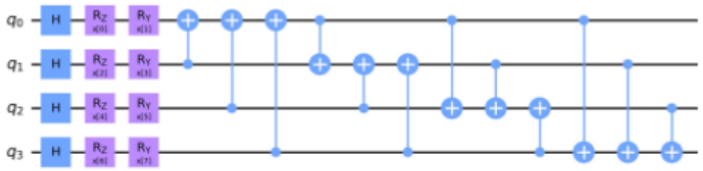


- Mapping $\vec{x}_i$, for each $\vec{x}_i \in \mathbb{R}$, such that:

$$\vec{x}_i : \vec{x}_i \to \vec{x}_i, \text{ where } \vec{x}_i \in [1, -1]$$

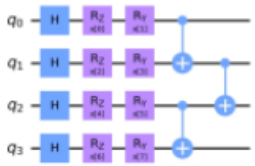
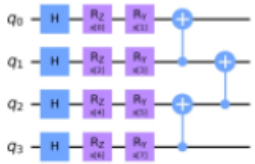
Optimising the feature map

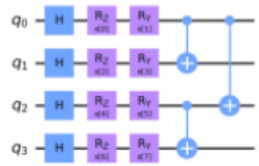
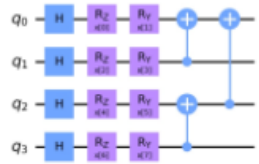
Full forward and backward entanglements

Entanglement	Rotation	Repetition	AUC
	$R_x - R_y$	1	0.763
		2	0.825
		3	0.828
		10	0.719
	$R_y - R_x$	1	0.783
		2	0.831
		3	0.829
		10	0.748
	$R_y - R_z$	1	0.828
		2	0.803
		3	0.820
		10	0.720
	$R_z - R_y$	1	0.837
		2	0.798
		3	0.836
		10	0.750
	$R_x - R_y$	1	0.754
		2	0.842
		3	0.813
		10	0.708
	$R_y - R_x$	1	0.768
		2	0.818
		3	0.837
		10	0.755
	$R_y - R_z$	1	0.848
		2	0.829
		3	0.830
		10	0.745
	$R_z - R_y$	1	0.840
		2	0.845
		3	0.835
		10	0.746

Optimising the feature map

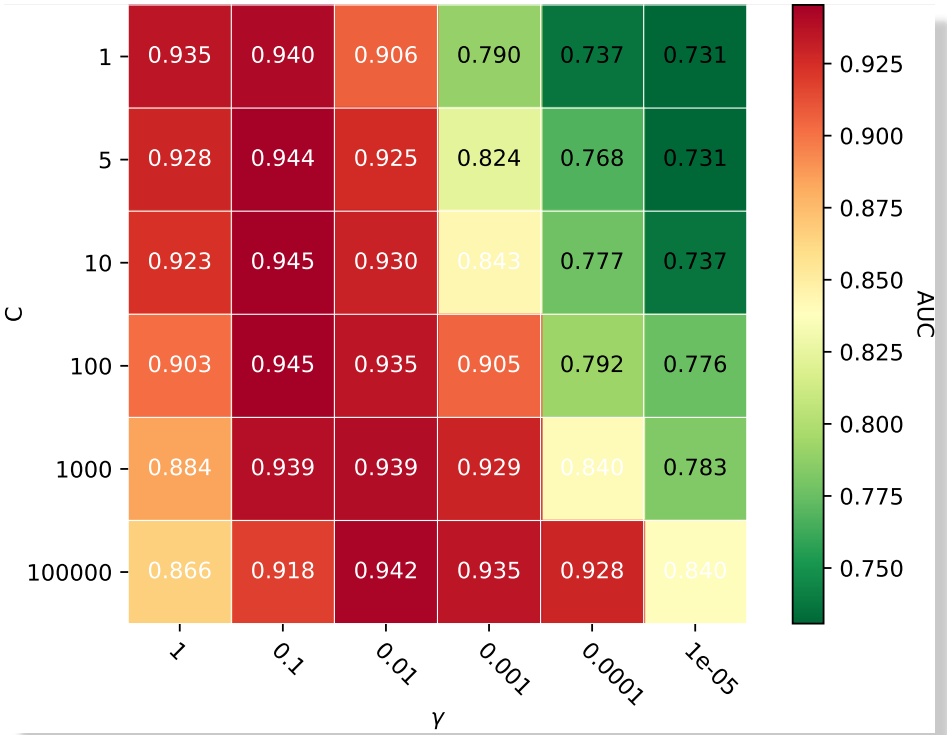
Partial forward and backward entanglements

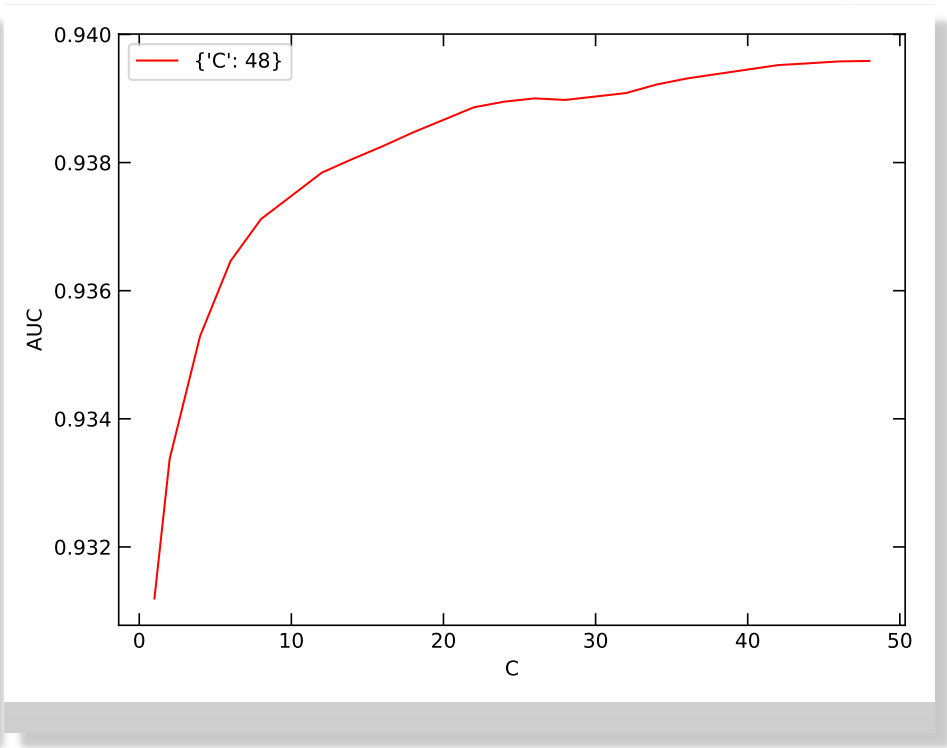
Entanglement	Rotation	Repetition	AUC
	$R_x - R_y$	1	0.780
		2	0.813
		3	0.787
		10	0.767
	$R_y - R_x$	1	0.787
		2	0.830
		3	0.806
		10	0.783
	$R_y - R_z$	1	0.842
		2	0.813
		3	0.819
		10	0.788
	$R_z - R_y$	1	0.852
		2	0.811
		3	0.809
		10	0.756
	$R_x - R_y$	1	0.795
		2	0.835
		3	0.829
		10	0.776
	$R_y - R_x$	1	0.750
		2	0.810
		3	0.826
		10	0.815
	$R_y - R_z$	1	0.841
		2	0.808
		3	0.823
		10	0.769
	$R_z - R_y$	1	0.849
		2	0.810
		3	0.809
		10	0.769

Entanglement	Rotation	Repetition	AUC
	$R_x - R_y$	1	0.769
		2	0.834
		3	0.785
		10	0.751
	$R_y - R_x$	1	0.752
		2	0.819
		3	0.822
		10	0.771
	$R_y - R_z$	1	0.839
		2	0.833
		3	0.806
		10	0.747
	$R_z - R_y$	1	0.829
		2	0.822
		3	0.825
		10	0.759
	$R_x - R_y$	1	0.760
		2	0.839
		3	0.835
		10	0.753
	$R_y - R_x$	1	0.770
		2	0.803
		3	0.817
		10	0.780
	$R_y - R_z$	1	0.843
		2	0.792
		3	0.837
		10	0.753
	$R_z - R_y$	1	0.822
		2	0.812
		3	0.834
		10	0.777

Hyperparameters tuning

C and γ regularisation parameters

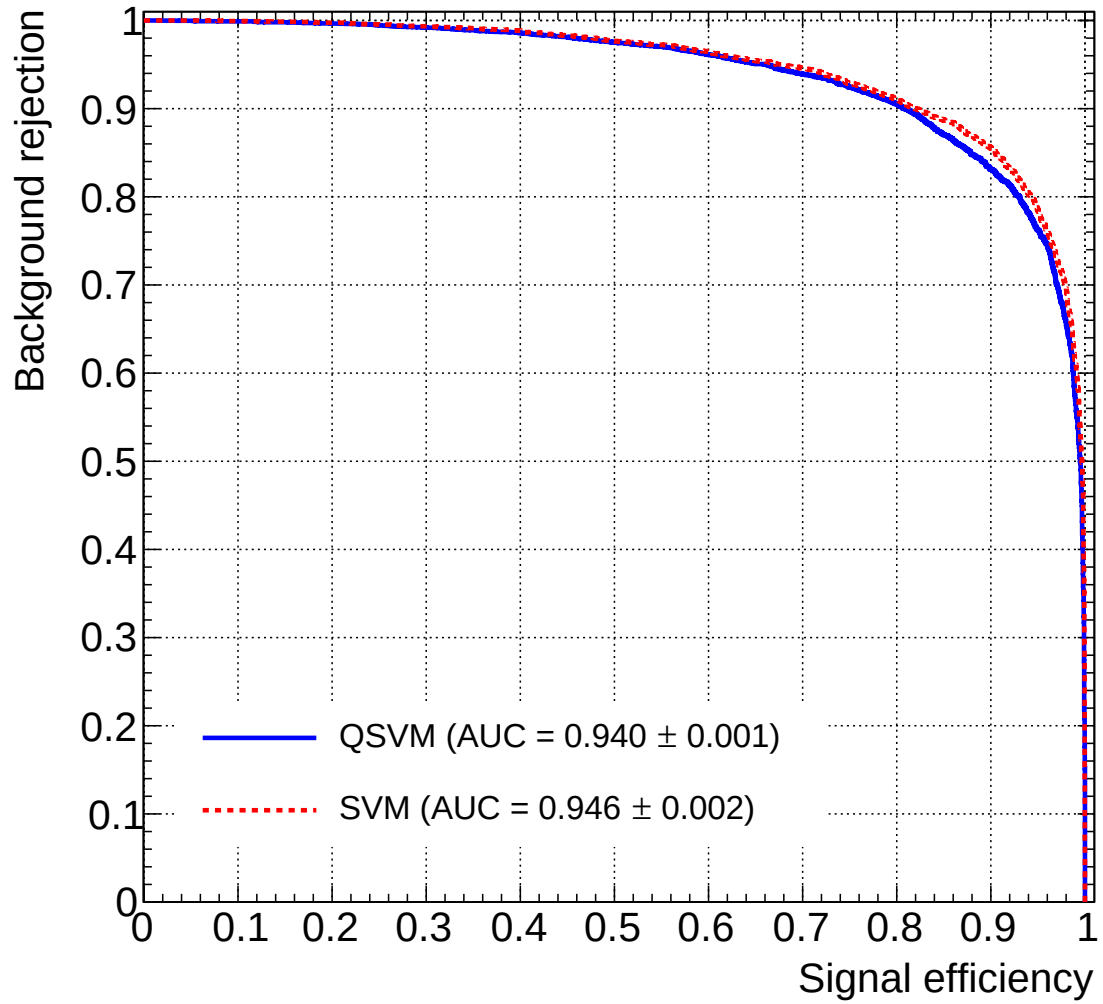




- C is the penalty parameter, which represents misclassification or error term.
- γ defines how far influences the calculation of plausible line of separation.
- SVM (left): $(C, \gamma) = (10/100, 0.1)$; and for QSVM (right): $C = 48$
- IBM quantum simulator, statevector_simulator, is used for the QSVM.
- 12k events is used with cross-validation to find optimal parameters.

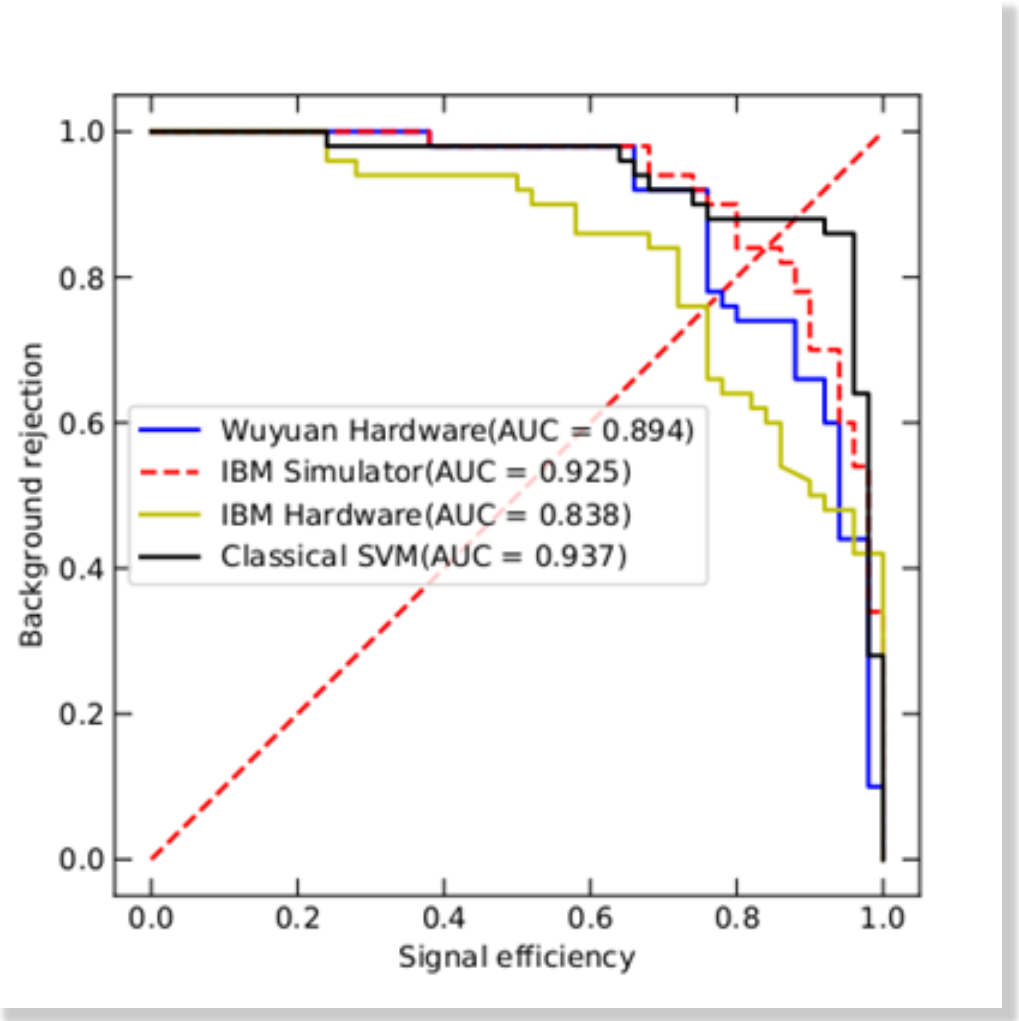
ROC curve result

QSVM, using IBM quantum simulator, vs SVM



Real quantum computing system

IBM quantum computer vs IBM simulator



- Using 100 events for both training and testing the quantum algorithm (6 qubits).

Summary

- We studied the $e^+e^- \to ZH \to \gamma\gamma q\bar{q}$ signal optimisation using machine learning.
- Support-vector machines were compared:
  - Quantum support-vector machines (QSVM) with IBM quantum simulator
  - Classical support-vector machines (SVM)
- Each QSVM and SVM algorithm is optimised to its best before comparing them.
- Real quantum computing system with 100 events:
  - Wuyuan vs IBM
  - IBM vs IBM simulator
- Memory consumption problem with the QSVM when using cross-validation;

```
for i, (train, test) in enumerate(KFold.split(X, y)):
    clfq = qsvc.fit(X[train], y[train])
    scores = clfq.decision_function(X[test])
    fpr, tpr, threshold = sklearn.metrics.roc_curve(y[test], scores)
    roc_auc = sklearn.metrics.auc(fpr, tpr)


    interp_tpr = np.interp(mean_fpr, fpr, tpr)

    interp_tpr[0] = 0.0
    tprsq.append(interp_tpr)
    aruc_QSVMs.append(roc_auc)
```

- when using over 25k events the process is killed "GridSearchCV".

Backup slides

Additional slides
Quantum logic gates

One-qubit gates	Multi-qubit gates
Hadamard $= H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$	CNOT $= CX = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$
$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, $S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$	$CZ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$
$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$	Controlled-U $= CU = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & U_{00} & U_{01} \\ 0 & 0 & U_{10} & U_{11} \end{pmatrix}$
NOT $= X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$	SWAP $= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$, $Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$	Toffoli (CCNOT) $= \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$
$R(\theta) = P(\theta) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}$	

Additional slides