

Application of Quantum Machine Learning to High Energy Physics Analysis at LHC using IBM Quantum Computer Simulators and IBM Quantum Computer Hardware

Speaker: Chen Zhou

**Jay Chan, Wen Guan, Shaojun Sun, Alex Wang, Sau Lan Wu, Chen Zhou
Physics Department, University of Wisconsin-Madison**

and

Miron Livny

**Computing Sciences Department and Wisconsin Institute for Discovery,
University of Wisconsin-Madison**

and

**Federico Carminati, Alberto Di Meglio
CERN Openlab, IT Department, CERN**

and

**Panagiotis Barkoutsos, Ivano Tavernelli, Stefan Woerner, Christa Zoufal
IBM Research Zurich**

July 10-17, 2019 - Ghent, Belgium

**European Physics Society Conference
on High Energy Physics**



Machine learning and quantum computing

- Machine Learning has become one of the most popular and powerful techniques and tools for HEP data analysis
- **Machine Learning: This is the field that gives computers “the ability to learn without explicitly programming them”.**
- Issues raised by machine learning
 - Heavy CPU time is needed to train complex models
 - With the size of more data, the training time increases very quickly
 - May lead to local optimization, instead of global optimization
- Quantum computing
 - **A way of parallel execution of multiple processes using Qubits**
 - **Can speed up certain types of problems effectively**
 - **It is possible that quantum computing can find a different, and perhaps better, way to perform machine learning.**

Ref: “Global Optimization Inspired by Quantum Physics”, 10.1007/978-3-642-38703-6_41

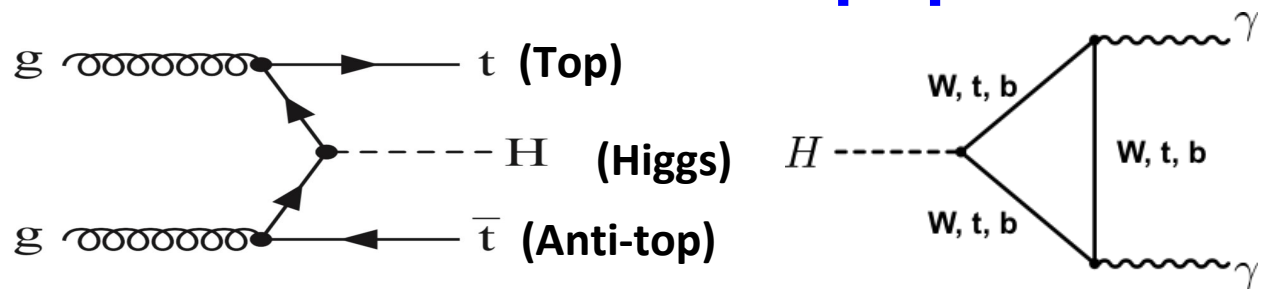
Our program with IBM Qiskit

Our Goal:

Perform LHC High Energy Physics analysis with Quantum computing

Our preliminary program is to:

Employing SVM Quantum Variational (QSVM) method for LHC High Energy Physics (HEP) analysis with the environment of IBM Qiskit, for example ttH ($H \rightarrow \gamma\gamma$), Higgs production in association with two top quarks analysis.



- * SVM = Support Vector Machine
- * IBM Qiskit = IBM Quantum Information Science Kit

An example of classical machine learning: ttH ($H \rightarrow \gamma\gamma$) analysis by the ATLAS Collaboration (ttH: Higgs production in association with two top quarks)

[Phys. Lett. B 784 \(2018\) 173](#)

Select events with **two photons**

[ATLAS-CONF-2019-004](#)

→ Separate to **hadronic channel** ($n_{\text{lep}} = 0$) and **leptonic channel** ($n_{\text{lep}} \geq 1$)

Background: continuum bkg. ($\gamma\gamma$, etc.) and resonant bkg. from other Higgs production modes (ggH, etc.)

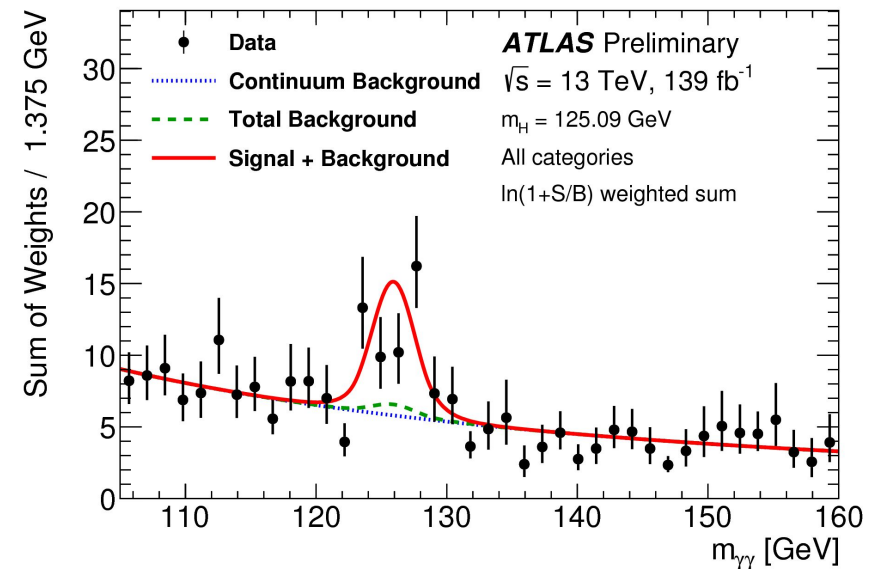
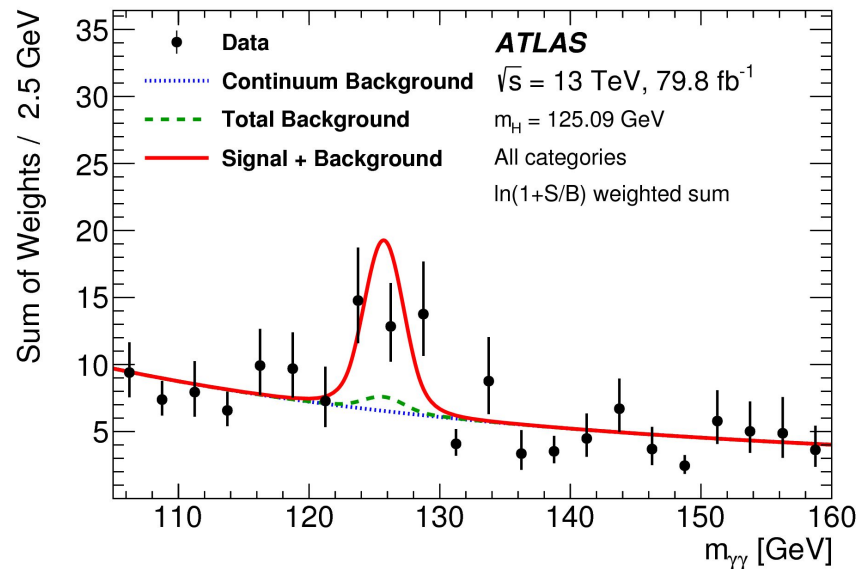
→ In each channel, train a **Boost Decision Tree** (BDT, a classical machine learning technique) with XGBoost package

create **categories based on BDT output**

→ Fit **diphoton mass** over 7 categories

Measure ttH production signal strength, etc.

An example of classical machine learning: ttH ($H \rightarrow \gamma\gamma$) analysis by the ATLAS Collaboration



[Phys. Lett. B 784 \(2018\) 173](#), **80 fb⁻¹**

[ATLAS-CONF-2019-004](#), **139 fb⁻¹**

- The observed significance is **4.1σ** (**4.9σ**) in ATLAS ttH ($H \rightarrow \gamma\gamma$) analysis using 80 fb^{-1} (139 fb^{-1}) of 13 TeV data
- This talk is to perform the machine learning step of the ATLAS ttH ($H \rightarrow \gamma\gamma$) analysis with delphes simulation events **using quantum machine learning**

Our program with IBM Qiskit

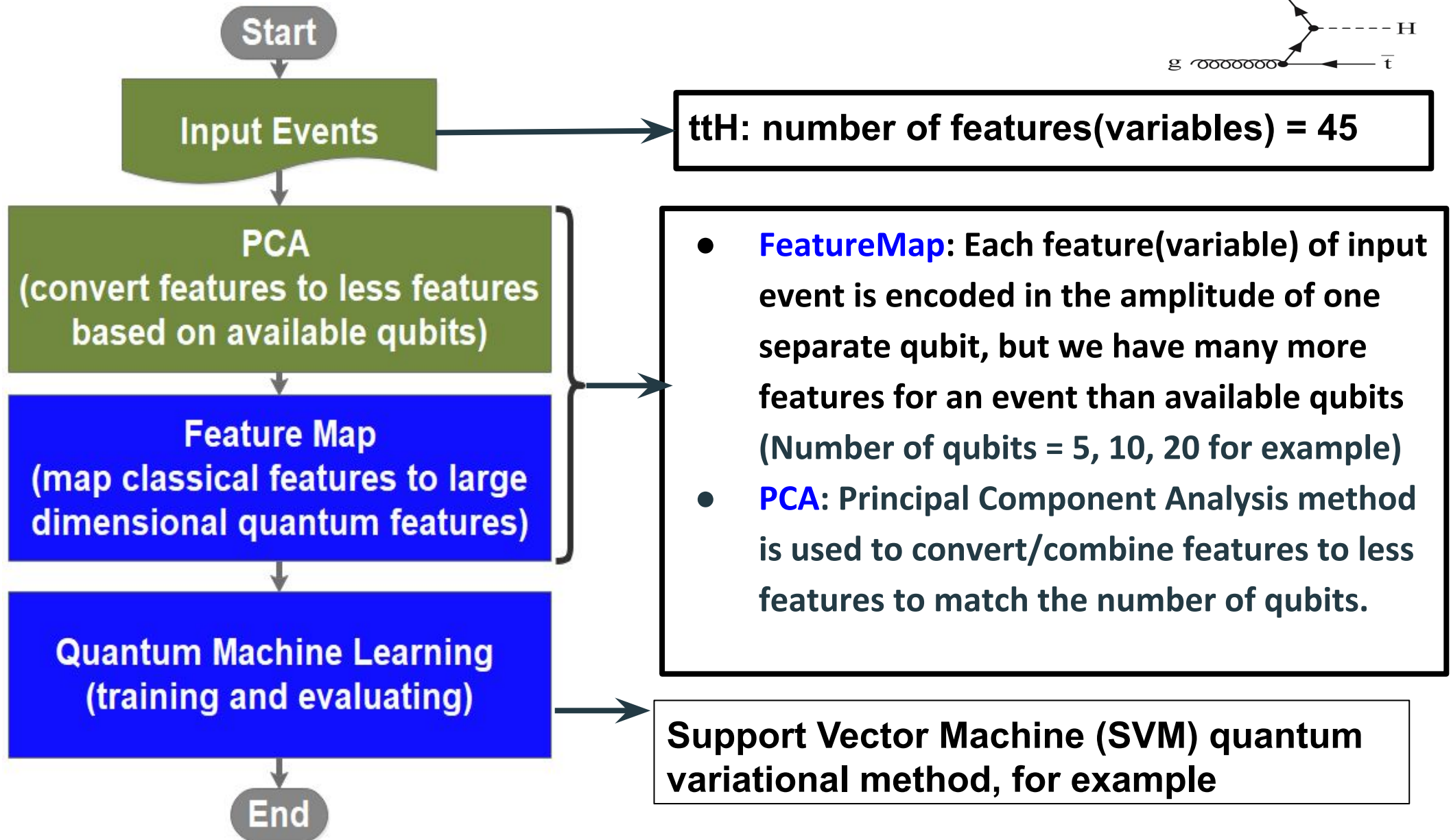
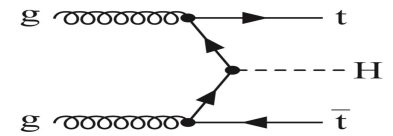
Our preliminary program can be divided into three parts with the Environment of IBM Qiskit:

Part 1. Our workflow for quantum machine learning process.

Part 2. Employing the quantum method for LHC High Energy Physics (HEP) analysis, with quantum simulators, for example IBM Qiskit qasm simulator.

Part 3. Employing the quantum method for LHC High Energy Physics (HEP) analysis, with IBM quantum hardware, for example IBM Q Experience hardware.

Part 1: Our Workflow for Quantum Machine Learning process



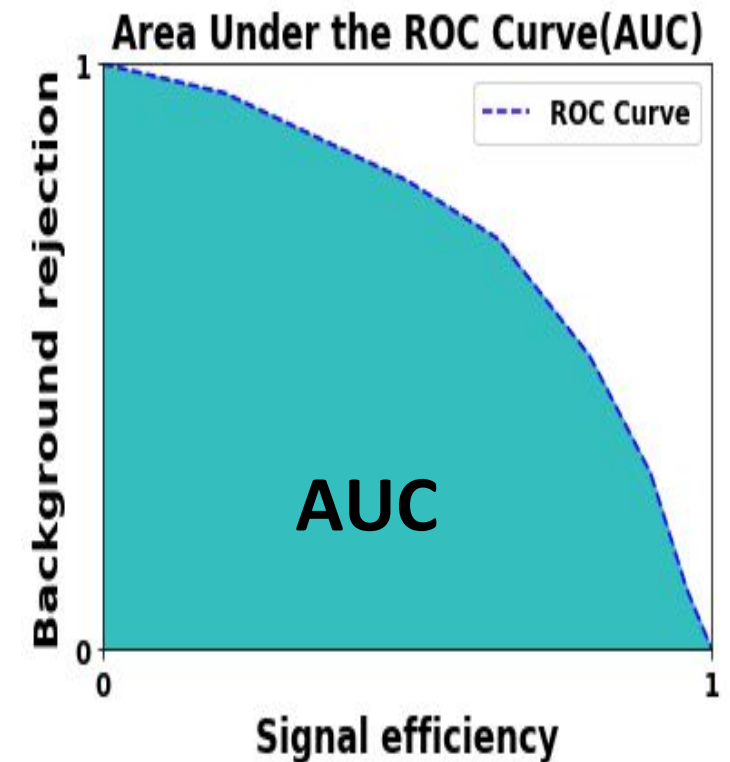
Part 2: Employing QSVM Variational with Q simulators

- **Employing SVM Quantum Variational for LHC HEP analysis**
 - **For example, ttH ($H \rightarrow \gamma\gamma$), Higgs production in association with two top quarks analysis**
 - **Exploring different feature maps and entanglement methods**
 - **Training and evaluating quantum machine learning methods with different numbers of qubits, different number of events, different parameters and optimizers**

Part 2: Employing QSVM Variational with Q simulators

● Definitions

- **BDT**(Boosted Decision Tree) method is a classical machine learning method. Here we are using XGBoost.
- **Q simulator**: Quantum circuits simulator, such as Qasm simulator.
- **Accuracy**: The fraction of correct predictions from total predictions.
- **ROC Curve**: a graph showing background rejection vs signal efficiency.
- **AUC**: Area Under the ROC Curve



Ref: <https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc>

Part 2: Employing QSVM Variational with Q simulators

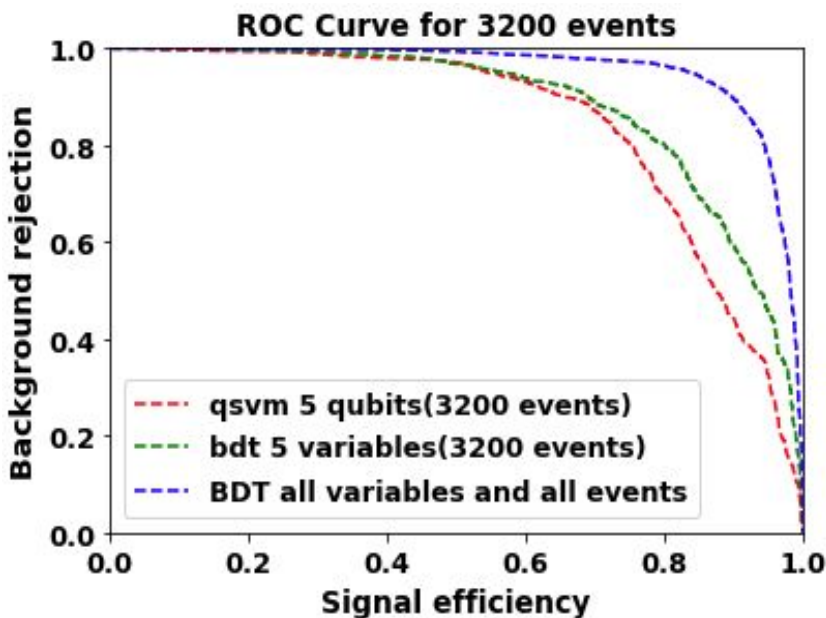
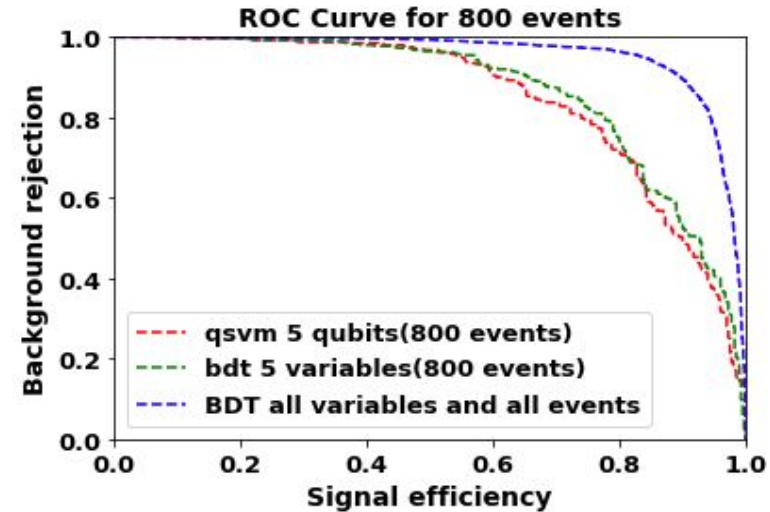
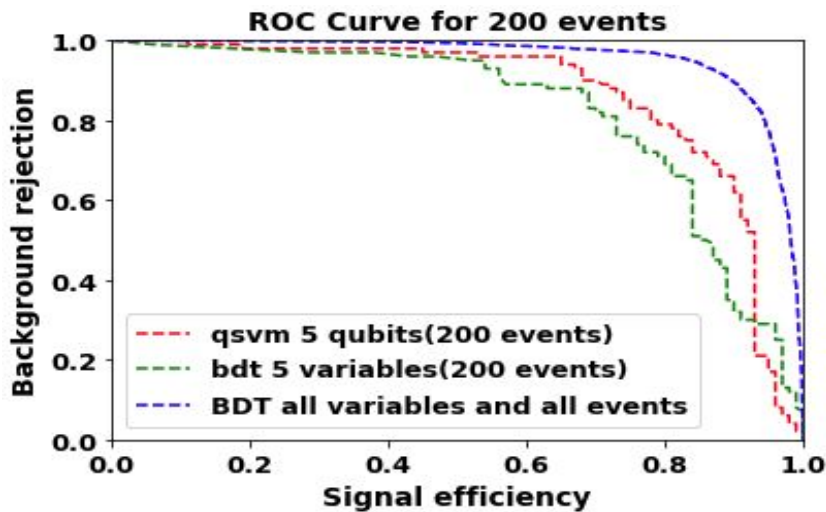
- With 5 qubits, we successfully finished training and testing with 200 events, 800 events and 3200 events with IBM Qiskit qasm simulator (where '200' events means 200 training events and 200 test events; same for others. **Here events are simulated with delphes**).
 - For QSVM, SPSA optimizer is used with 3000 iterations.
 - **BDT and QSVM are using exactly the same inputs for comparison.**
 - Q simulator: Here Qiskit Qasm simulator is used.

ttH(H- \rightarrow $\gamma\gamma$) ACCURACY	200	800	3200
QSVM	0.795	0.802	0.768
BDT	0.75	0.785	0.780

ttH(H- \rightarrow $\gamma\gamma$) AUC	200	800	3200
QSVM	0.865	0.859	0.837
BDT	0.821	0.869	0.863

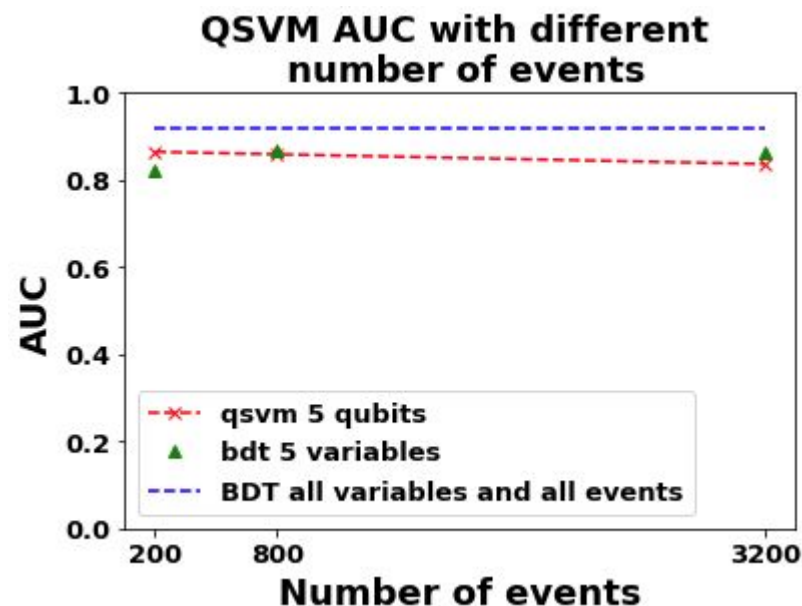
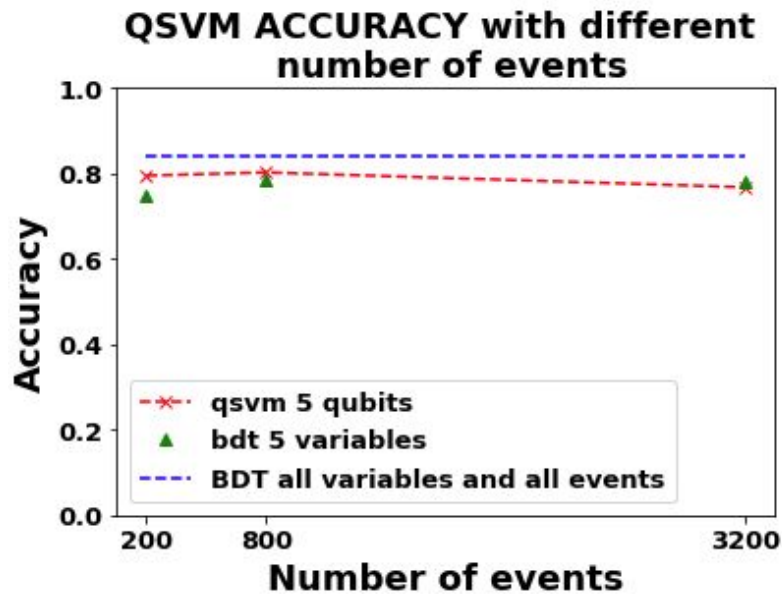
Part 2: Employing QSVM Variational with Q simulators

- Here **BDT(green)** and **QSVM(red)** are using exactly the same inputs for comparison.



- Here are the ROC Curve plots with **QSVM(red)** and **BDT(green)**, for 200 events, 800 events and 3200 events, with 5 qubits.
- ROC curve(Blue):** classical machine learning BDT with all 45 variables and all simulated events.
- ROC curve(Red):** QSVM, 5 qubits
- ROC curve(Green):** classical machine learning BDT with the same inputs of 5 variables per event as QSVM for comparison.
- For 800 events, the **red** curve is close to the **green** curve

Part 2: Employing QSVM Variational with Q simulators

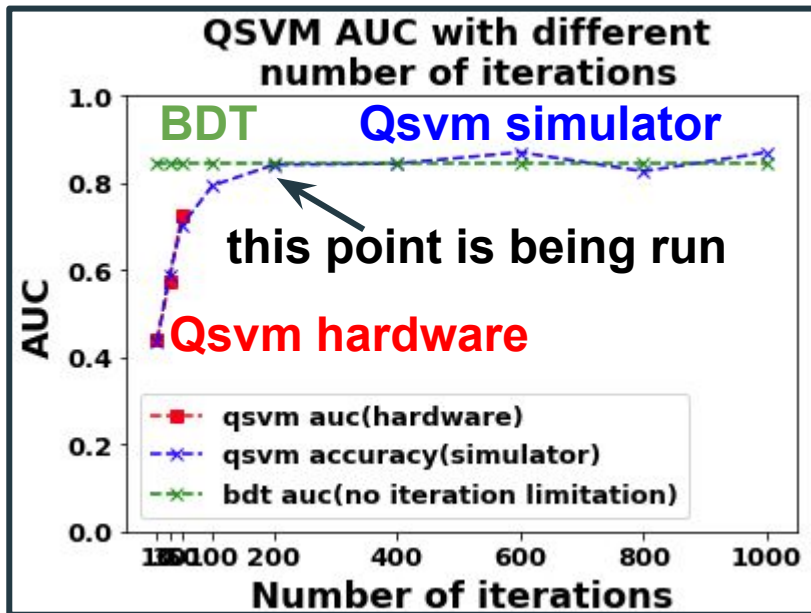
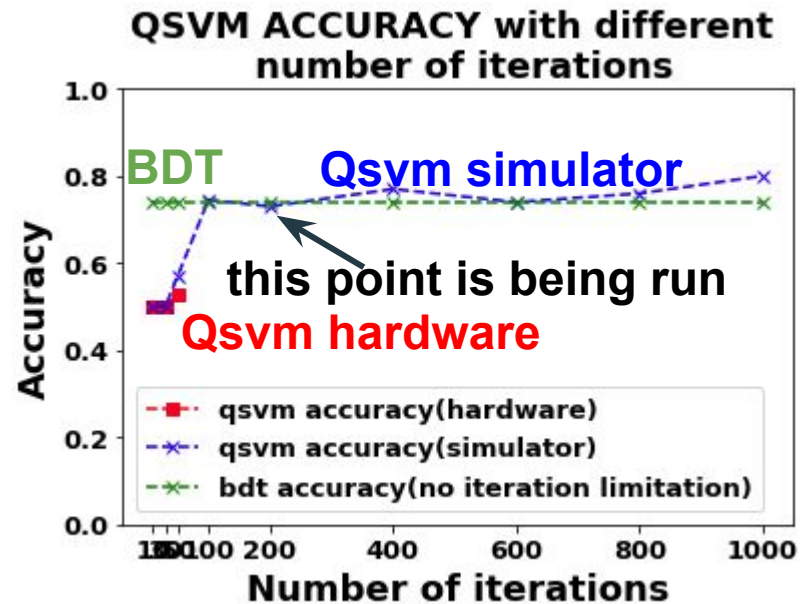


- Here are the ACCURACY and AUC between **QSVM(red)** and **BDT(green)**, with 200 events, 800 events and 3200 events.
 - **QSVM(red)** method got close accuracy and AUC with **BDT(green)** with 5 qubits and limited number of events. But it's still far from BDT with all variables and all events(blue).
 - Working on the loss functions to improve qsvm AUC.

Part 3: Employing QSVM Variational with IBMQ hardware

- With the help of IBM Research Zurich, we finished some training on the IBMQ hardware with 100 training events and 100 test events, 5 qubits.
- Because of hardware access time and timeout limitation, we only finished **very few iterations (for example 10,30,50)** on the hardware, instead of several **thousands of iterations** on the simulators.

Part 3: Employing QSVM Variational with IBMQ hardware



- Here are the ACCURACY and AUC plots with different number of iterations and 100 delphes events.
 - Because of access time limitation, on the hardware we only finished 10,30 and 50 iterations.
- Within limited iterations, the result from hardware(red) is compatible with the result from simulator(blue) in tested iterations.
- The result from simulator(blue) reached similar result with classical BDT(green) method with enough iterations.

Part 3: Employing QSVM Variational with IBMQ hardware

- **Temporary limitation with IBMQ hardware**
 - **Only few iterations are tested currently**
 - **Limited access time**
 - **Long queue time**
 - **Input preparation and output reading is not optimized**
 - **We are working on running hardware with a larger number of iterations**

Summary

Referring to Part 1 of this presentation:

- **We introduced our workflow to employ quantum methods for LHC High Energy Physics analysis.**

Summary

Referring to Part 2 of this presentation:

- Using IBM Qiskit simulator, we have successfully employed Quantum Support Vector Machine method for ttH ($H \rightarrow \gamma\gamma$), Higgs production in association with two top quarks analysis at LHC with delphes simulation events. We have measured the accuracy and AUC with different number of events.
- At current stage, with 5 qubits, we reached very close accuracy of 0.77 and very close AUC of 0.84, comparing with the classical machine learning method(BDT) with accuracy 0.78 and AUC 0.88 (BDT and QSVM are using exactly the same inputs for comparison). At the same time, we are working on the loss functions to improve AUC.

Summary

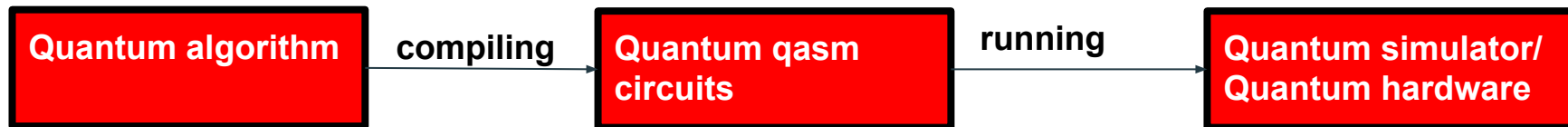
Referring to Part 3 of this presentation:

- **Using IBM Q Experience hardware, we have successfully employed Quantum Support Vector Machine method(5 qubits) for ttH ($H \rightarrow \gamma\gamma$), Higgs production in association with two top quarks analysis at LHC with delphes simulation events.**
- **Again, the accuracy and AUC is limited by the iterations. But the result is compatible with simulator result, which reached similar result with classical machine learning BDT with enough iterations. We are working on running hardware with a larger number of iterations**

BACKUP SLIDES

Quantum algorithm running flow

- Quantum algorithm running flow, for example IBM Qiskit



- **Issues:**
 - The quantum compiling process compiles codes and input data together, while classical compiling separates codes and input data.
 - With more data, the compiling process will use more time and more memory.
 - With different data, a new compiling is required.

* **Qasm = Quantum assembly language**

Quantum measurement

- Quantum state is a superposition which contains the probabilities of possible positions.
- When the final state is measured, they will only be found in one of the possible positions.
 - **The quantum state ‘collapses’ to a classical state** as a result of making the measurement.
- “No-cloning theorem”
 - Impossible to create an identical copy of an arbitrary unknown quantum state.
- To obtain the probability of a possible position, some number of shots are needed.

Hardware Information

- **Hardware status currently**
 - **Classical computer:**
 - **3~4 GHz**
 - **Millions of circuits with many cores, GPU can have thousands of cores**
 - **Quantum computer**
 - **200 ns per operation**
 - **5M Hz**
 - **Not many parallel channels or threads**
 - <https://quantumcomputing.stackexchange.com/questions/2402/how-many-operations-can-a-quantum-computer-perform-per-second>

How to use quantum computer

- How to use quantum computers
 - a. Convert classical features to be able to be processed to quantum computers
 - **Feature map**
 - b. Using quantum algorithms to process the data
 - Algorithms developed based on quantum computers, such as Quantum Support Vector Machine, Quantum annealing, Grover Search and so on

Tensor product feature map

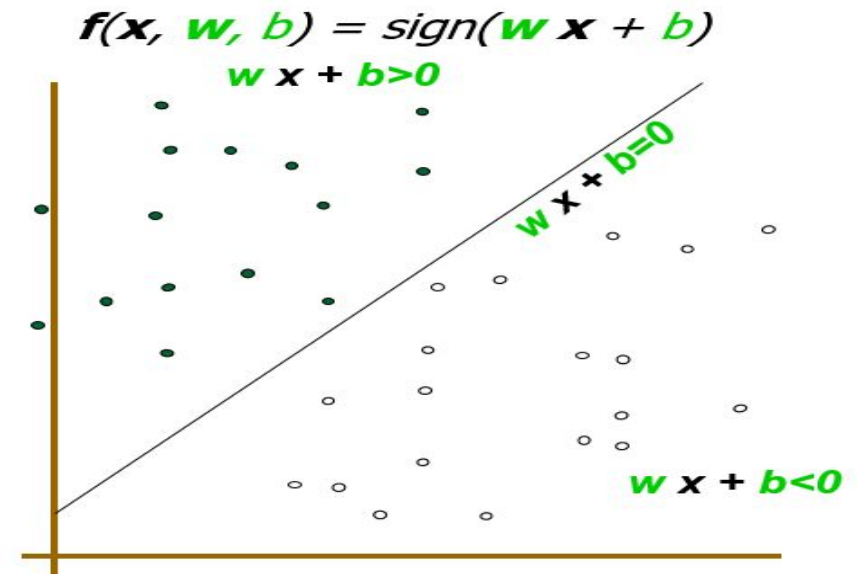
- **Quantum feature map: Map bit info non-linearly to quantum ‘feature Hilbert space’**
 - **Tensor product encoding**
 - Each feature(variable) of input event is encoded in the amplitude of one separate qubit
 - All features of one event is the tensor product of corresponding qubits
 - **Entanglement between features**
 - Without entanglement
 - Between next one feature(linear entanglement)
 - Between all of the next features(full entanglement)

Other feature map methods

- Basic encoding
 - One bit maps to one qubit
 - For example, two bits “01” maps to two qubits “ $|01\rangle$ ”
- Amplitude encoding
 - N classical features maps to $\log_2 N$ qubits
 - $\mathbf{X} = (x_0, \dots, x_{N-1})$, $N=2^n$
 - $|\varphi_x\rangle = \sum x_i^* |i\rangle$ (qubit “ $|i\rangle$ ” is the i 'th computational basis state)
 - Looking whether it's possible and how to do it

Support Vector Machine

- Support Vector Machine (SVM)
 - a supervised ML that draws a decision boundary between two classes to classify data points
 - Originally it's constructed as a linear classifier
 - Maximize the distance from the line or hyperplane to the nearest data point on each side

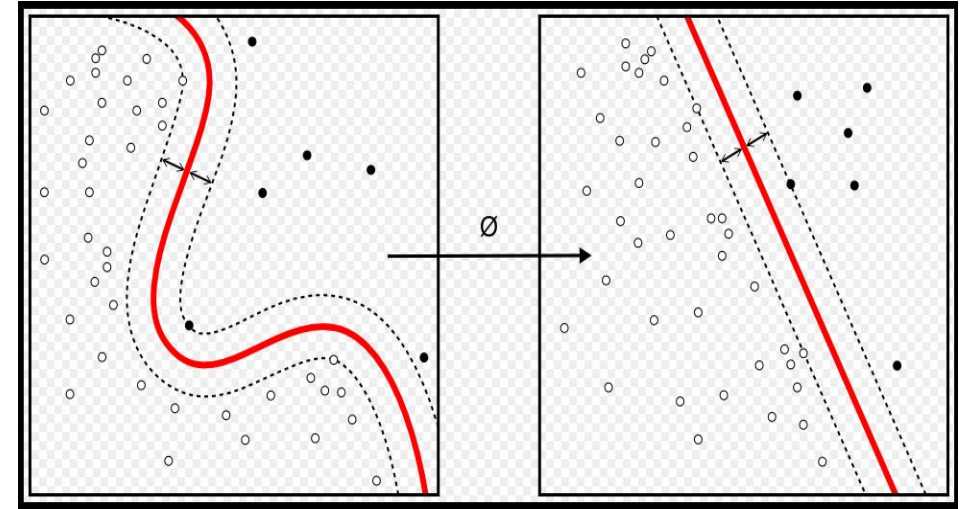


Ref: Support Vector Machine and Its Application(Mingyue Tan, 2004)

Ref: Support vector machine(Wikipedia)

SVM kernel function

- **Kernel function**
 - Often the sets of data points are not linearly separable
 - Map data points to a much higher dimensional space which presumably making the separation easier



Ref: Support vector machine(Wikipedia)

- Performance depends on different kernel functions
- Limitation to successful solutions when feature space becomes large
- Computationally expensive to estimate the kernel

Quantum SVM

- Quantum SVM
 - Take advantage of the large dimensionality of quantum Hilbert space
 - Non-linearly maps input data into a very large dimensional feature Hilbert space
 - Exploiting an exponentially large quantum state space
 - Take advantage of the quantum speedup
 - Estimate the kernel and optimize the classifier