Application of Quantum Machine Learning to HEP Analysis at LHC using Quantum Computer Simulators and Quantum Computer Hardware

Sau Lan Wu

I am new in this field, since two and a half years.
I have assembled an international and interdisciplinary team of High Energy Physicists and Quantum Computing Scientists:

Jay Chan, Alkaid Cheng, Wen Guan, Shaojun Sun, Alex Wang, Sau Lan Wu, Rui Zhang, Chen Zhou
Physics Department, University of Wisconsin-Madison
Miron Livny
Computer Sciences Department, University of Wisconsin-Madison
Federico Carminati, Alberto Di Meglio
CERN Quantum Technology Initiative, IT Department, CERN
Panagiotis Barkoutsos, Ivano Tavernelli, Stefan Woerner, Jennifer Glick
IBM Research Zurich and IBM T.J. Watson Research Center
Andy Li, Joseph Lykken, Panagiotis Spentzouris
Quantum Institute, Fermilab
Samuel Yen-Chi Chen, Shinjae Yoo
Computational Science Initiative, BNL
Tzu-Chieh Wei
C.N. Yang Institute for Theoretical Physics, State University of New York at Stony Brook
Pavel Lougovski, Sanjay Padhi, Simone Severini, Dewayne Walker
Quantum Computing and AI Research, Amazon Web Services

15 July, 2021, CERN
Perspectives on Quantum Sensing and Computing for Particle Physics
CERN TH Institute Workshop

Machine learning for High Energy Physics

- One of the major objectives of the experimental programs at the LHC is the discovery of new physics.
- Machine Learning: “application of artificial intelligence that provides systems the ability to automatically learn and improve from experience without being explicitly programmed”
  - It has become one of the most popular and powerful techniques and tools for High Energy Physics (HEP) data analysis
  - It greatly enhances our ability to identify rare signal against immense backgrounds: important for discovery of new physics
- Issues raised by machine learning
  - Heavy CPU time is needed to train complex models
    - The training time increases with more data
  - May lead to local optimization, instead of global optimization

Machine Learning for High Energy Physics

- Classical Machine learning algorithms commonly used in High Energy Physics data analysis
  - Boosted Decision Tree (BDT): an algorithm that incrementally builds an ensemble of decision trees and combines all the decision trees to form a strong classifier.
  - Support Vector Machine (SVM): it maps the input vectors X into a high-dimensional feature space Z through some nonlinear mapping, chosen a priori. In this space, an optimal separating hyperplane is constructed to separate signal from background.
  - Neural Network (NN): a computing system made up of a number of simple, highly interconnected processing elements, which process information by their response to external inputs.

Quantum Machine Learning

- Quantum computing
  - Perform computation using the quantum state of qubits
  - A way of parallel execution of multiple processes
  - Can speed up certain types of problems effectively
- Quantum machine learning
  - Intersection between machine learning and quantum computing
  - May lead to more powerful solutions and offer a computational "speed up", by exploiting the exponentially large quantum state space through the action of superposition, entanglement, etc
  - Quantum machine learning could possibly become a valuable alternative to classical machine learning for HEP data analysis

Our program with Quantum Machine Learning

Our Goal:

To perform LHC High Energy Physics analysis with Quantum Machine Learning, to explore and to demonstrate that the potential of quantum computers can be a new computational paradigm for big data analysis in HEP, as a proof of principle

Our present program is to employ the following 3 quantum machine learning methods

Method 1. Variational Quantum Classifier Method

Method 2. Quantum Support Vector Machine Kernel Method

Method 3. Quantum Neural Network Method

to LHC High Energy Physics analysis, for example ttH ($H \rightarrow \gamma\gamma$) and $H \rightarrow \mu\mu$ (two LHC flagship analyses).

Our program with Quantum Machine Learning

- We study the quantum machine learning methods on gate-based* quantum computer simulators and hardware:
  - 1. IBM quantum computer simulator and hardware (using IBM Qiskit libraries)
  - 2. Google quantum computer simulator (using Google Cirq and TensorFlow Quantum libraries)
  - 3. Amazon quantum computer simulator (using Amazon Braket Cloud Service)

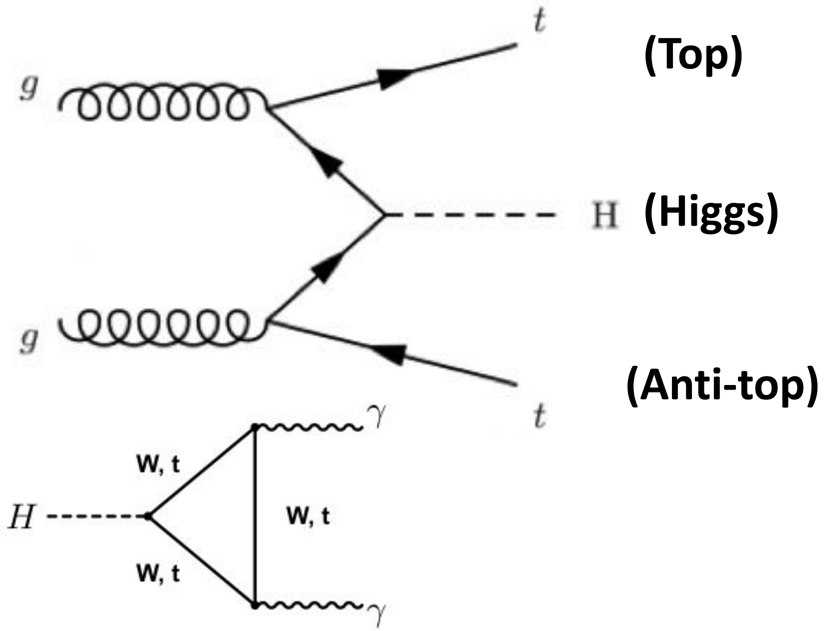
* gate-based: computing is achieved by a sequence of quantum gates, as opposed to D-wave quantum annealers

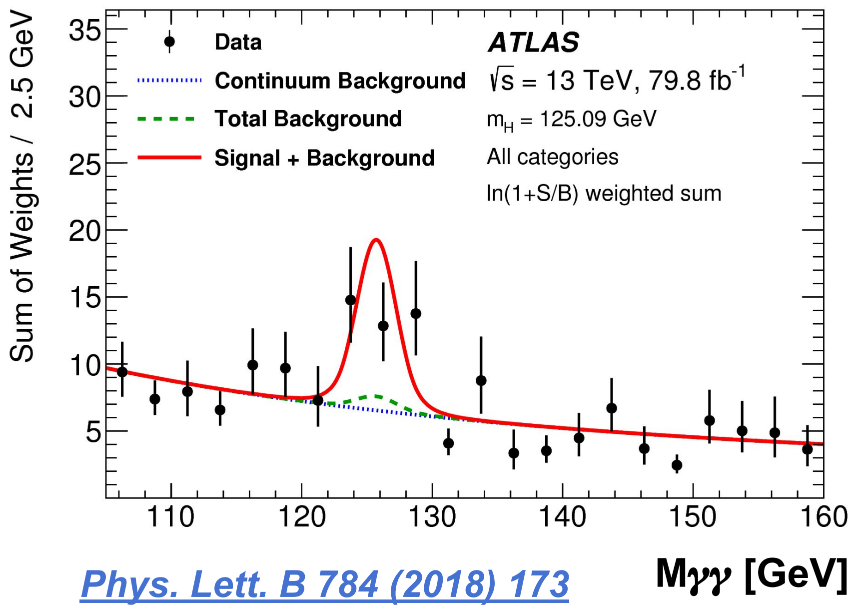
Artist's rendition of the Google quantum processor

We have applied quantum machine learning to two LHC flagship analyses:
ttH ($H \to \gamma\gamma$) and $H \to \mu\mu$

ttH (H → γγ) analysis at the LHC

The observation of ttH production (Higgs boson production in association with a top quark pair) by ATLAS and CMS at the LHC directly confirmed the interaction between the Higgs boson and the top quark, which is the heaviest known fundamental particle





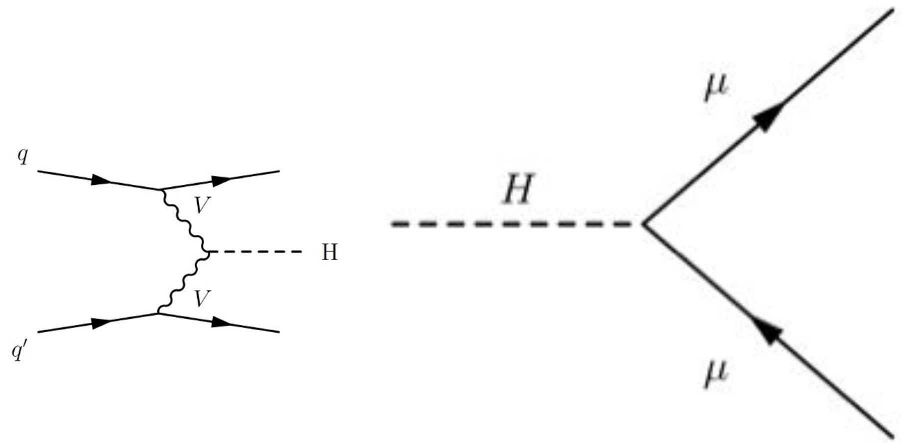
[Phys. Lett. B 784 (2018) 173]

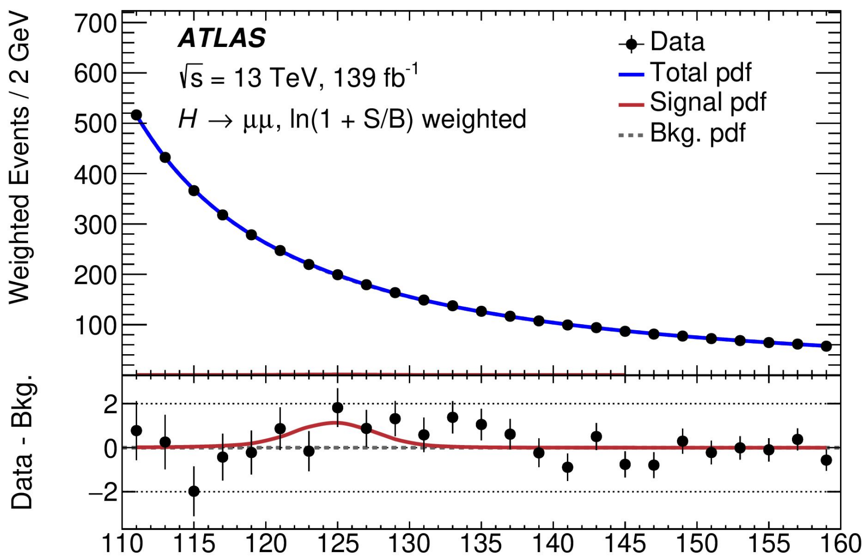
- **Using Boosted Decision Tree (BDT, a classical machine learning technique) with XGBoost package, the ATLAS Collaboration observes the ttH (H→γγ) process**
- **Our study performs the event classification of the ttH (H→γγ) analysis (hadronic channel) with delphes simulation samples and quantum machine learning**

H → $\mu\mu$ analysis at the LHC

Although the coupling between the Higgs boson and 3rd-generation fermions has been observed, currently the coupling between the Higgs boson and 2nd-generation fermions is under intensive investigation. H→$\mu\mu$ is the most promising process to observe such a coupling by ATLAS and CMS at the LHC

ATLAS: 2.0σ, Phys. Lett. B 812, 135980 (2021)
CMS: 3.0σ, JHEP 01 148 (2021)





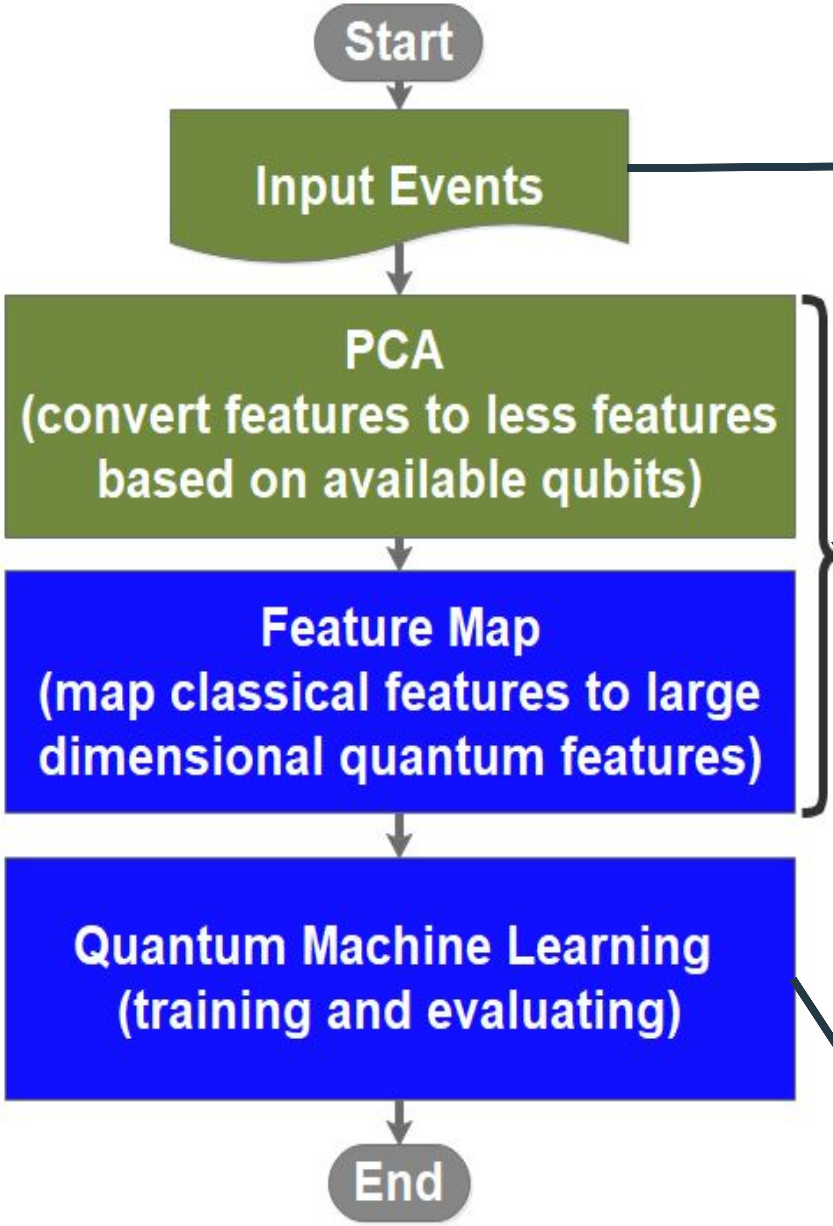
[Phys. Lett. B 812 (2021) 135980](#)

- **Using Boosted Decision Tree (BDT, a classical machine learning technique) with XGBoost package, the ATLAS Collaboration searches for the H→$\mu\mu$ decay**
- **Our study performs the event classification of the H→$\mu\mu$ analysis (VBF channel) with delphes simulation samples and quantum machine learning**

Delphes Simulation

- Delphes [JHEP 02 057 (2014)] is a program that performs fast simulation of multipurpose detectors' response
- It reconstructs physics objects for physics analyses, including photons, electrons, muons, jets and missing transverse momentum

Our Workflow for Quantum Machine Learning



**e.g. in ttH (H→γγ) analysis,
number of features (variables) = 23**

- **PCA: Principal Component Analysis method is used to convert/combine features into fewer features to match the number of qubits (features = variables)**
- **FeatureMap: encoding classical variables to quantum states. Currently, in our studies, number of encoded variables has to equal number of qubits (e.g. 5, 10, 20)**

Variational Quantum Classifier method, for example

Method 1

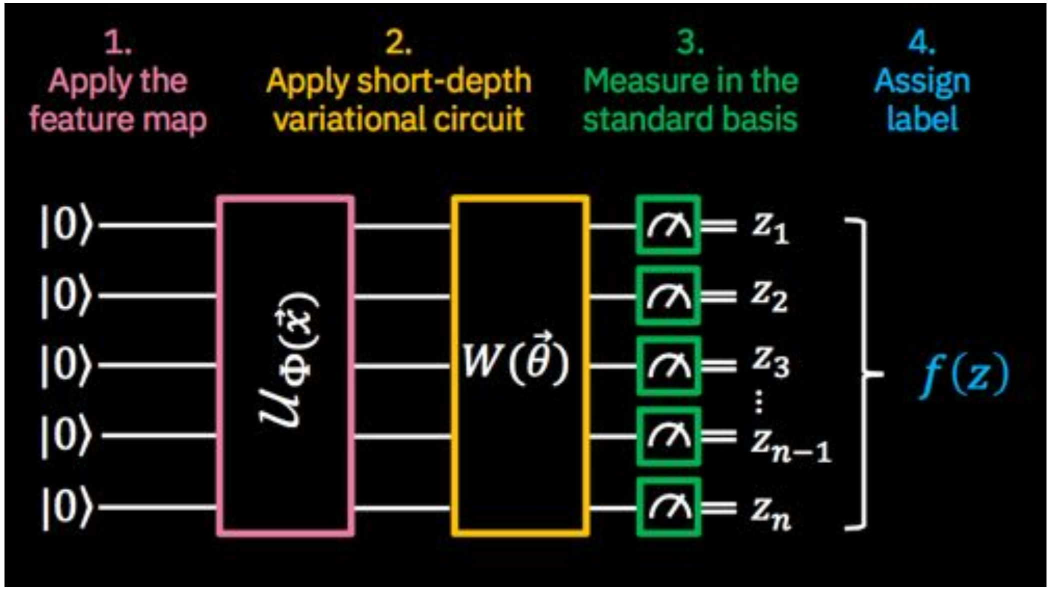
Employing Variational Quantum Classifier for ttH ($H \to \gamma\gamma$) and $H \to \mu\mu$ analyses

Method 1: Variational Quantum Classifier (VQC)

- In 2018, a Variational Quantum Classifier method was introduced by IBM, published in Nature 567 (2019) 209.
- The Variational Quantum Classifier method can be summarized in four steps.

Method 1: Variational Quantum Classifier (VQC)

- 1. Apply feature map circuit $U_{\Phi(\vec{x})}$ to encode input data $\vec{x}$ into quantum state $|\Phi(\vec{x})\rangle$
- 2. Apply short-depth quantum variational circuit W(θ) which is parameterized by gate angles θ
- 3. Measure the qubit state in the standard basis (standard basis: $|0\rangle$, $|1\rangle$ for 1 qubit; $|00\rangle$, $|01\rangle$, $|10\rangle$, $|11\rangle$ for 2 qubits; ...)
- 4. Assign the label ("signal" or "background") to the event through the action of a diagonal operator f in the standard basis



- We have two independent sets of events: one for training and one for testing
- During the training phase, a set of events are used to train the circuit W(θ) to reproduce correct classification
- Using the optimized W(θ), the testing events are used for evaluation

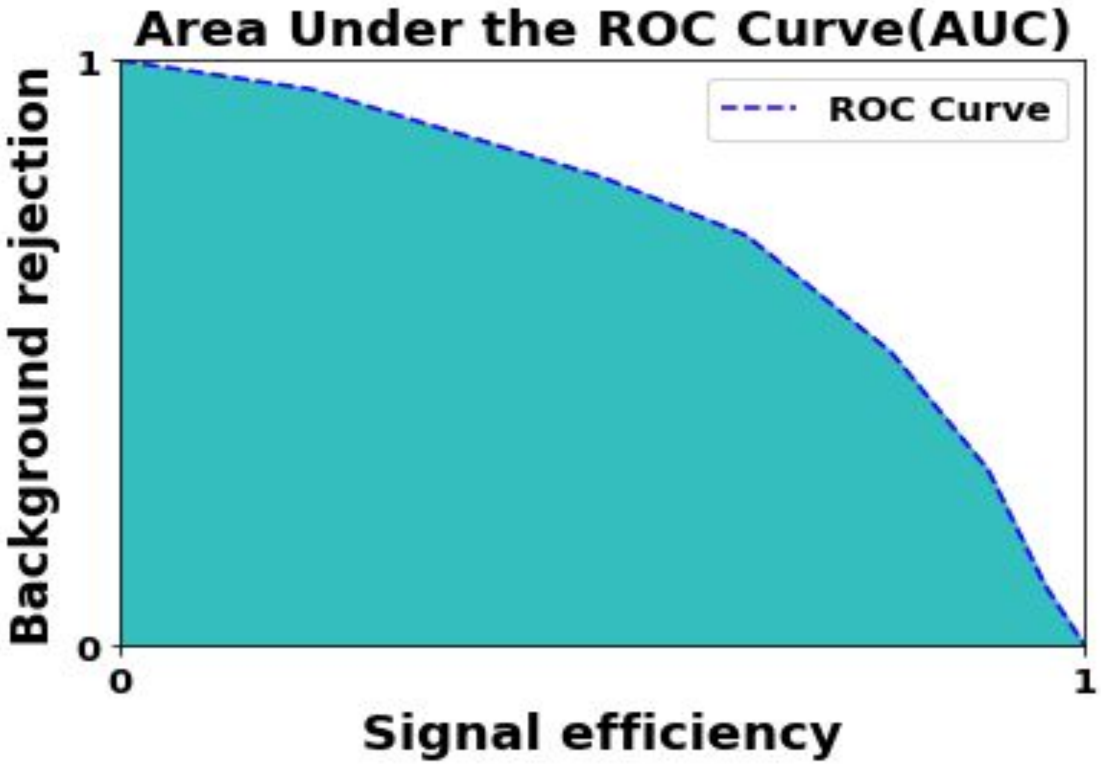
Method 1: Employing VQC (Variational Quantum Classifier) with IBM Q simulator for ttH ($H \to \gamma\gamma$) analysis and $H \to \mu\mu$ analysis

Using 10 qubits, we successfully finished training and testing 100 events with IBM Qiskit QASM simulator (where '100' events means 100 training events and 100 testing events).

- **Q simulator (Quantum circuits simulator): here IBM Qiskit QASM simulator is used. This simulation incorporates the hardware noise**
- **Quantum circuits are optimized to best fit the constraints imposed by hardware (e.g. qubit connectivity, hardware noise) and the nature of data**

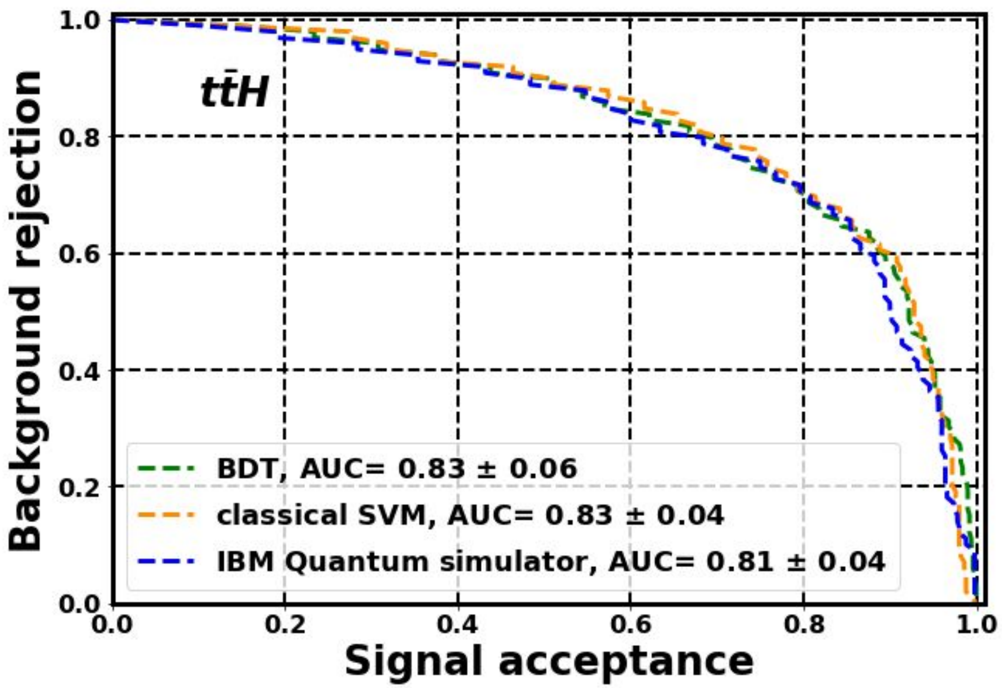
Method 1: Employing VQC (Variational Quantum Classifier) with IBM Q simulator for ttH ($H \to \gamma\gamma$) analysis and $H \to \mu\mu$ analysis

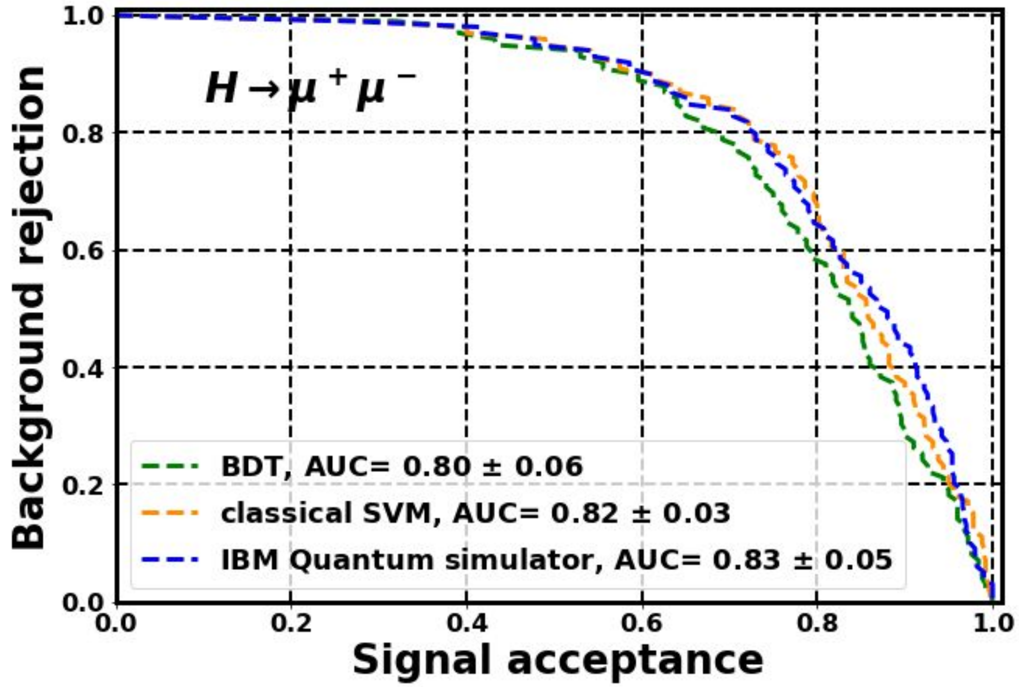
- **Definitions**
  - **ROC (Receiver Operating Characteristic) Curve**: a graph showing background rejection vs signal efficiency.
  - **AUC**: Area Under the ROC Curve, for quantifying discrimitation power of machine learning algorithms



ROC curves and AUC are standard metrics for machine learning applications

Method 1: Employing VQC (Variational Quantum Classifier) with IBM Q simulator for ttH ($H \to \gamma\gamma$) analysis and $H \to \mu\mu$ analysis





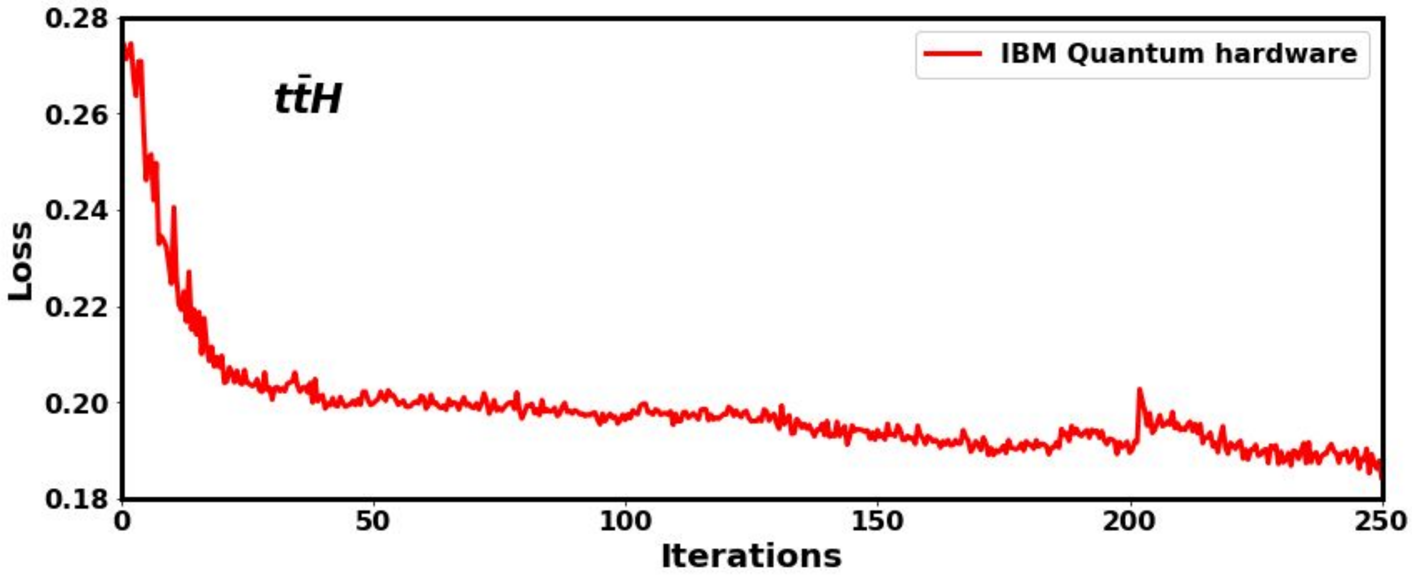
Using ttH analysis dataset (100 events, 10 variables) and $H \to \mu\mu$ analysis dataset (100 events, 10 variables), Variational Quantum Classifier on simulator (blue) performs similarly with classical BDT (green) and classical SVM (yellow). (Results are average over ten datasets)

	AUC (ttH)	AUC ($H \to \mu\mu$)
VQC	0.81	0.83
BDT	0.83	0.80
SVM	0.83	0.82

Method 1: Employing VQC (Variational Quantum Classifier) with IBM hardware for ttH ($H \rightarrow \gamma\gamma$) analysis and $H \rightarrow \mu\mu$ analysis

- With the help of IBM Research Zurich, Fermilab and BNL, we have carried out a number of jobs on the IBM superconducting quantum computers (ibmq_boeblingen, a 20-qubit machine and ibmq_paris, a 27-qubit machine). In each job, 10 qubits of the quantum computer are used to study 100 training events and 100 testing events.
  - The hardware running time for 100 events is 200 hours
- For each analysis, due to current limitation of hardware access time, we apply the Variational Quantum Classifier method to one dataset on quantum hardware (rather than ten datasets on quantum simulator)

Method 1: Employing VQC (Variational Quantum Classifier) with IBM hardware for ttH ($H \to \gamma\gamma$) analysis and $H \to \mu\mu$ analysis



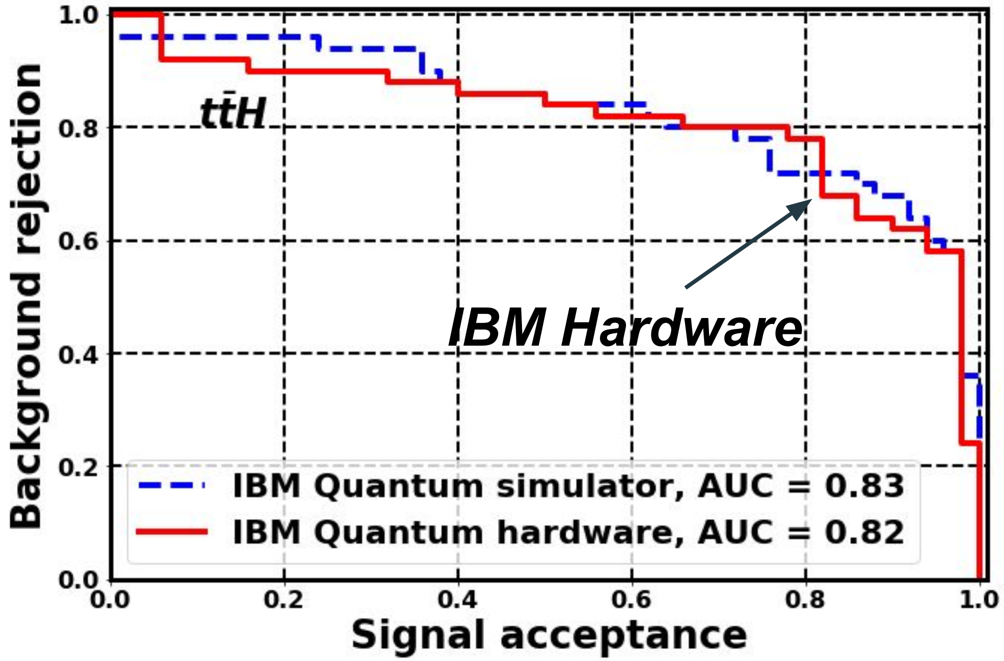
Red: Quantum Hardware

Loss: the mean of the squared differences between the output scores from the quantum algorithm and the ideal scores

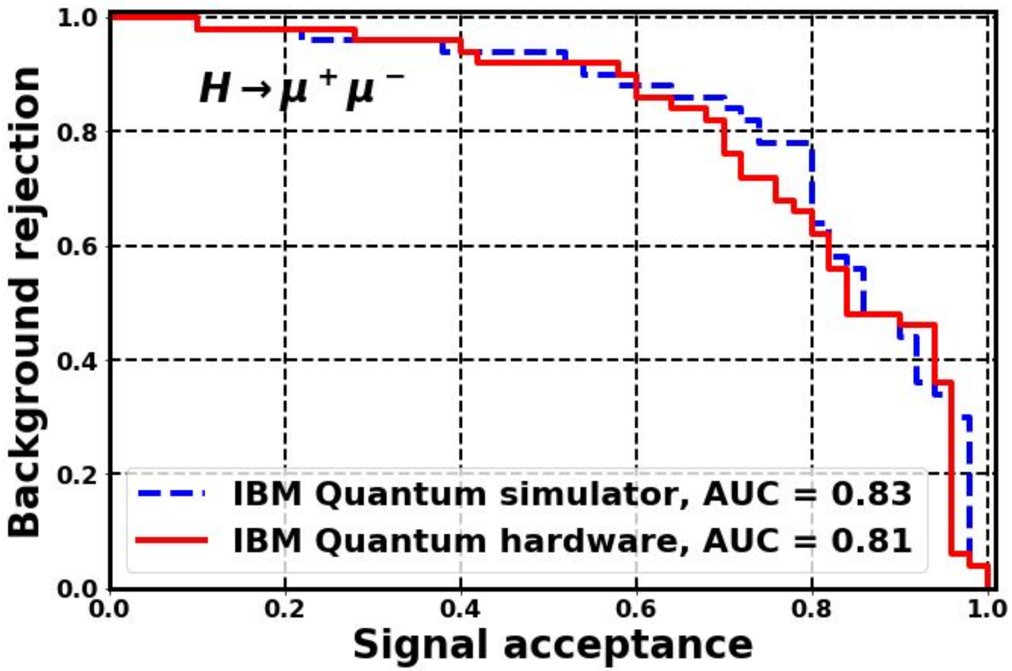
- **The hardware loss (red) is decreasing with the increase of number of iterations*. This indicates that the Quantum Computer has the ability to learn how to differentiate between the signal and the background for a HEP analysis.**

*** "iteration" indicates the number of times the algorithm's parameters are updated in training**

Method 1: Employing VQC (Variational Quantum Classifier) with IBM hardware for ttH ($H \rightarrow \gamma\gamma$) analysis and $H \rightarrow \mu\mu$ analysis



hardware AUC = 0.82, simulator AUC = 0.83



hardware AUC = 0.81, simulator AUC = 0.83

- **Using ttH analysis dataset (100 events, 10 variables) and $H \rightarrow \mu\mu$ analysis dataset (100 events, 10 variables), with 250 iterations, the result of Variational Quantum Classifier from Quantum Hardware and result from Quantum Simulator are in good agreement.**

* "iteration" indicates the number of times the algorithm's parameters are updated in training

Method 2

Employing Quantum Support Vector Machine (QSVM) Kernel method for ttH ($H \rightarrow \gamma\gamma$) analysis

Method 2: Quantum SVM Kernel method

- ***Quantum SVM Kernel method*** ***(introduced by IBM, published in Nature 567 (2019) 209):***
  - *map classical data $\vec{x}$ to a quantum state $|\Phi(\vec{x})\rangle$ using a Quantum Feature Map function;*
  - *calculate the similarity between any two data events ("kernel entry") as $K(\vec{x}_1,\vec{x}_2)=|\langle\Phi(\vec{x}_1)|\Phi(\vec{x}_2)\rangle|^2$ using a quantum computer;*
  - *then using the kernel entries to find an optimal separating hyperplane that separates signal from background.*

map classical data

$\vec{x}_1 \rightarrow |\Phi(\vec{x}_1)\rangle$

$\vec{x}_2 \rightarrow |\Phi(\vec{x}_2)\rangle$

$\vec{x}_3 \rightarrow |\Phi(\vec{x}_3)\rangle$

…

calculate kernel entries

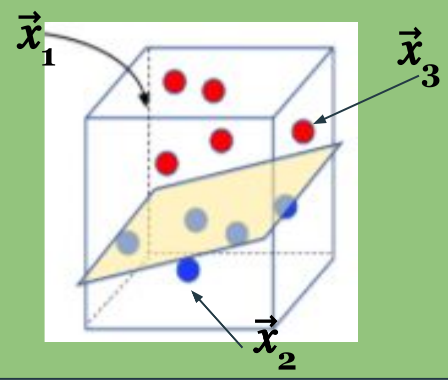
$K(\vec{x}_1,\vec{x}_2)=|\langle\Phi(\vec{x}_1)|\Phi(\vec{x}_2)\rangle|^2$

$K(\vec{x}_1,\vec{x}_3)=|\langle\Phi(\vec{x}_1)|\Phi(\vec{x}_3)\rangle|^2$

$K(\vec{x}_2,\vec{x}_3)=|\langle\Phi(\vec{x}_2)|\Phi(\vec{x}_3)\rangle|^2$

…

find separating hyperplane



Method 2: Employing Quantum SVM Kernel method with quantum simulators for ttH ($H \rightarrow \gamma\gamma$) analysis

We are performing the ttH analysis using QSVM Kernel method with up to 20 qubits:

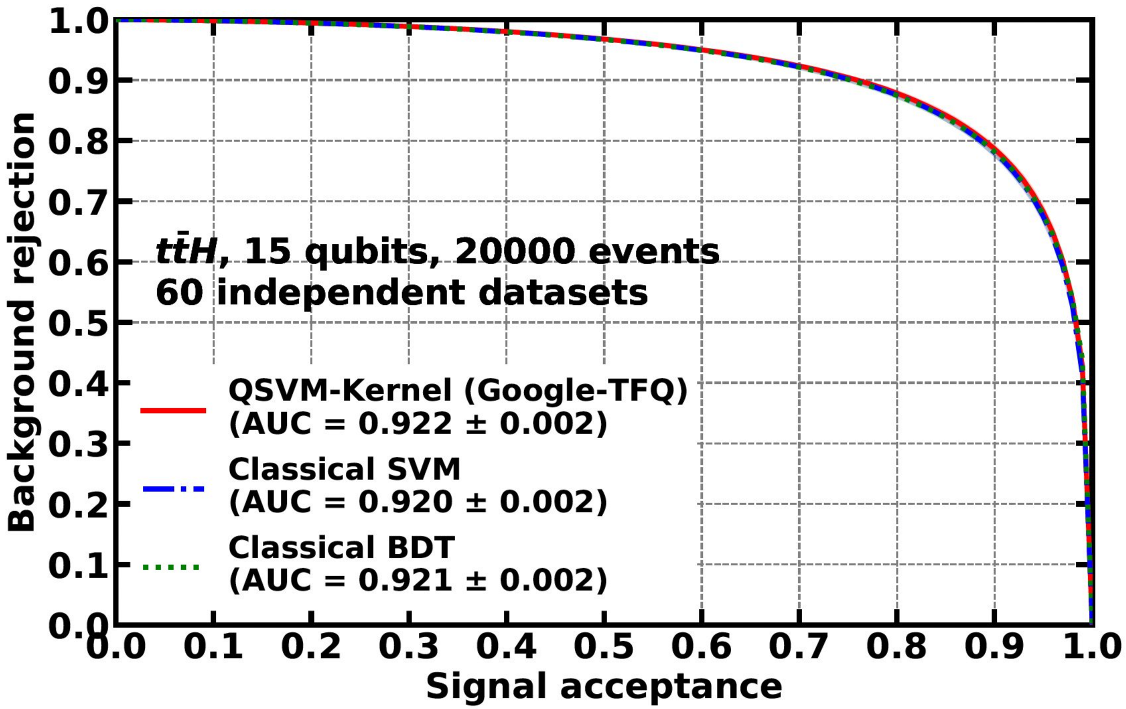
- ***A customized FeatureMap is used. The quantum FeatureMap circuit encodes classical data to a quantum state***
- ***Grid-Search with cross-validation* is used to optimize the QSVM Kernel performance***

* **Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample**
https://machinelearningmastery.com/k-fold-cross-validation/

Method 2: Employing Quantum SVM Kernel method with quantum simulators for ttH ($H \rightarrow \gamma\gamma$) analysis

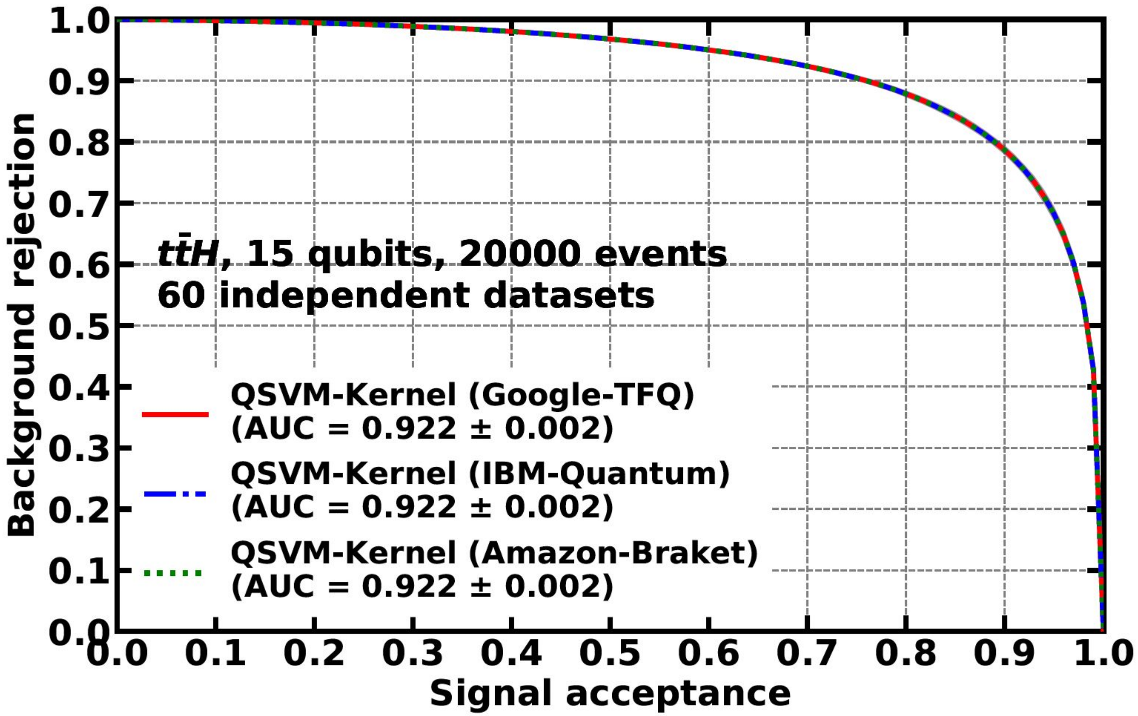
- ***Our group has implemented the QSVM Kernel algorithm using the qsim Simulator from the Google TensorFlow Quantum framework, the Statevector Simulator from the IBM Qiskit framework and the Local Simulator from the Amazon Braket framework***
  - ***These simulators represent the ideal quantum hardware that performs infinite measurement shots and experiences no hardware device noise***
  - ***We have overcome the challenges of heavy computing resources in the use of up to 20 qubits and up to 50000 events on the quantum computer simulators***

Method 2: Employing Quantum SVM Kernel method with quantum simulators for ttH ($H \rightarrow \gamma\gamma$) analysis



- For 15 qubits, using ttH analysis dataset (20000 events), QSVM Kernel on simulator (red) achieves similar performances with classical SVM (blue) and classical BDT (green). (Results are averaged over sixty datasets)

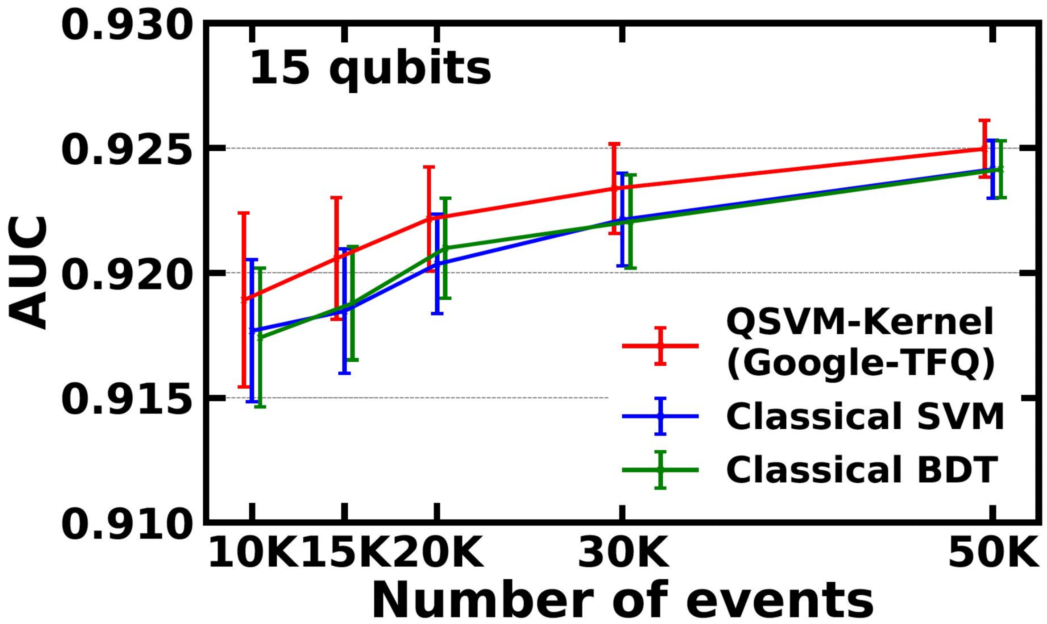
Method 2: Employing Quantum SVM Kernel method with quantum simulators for ttH (H → $\gamma\gamma$) analysis



- **For 15 qubits, using ttH analysis dataset (20000 events), Google qsim simulator (red), IBM statevector simulator (blue), and Amazon local simulator (green) provide identical performances for QSVM Kernel method**

Method 2: Employing Quantum SVM Kernel method with quantum simulators for ttH ($H \rightarrow \gamma\gamma$) analysis

AUC vs number of events

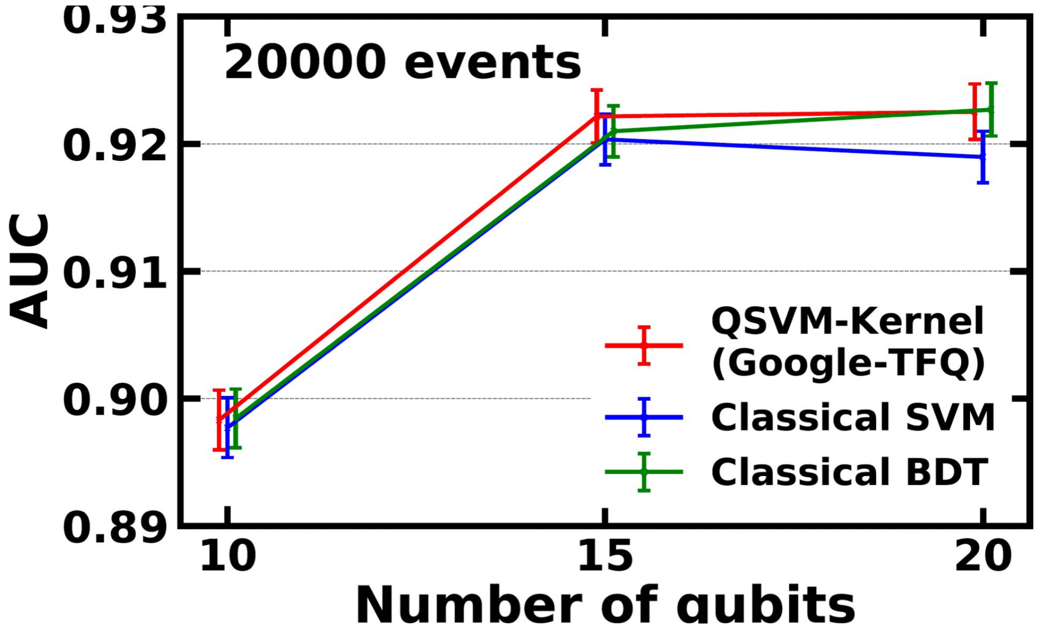


- QSVM Kernel method and noiseless simulators enable us to work with a larger number of events.

- For 15 qubits, using ttH analysis dataset (10000-50000 events), QSVM Kernel on simulator (red) achieves similar performances with classical SVM (blue) and classical BDT (green).

Method 2: Employing Quantum SVM Kernel method with quantum simulators for ttH ($H \rightarrow \gamma\gamma$) analysis

AUC vs number of qubits



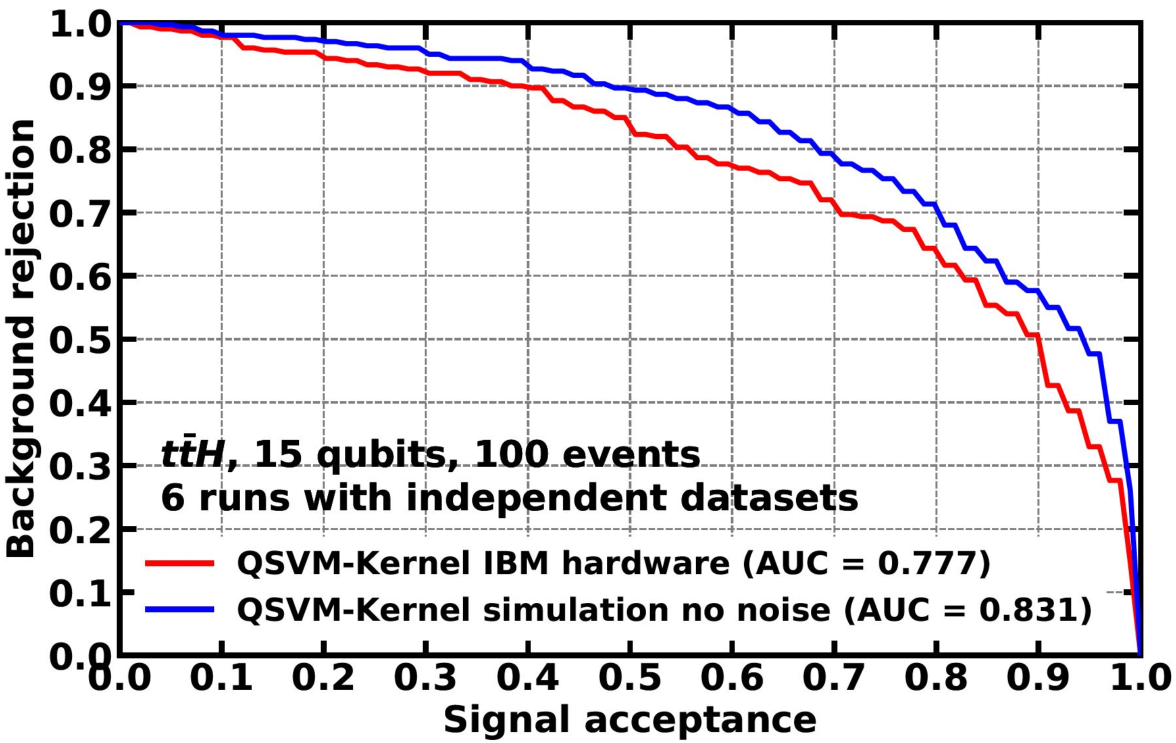
- QSVM Kernel method and noiseless simulators also enable us to work with a larger number of qubits.

- For 10-20 qubits, using ttH analysis dataset (20000 events), QSVM Kernel on simulator (red) achieves similar performances with classical SVM (blue) and classical BDT (green).

Method 2: Employing QSVM Kernel with IBM hardware (ibmq_paris, a 27-qubit machine) for ttH ($H \rightarrow \gamma\gamma$) analysis

- ***We have also been running the QSVM Kernel algorithm on quantum computer hardware provided by IBM (based on superconducting circuits)***
  - *to assess the quantum machine learning performances on today's noisy quantum computer hardware*
  - *due to current limitation of access time on imbq_paris, we only process six datasets of 100 training events and 100 testing events*
  - *for the six datasets, the average hardware running time is approximately 680 minutes per run*

Method 2: Employing QSVM Kernel with IBM hardware (ibmq_paris, a 27-qubit machine) for ttH ($H \rightarrow \gamma\gamma$) analysis



hardware AUC = 0.777

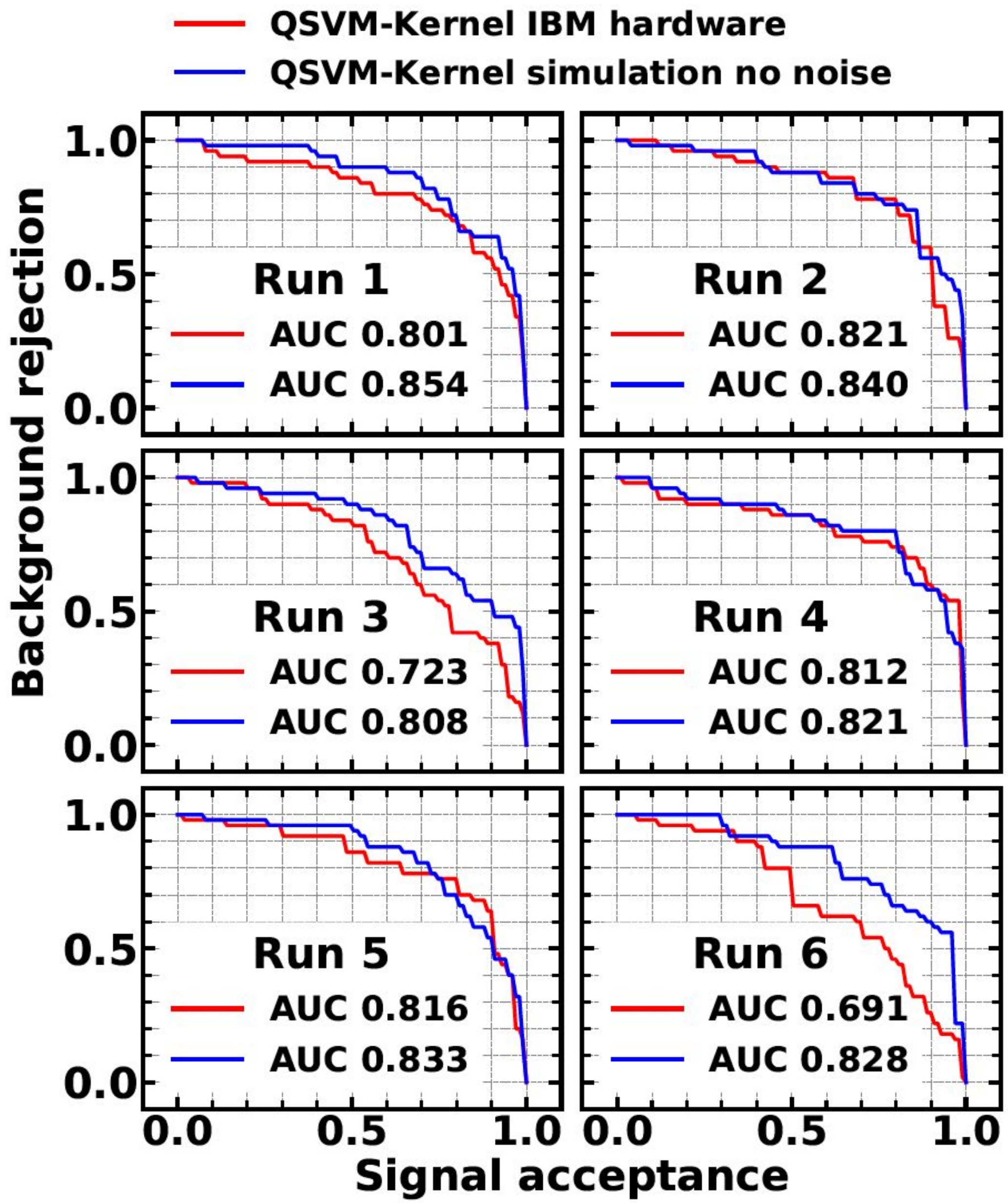
simulator AUC = 0.831

- Using ttH analysis dataset (100 events, 15 variables), the QSVM Kernel results on the Quantum Hardware (15 qubits) are promising and approaching the QSVM Kernel results on Quantum Simulator (the difference is likely due to effect of hardware noise)

Method 2: Employing QSVM Kernel with IBM hardware (ibmq_paris, a 27-qubit machine) for ttH ($H \rightarrow \gamma\gamma$) analysis

- ROC curves for each of the six runs:

- The effect of quantum hardware noise seems to fluctuate among the runs



Method 3

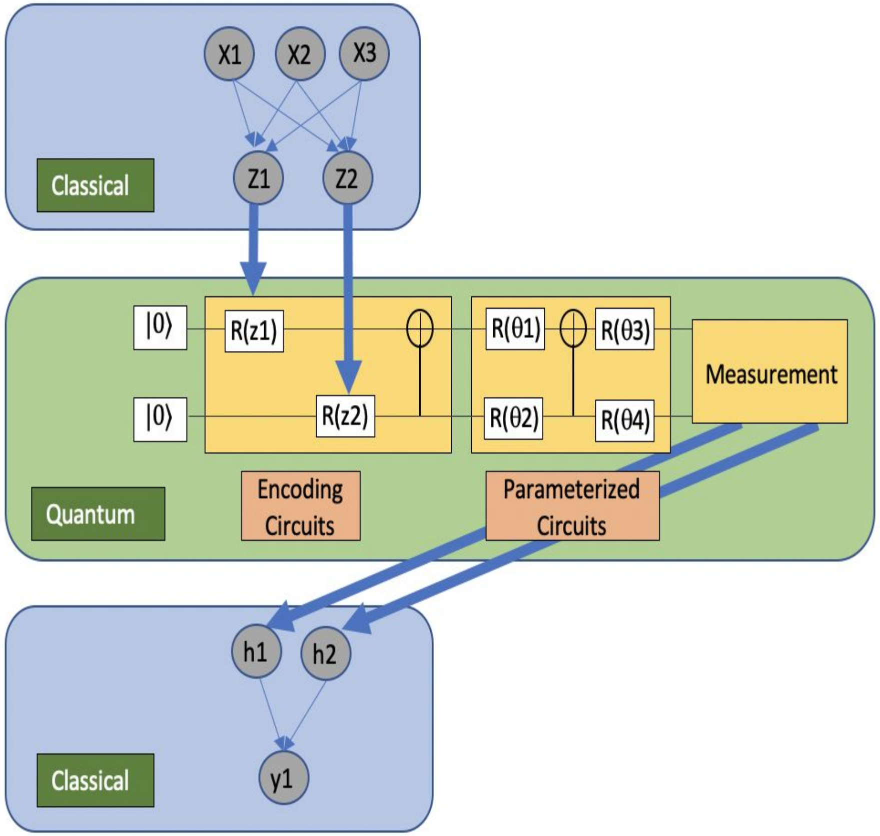
Employing Quantum Neural Network for ttH ($H \rightarrow \gamma\gamma$) analysis

Method 3: Quantum Neural Network (QNN)

- ***Quantum neural networks (QNNs): combining neural network algorithms and quantum computing***
  - ***Perform the computational intensive part of a neural network algorithm on a quantum computer with the aim of better efficiency and performance***
- ***Many QNN models have been recently studied in the field of quantum machine learning, for example, using Google Tensorflow quantum library and IBM Qiskit library***

Method 3: Hybrid Quantum Neural Network (QNN)

We have been exploring a hybrid QNN of three layers:



- ***Classical layer 1: transform input data so that its number of outputs matches number of qubits (PCA is no longer necessary)***
- ***Quantum layer (the core part): encode classical data into a quantum state, apply variational circuit containing trainable parameters, and measure the quantum state***
- ***Classical layer 2: convert the measurement of qubits to classification labels***

Three layers are trained together to maximize the overall performance

Method 3: Employing QNN for ttH (H $\rightarrow \gamma\gamma$) analysis

- **We employ the hybrid quantum neural network method for the ttH (H $\rightarrow \gamma\gamma$) analysis, using:**
  - **Google quantum computer simulator (using Google Cirq and TensorFlow Quantum libraries)**
  - **IBM quantum computer simulator and hardware (using IBM Qiskit libraries)**

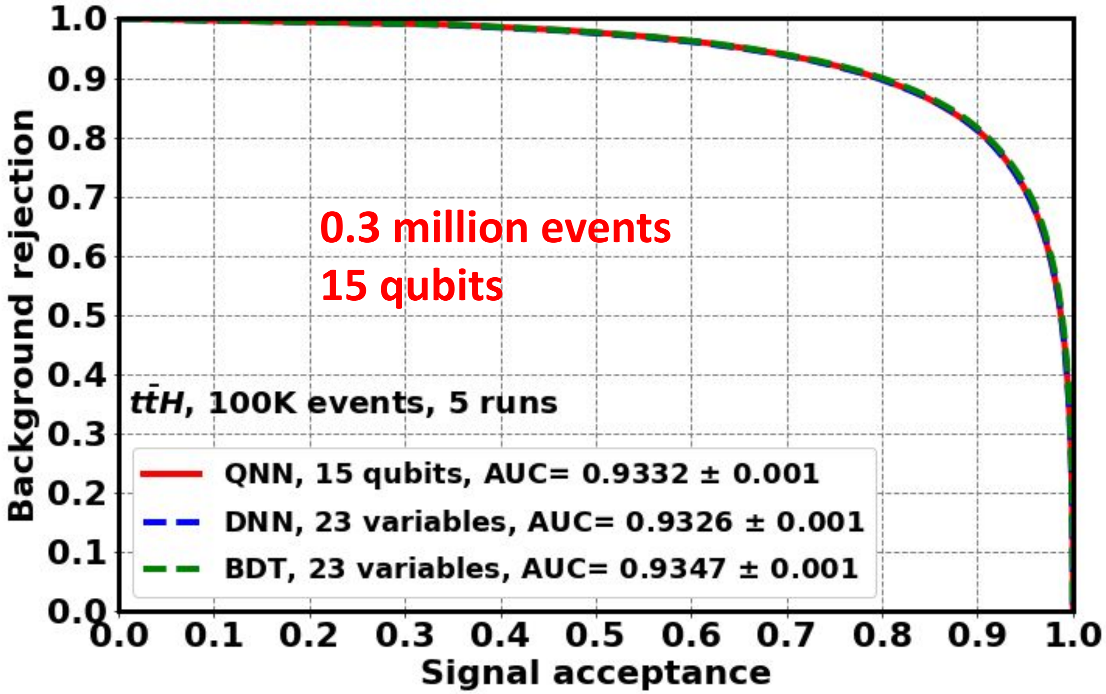
Method 3: Employing QNN with Google simulator for ttH ($H \to \gamma\gamma$) analysis

Work under development

- In the official ATLAS ttH ($H \to \gamma\gamma$) analysis with LHC data, ~0.5 million events are used for training+validation+testing

- On Google simulator, we recently apply the QNN to a ttH analysis dataset (simulation data using Delphes) of ~0.3 million events (splitting between training, validation and testing samples), which is similar to the sample size used in the official ATLAS data analysis

Method 3: Employing QNN with Google simulator for ttH ($H \rightarrow \gamma\gamma$) analysis



Work under development

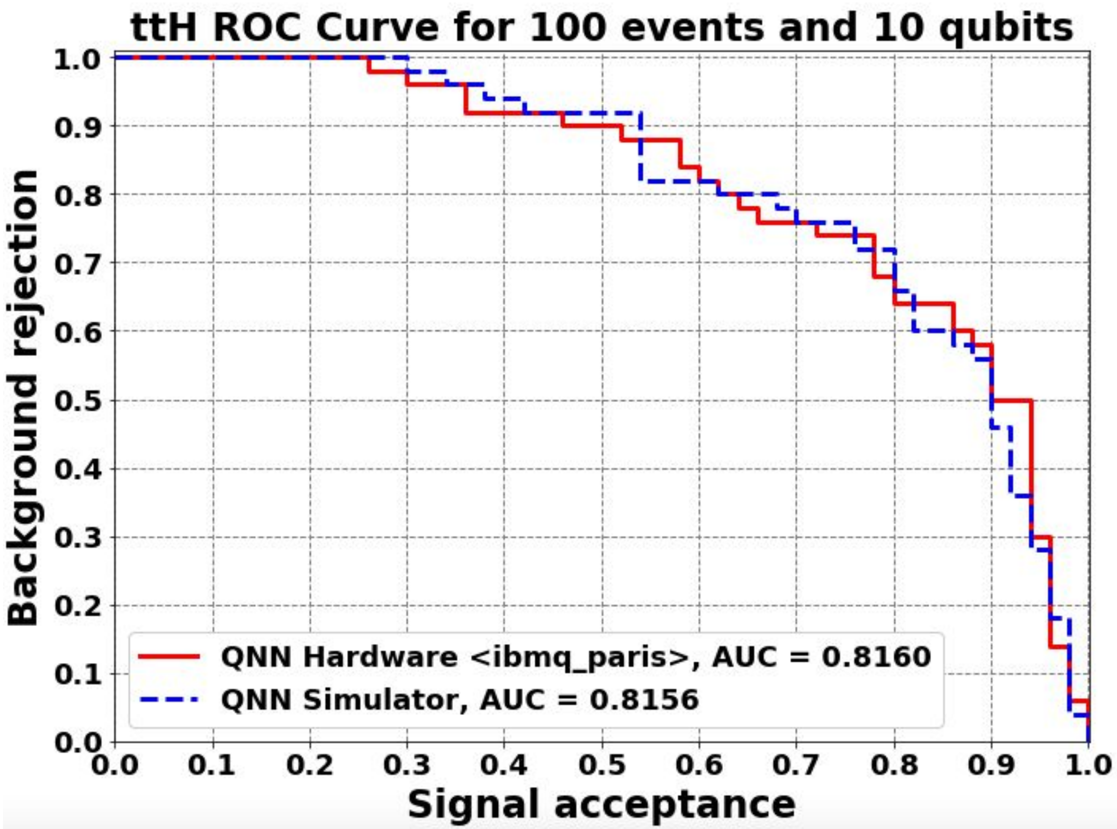
QNN AUC: 0.9332

DNN AUC: 0.9326

BDT AUC: 0.9347

- **Using the ttH analysis dataset with 0.3 million Delphes events and 15 qubits, QNN on simulator (red) now performs similarly with classical Deep Neural Network (DNN) (blue) and classical BDT (green).**
- **The optimization of this QNN is still under development (e.g. more qubits), and we hope to achieve quantum advantage with large datasets**

Method 3: Employing QNN with IBM Q hardware (10 qubits) for ttH ($H \rightarrow \gamma\gamma$) analysis



- 100 events, 10 qubits, 1 run
- QNN hardware: ibmq_paris
- QNN simulator: IBM simulator with no noise

	AUC (100 events)
Hardware	0.816
Simulator	0.816

- The performance with quantum hardware is close to the performance with no-noise simulation.
- Hardware running time: 384 hours

Summary (part 1)

- **We form** an international and interdisciplinary collaboration with the Department of Physics and Department of Computer Sciences of University of Wisconsin, CERN Quantum Technology Initiative, IBM Research Zurich and **IBM T.J. Watson Research Center**, Fermilab Quantum Institute, BNL Computational Science Initiative, State University of New York at Stony Brook, Quantum Computing and AI research of Amazon Web Services

- Although the era of efficient quantum computing may still be years away, we have made promising progress and obtained preliminary results in applying quantum machine learning to High Energy Physics. **A PROOF OF PRINCIPLE.**

Summary (part 2)

- **We have employed 3 methods of Quantum Machine Learning**

    - **Method 1: VQC-Variational Quantum Classifier**

      (accepted by *J. Phys. G: Nucl. Part. Phys.* https://doi.org/10.1088/1361-6471/ac1391)

    - **Method 2: QSVM-Quantum Support Vector Machine Kernel method**

      (arXiv:2104.05059)

    - **Method 3: QNN-Quantum Neural Network**

- **We have applied the three methods to two LHC HEP flagship analyses (ttH ($H \rightarrow \gamma\gamma$) and $H \rightarrow \mu\mu$) with Delphes simulation events.**

Summary (part 3)

- **Results from Quantum Simulator**
  - **With 100 events and 10 qubits, method 1: VQC (Variational Quantum Classifier) method on IBM Quantum Simulator performs similarly to classical BDT and classical SVM.**
  - **With up to 50000 events and up to 20 qubits, method 2, QSVM (Quantum Support Vector Machine) Kernel method on Google, IBM and Amazon Quantum Simulators performs similarly to classical BDT and classical SVM in the ttH (H $\rightarrow \gamma\gamma$) channel.**
  - **With 0.3 million events and 15 qubits, method 3, QNN (Quantum Neural Network) method on Google Quantum Simulator performs similarly to classical BDT and classical DNN in the ttH (H $\rightarrow \gamma\gamma$) channel.**
- **Results from Quantum Hardware**
  - **With 100 events, for method 1 (10 qubits), method 2 (15 qubits), method 3 (10 qubits), IBM Quantum Hardware and IBM Quantum Simulator show comparable performance.**

Summary (part 4)

- Our results (on both simulators and hardware) demonstrate quantum machine learning on the gate-model quantum computers has the ability to differentiate signal and background in realistic physics datasets

- Future developments:
  - We will investigate further and hopefully will see soon quantum machine learning outperforms classical machine learning, in particular, when more qubits are utilized
  - Furthermore, future quantum computers might offer speed ups in quantum machine learning which could be critical for the HEP community

Challenges ahead

- Difficulties at present:
  - Only 100 events are used in hardware jobs
    - Limited access time
  - Only 10-15 qubits are used in hardware jobs
    - So far circuit length and number of CNOT gates are limited in our present study.
- To use Quantum Computer Hardware for Machine Learning in future High-Luminosity LHC physics analyses, we need to extend our studies to larger event sample sizes and more qubits
- As of today, the maximal number of hardware qubits that I know of: 65 (IBM) and 54 (Google)
- To demonstrate that future Quantum Computers offer speed up in Quantum Machine Learning

Prediction

- I am confident that, in the near future,

  the quantum machine learning methods can demonstrate, in quantum simulation, the quantum advantage with a larger number of qubits (e.g. greater than 30 qubits).

  This is in the context of application to High Energy Physics data analysis.

Prediction

- From the roadmap presented by IBM and Google, it is expected that quantum hardware in the future will reduce noise and achieve a performance close to noiseless quantum simulators. In addition, they are working hard to speed up the quantum hardware running time.
- With the large investments in quantum computing and fierce international competitions in technology, this expectation is realistic.

Other studies on Quantum Machine Learning application for HEP that we know of

- Maria Spiropulu et al, “Solving a Higgs optimization problem with quantum annealing for machine learning”, Nature 550, 375 (2017)
- Koji Terashi et al, “Event Classification with Quantum Machine Learning in High-Energy Physics”, Comput. Softw. Big Sci. 5, 2 (2021)
- Davide Zuliani, Donatella Lucchesi, et al, “Quantum Machine Learning for jet tagging @ LHCb”, PyHEP 2021 (virtual) workshop
- Plus more...

The following members from the Wisconsin group would answer your technical questions

Wen Guan

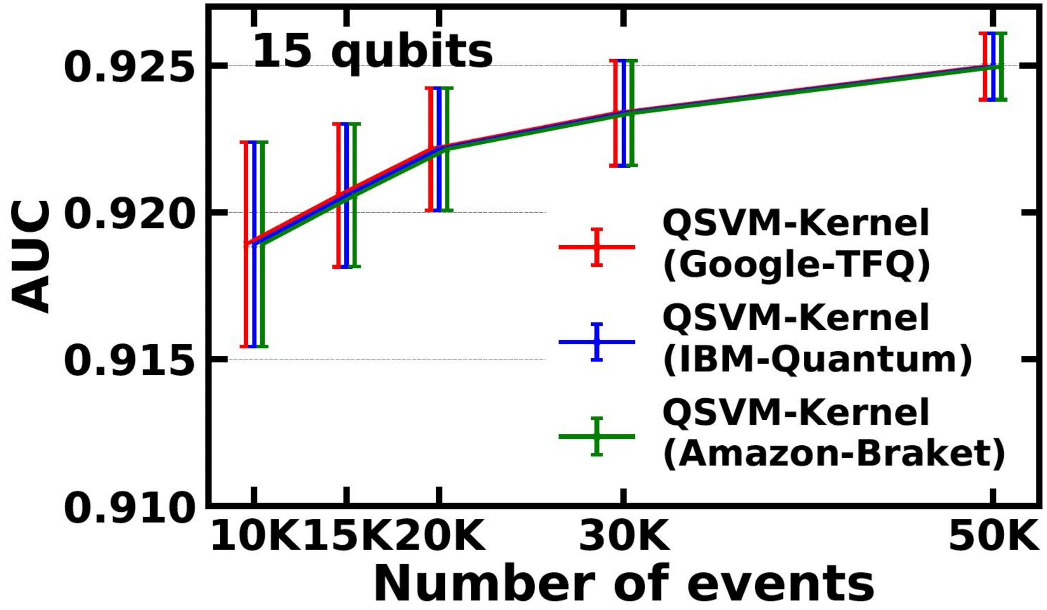
Shaojun Sun

Chen Zhou

BACKUP SLIDES

Method 2: Employing Quantum SVM Kernel method with quantum simulators for ttH ($H \rightarrow \gamma\gamma$) analysis

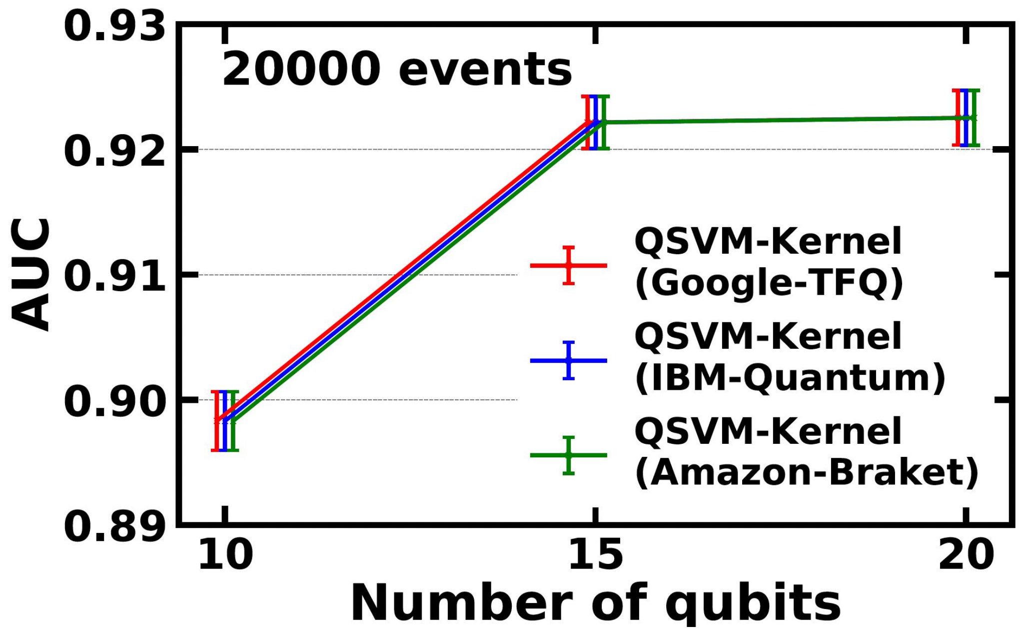
AUC vs number of events



- **Using ttH analysis dataset (10000-50000 events, 15 variables), Google qsim simulator (red), IBM statevector simulator (blue), and Amazon local simulator (green) provide identical performances for QSVM Kernel method**

Method 2: Employing Quantum SVM Kernel method with quantum simulators for ttH ($H \rightarrow \gamma\gamma$) analysis

AUC vs number of qubits



- Using ttH analysis dataset (20000 events, 10-20 variables), Google qsim simulator (red), IBM statevector simulator (blue), and Amazon local simulator (green) provide identical performances for QSVM Kernel method