Advanced Quantum Machine Learning with Qiskit

Course Number: QNTM-201

Duration: 2 days

Course Description

This Quantum course teaches technical practitioners how to build, train, and deploy Quantum Machine Learning (QML) models using advanced Qiskit toolkits. Learners construct parameterized 2x2 quantum blocks using the Cayley Transform, a mapping technique that guarantees strict unitarity during gradient descent so that trained circuits remain physically valid. The curriculum applies Simultaneous Perturbation Stochastic Approximation (SPSA) to train circuits on classical GPUs before deploying them to simulated noisy hardware, and addresses the two central scaling constraints practitioners encounter in the field: the memory limits of classical simulators and the qubit limits of near-term hardware. Participants implement circuit cutting, classical stitching using Quasi-Probability Decomposition (QPD), and Matrix Product State tensor networks to work around both constraints. The course culminates in execution on actual IBM Quantum hardware, where participants apply Zero Noise Extrapolation (ZNE) to recover signal accuracy lost to hardware noise.
Throughout, the course emphasizes practical, near-term applications relevant to fraud detection and anomaly detection, including use cases applicable to federal financial-integrity programs such as improper-payment and claims-fraud detection. As agencies build the quantum-literate technical staff called for under quantum-readiness workforce planning, this course provides hands-on grounding in how quantum circuits are trained, scaled, and executed under real hardware noise conditions, rather than a conceptual survey.

Skills Gained

By the end of this course, participants will be able to:

- Construct parameterized 2x2 quantum blocks using the Cayley Transform to ensure strict unitarity during model training.
- Train Quantum Machine Learning circuits using SPSA optimization and classical backpropagation on local GPUs.
- Implement circuit cutting and classical stitching using Quasi-Probability Decomposition (QPD) to execute large circuits on limited hardware.
- Apply Zero Noise Extrapolation (ZNE) to mitigate hardware noise during quantum inference.
- Deploy trained quantum circuits to IBM Quantum hardware to solve pattern recognition tasks, such as image classification and fraud detection.

Who Can Benefit

This course is designed for:

- Software engineers and data scientists who have completed foundational quantum computing training.
- AI/ML practitioners looking to implement quantum architectures for complex pattern recognition.
- Federal agency technical staff and contractors applying QML to fraud-detection and anomaly-detection use cases.

Prerequisites

Participants should enter this course with:

- Completion of the Quantum Computing Foundations course (or equivalent foundational quantum knowledge).
- Proficiency in Python and familiarity with standard machine learning concepts.
- Basic understanding of PyTorch or TensorFlow workflows.

Organizational Objectives

This course assists organizations to:

- Build scalable QML pipelines that bypass classical memory limits using tensor networks and circuit cutting.
- Enable anomaly detection and pattern recognition capabilities for federal and enterprise clients using near-term quantum hardware.
- Reduce AI model training costs by equipping teams with quantum architectures that use exponentially fewer parameters while maintaining mathematical stability.
- Establish a bench of practitioners capable of executing complex quantum circuits on noisy hardware using advanced mitigation techniques.

- Build quantum-literate technical teams capable of applying QML architectures to fraud-detection and anomaly-detection use cases as agencies and enterprises expand quantum-readiness initiatives beyond cryptography alone.

Software

All participants require a laptop or desktop computer with a modern web browser and broadband internet.

Course Details

Module 1 – Parameterization and the Cayley Transform

By the end of this module, you will be able to design fully trainable quantum circuits using the Cayley Transform. You will also be able to explain how mapping real numbers to complex unitary matrices guarantees physical legality during gradient descent.

- The limitations of standard quantum gates in machine learning
- Constructing 2x2 Hermitian matrices using real parameters
- Applying the Cayley Transform to generate strict unitary gates
- Building custom CayleySingleQubitGate and CayleyTwoQubitGate classes in Qiskit
- Hands-on Lab: Define the Cayley Transform logic in Python and integrate custom parameterized gates into a Qiskit quantum circuit.

Module 2 – Circuit Training and SPSA Optimization

By the end of this module, you will be able to train parameterized quantum circuits using SPSA optimization on a local simulator. You will also be able to map continuous expectation values to discrete quantum measurements for ML loss functions.

- The division of labor: Aer Simulator (engine) vs. SPSA (brain)
- Mapping data labels to quantum expectation values
- Estimating gradients using Simultaneous Perturbation Stochastic Approximation (SPSA)
- Executing training loops on local CPUs/GPUs using Qiskit Aer
- Hands-on Lab: Build a 9-qubit image recognizer using Cayley blocks and train it locally using SPSA to distinguish vertical and horizontal lines.

Module 3 – Scaling with Tensor Networks and Tournament Trees

By the end of this module, you will be able to scale quantum circuit simulations beyond standard memory limits using Matrix Product States (MPS). You will also be able to design shallow circuit topologies to reduce depth and mitigate barren plateaus.

- The memory wall: exponential memory scaling in statevectors
- Using Matrix Product States (MPS) for linear memory scaling

- Designing Tournament Tree topologies for logarithmic depth
- Preventing overfitting in expressive quantum circuits using L2 regularization
- Hands-on Lab: Modify the linear chain topology to a Tournament Tree and retrain the circuit using the MPS simulator to observe memory efficiency.

Module 4 – Circuit Cutting and Classical Stitching

By the end of this module, you will be able to execute large quantum circuits on limited hardware by cutting the circuit into smaller fragments. You will also be able to reconstruct the global expectation value using Quasi-Probability Decomposition (QPD).

- The hardware wall: running 10-qubit circuits on 5-qubit machines
- Quasi-Probability Decomposition (QPD) math for CNOT gates
- Using IBM's qiskit-addon-cutting to physically slice circuits
- Classical stitching and the sampling overhead (variance explosion)
- Hands-on Lab: Physically cut a 10-qubit trained circuit into two 5-qubit fragments, execute them independently, and mathematically stitch the results back together.

Module 5 – Hardware Execution and Noise Mitigation

By the end of this module, you will be able to deploy classically trained quantum circuits to real IBM Quantum hardware. You will also be able to apply Zero Noise Extrapolation (ZNE) to recover accuracy lost to hardware noise.

- The noise floor and NISQ limitations
- Transpiling abstract circuits to physical basis gates
- Manual Zero Noise Extrapolation (ZNE): folding circuits to amplify noise
- Integrating ZNE with cut circuit fragments for final inference
- Hands-on Lab: Run the cut and stitched image recognizer on a simulated noisy backend, apply ZNE to the fragments, and plot the mitigated results against the ideal output.