VIKHRAM S
Department of Artificial Intelligence and Machine Learning | Saveetha Engineering College

Quantum Machine Learning: Emerging Directions in Hybrid Quantum–Classical AI

By Vikhram S, Advisor, Machine Learning Community – Tech Society, Saveetha Engineering College

Invited lecture delivered to the Department of Artificial Intelligence and Machine Learning, Saveetha Engineering College, May 16, 2025

Quantum computing and machine learning are two of the most talked-about fields in computer science today, and it is natural to ask what happens where they meet. This lecture does not ask you to take that promise on faith – instead, we will build a small, working example together: a classifier trained on a genuinely hard toy problem, first with an ordinary classical model, and then with a hybrid quantum-classical one, so that you can see exactly where the quantum part enters the picture and what it changes.

1. Why hybrid, and why now

No quantum computer today is large or reliable enough to run a full machine learning pipeline on its own. Instead, the practical approach used throughout this field, and in today's demonstration, is a **hybrid quantum-classical** architecture: a small quantum circuit handles one specific piece of the computation, usually the part where data is transformed into a new representation, while ordinary classical code handles everything else – data loading, optimisation, and evaluation. This hybrid approach is what makes quantum machine learning usable on today's early, noisy quantum hardware, often referred to as the NISQ era, short for Noisy Intermediate-Scale Quantum.

2. A test case designed to be hard: concentric circles

To make the comparison meaningful, we need a dataset that a simple classical model genuinely struggles with. Today's example uses a synthetic dataset of points arranged in two concentric circles – an inner ring belonging to one class and an outer ring belonging to the other. This shape is a classic stress test in machine learning because it is **not linearly separable**: no single straight line can divide the two classes, which puts a linear classifier such as logistic regression at a structural disadvantage from the start. The data is generated with scikit-learn's `make_circles` function, and each feature is rescaled into the range zero to pi using a min-max scaler – a detail that matters once we reach the quantum side, because quantum rotation gates expect their inputs in radians.

3. The classical baseline

Before introducing anything quantum, we first fit an ordinary logistic regression classifier to the same data, giving us a fair point of comparison. Logistic regression works by finding the single straight decision boundary that best separates the two classes. Against concentric circles, this is close to a worst-case scenario: the model can do little better than guessing, since any straight line through the data will misclassify roughly half of each ring.

4. Building the quantum classifier

The quantum side of the pipeline is built using PennyLane, a Python library for differentiable quantum programming that lets a quantum circuit be trained with the same gradient-based tools used in classical deep learning. Our circuit uses just two qubits, simulated on PennyLane's `default.qubit` device, and is built from two components that appear in almost every variational quantum machine learning model.

4.1 Two building blocks: the feature map and the variational layer

The two-qubit hybrid classifier pipeline Graph 1

Feature Map	Variational Layer	Measurement
Encodes each classical data point into a quantum state, using RX rotations on both qubits followed by a CNOT gate and RY rotations that mix information between them	A trainable circuit of RY, CNOT, and RZ gates whose parameters are optimised during training, playing the same role as weights in a classical neural network	The qubits are measured, and the resulting values are turned into a class prediction, closing the loop back to ordinary classical post-processing

Source: Author's synthesis, adapted for classroom presentation.

The **feature map** is the step that actually gets classical data into the quantum computer. Each of the two input features of a data point is loaded onto a qubit using an RX rotation, and a CNOT gate is then applied to entangle the two qubits – a distinctly quantum resource with no classical equivalent, allowing the two features to influence each other in the resulting quantum state rather than remaining independent. Further RY rotations mix this entangled information more thoroughly before the circuit moves on.

The **variational layer** is where learning happens. Its RY and RZ rotation angles are not fixed; they are parameters, initialised randomly and then adjusted through training, exactly as the weights of a classical neural network are adjusted. A second CNOT gate entangles the qubits once more after this trainable rotation, giving the circuit more expressive power than a variational layer without any entanglement would have. Together, the feature map and the variational layer form a small quantum neural network: data goes in through the feature map, gets transformed by trainable quantum operations, and a measurement at the end yields a prediction.

5. Comparing the two models

With both models trained on the identical concentric-circles dataset, the difference in accuracy is modest but instructive.

Accuracy on the concentric-circles classification task Graph 2

49%	55%
Classical logistic regression – close to random guessing on this non-linear data	Two-qubit hybrid quantum classifier – a modest improvement from the entangled feature map

Source: Author's classroom demonstration notebook, PennyLane default.qubit simulator, 2025.

A few honest caveats belong alongside this result. The gap between 49% and 55% is real but small, and this is a toy demonstration on a tiny, two-feature, two-qubit example run on a classical simulator of a quantum computer rather than real quantum hardware – it is meant to illustrate the mechanics of a hybrid pipeline clearly, not to claim a practical quantum advantage. What the result does show convincingly is the underlying reason such an advantage is even plausible: the entangling CNOT gates let the quantum feature map capture a kind of correlation between the two input features that a linear classical model, by construction, cannot see. Whether that translates into a meaningful real-world advantage on larger problems and real hardware remains an open, actively researched question.

6. Emerging directions in the field

Today's two-qubit example is deliberately minimal, but it sits inside a much larger and fast- moving research area. A few directions worth knowing about as you go further: **variational quantum classifiers** like the one built today are being scaled up with more expressive feature maps and deeper variational circuits; **quantum kernel methods** use a quantum circuit only to compute a similarity measure between data points, which is then handed to an otherwise entirely classical support vector machine; and **quantum-enhanced optimisation** explores whether quantum circuits can speed up the training process itself, not just the model's representation of data. All of this work currently runs up against the same practical ceiling: today's quantum hardware is small, noisy, and error-prone, which is precisely why the hybrid quantum-classical pattern you saw today, rather than a fully quantum pipeline, is the dominant approach across the field.

Closing note

The purpose of walking through this small example was not to convince you that quantum computers are about to replace classical machine learning – they are not, certainly not yet. It was to make the idea of a hybrid quantum-classical model concrete enough that you could build one yourselves: two qubits, one feature map, one trainable layer, and a measurement, wired into an otherwise ordinary Python training loop. That is genuinely how much of this research area works today, and it is a reasonable place for any of you to start experimenting.

For queries, project proposals, ideas, or guidance related to this lecture, please write to vikhrams@saveetha.ac.in.