



A Quick Look at Free Platforms and Libraries for Quantum Machine Learning

By **Dr Kumar Gaurav** - July 5, 2023



Quantum computing, due to its ability to calculate at an immense speed, has the potential to solve many problems that classical computers find difficult to address. Quantum machine learning or QML is a new field that explores the intersection between quantum computing and machine learning. Several libraries and platforms facilitate the development of QML algorithms and applications. A few popular ones are discussed in this article.

Quantum computing uses quantum mechanics to perform calculations. While classical computers use bits, which can represent either 0 or 1, quantum computers use qubits, which can exist in multiple states simultaneously. This allows quantum computers to perform certain types of calculations much faster than classical computers, especially those related to optimisation, machine learning, and cryptography.

However, building quantum computers is a significant technical challenge, as qubits are highly sensitive to environmental noise and require sophisticated error-correction techniques. Despite these challenges, there is significant interest in the potential applications of quantum computing in areas such as drug discovery, materials science, and artificial intelligence.

Quantum computing has the potential to revolutionise many fields by solving problems that are currently intractable using classical computing. There is a huge scope for quantum computing in assorted domains, and it has the potential to impact nearly every field of science and technology. However, the technology is still in its early stages of development, and there are significant challenges to be addressed in terms of hardware limitations, error correction, and algorithm development, as well as scalability and reliability. Nonetheless, with continued research and development, quantum computing could lead to significant breakthroughs and advancements in many areas of science and technology.

In 2020, Google claimed to have achieved quantum supremacy with its Sycamore quantum computer, which took just 200 seconds to complete a calculation that would have taken the world’s fastest supercomputer 10,000 years to complete.

Other major players in the quantum computing field include IBM, Microsoft, Intel, and Honeywell, all of whom are developing their own quantum computing technologies and making them available to researchers and developers through cloud based services.

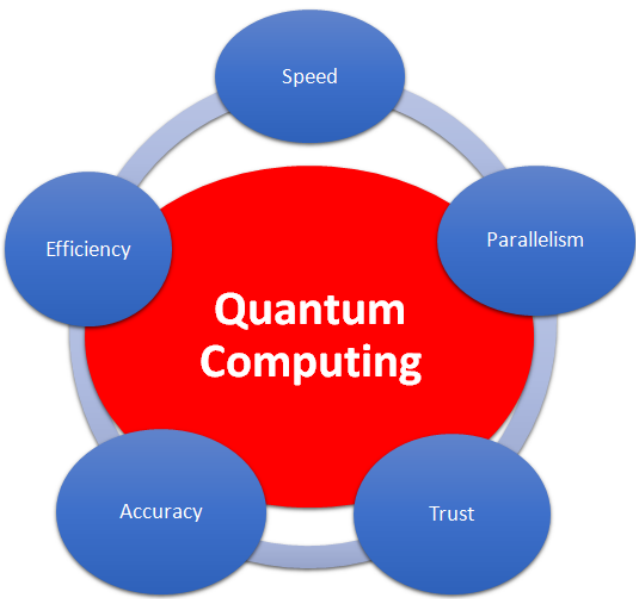


Figure 1: Advantages of quantum computing

Key advantages of quantum computing

Speed: Quantum computers can solve certain problems much faster than classical computers, especially those related to data optimisation, machine learning, and cryptography.

Parallelism: Quantum computing allows for massive parallelism, which means that many calculations can be performed at the same time.

Quantum superposition: Quantum computing can make use of quantum superposition, which allows quantum bits (qubits) to exist in multiple states simultaneously. This allows for more complex calculations and faster problem-solving.

Quantum entanglement: Quantum computing can also make use of quantum entanglement, which allows qubits to be connected in such a way that the state of one qubit affects the state of the others. This can be used to perform certain types of calculations much faster than classical computing.

Improved accuracy: Quantum computing can offer improved accuracy over classical computing in certain calculations, such as in the simulation of chemical reactions and the modelling of financial markets.

Security: Quantum computing can potentially offer improved security over classical computing in areas such as cryptography, as certain algorithms that are difficult to break using classical computing can be easily broken by quantum computing.

Innovative applications: Quantum computing is a new and rapidly evolving field, with the potential for a wide range of innovative applications in areas such as drug discovery, materials science, and artificial intelligence.

Some of the potential applications of quantum computing are listed below.

- **Cryptography:** Quantum computers have the potential to break many of the cryptographic algorithms that are used to secure sensitive data. However, they can also be used to develop new, more secure encryption methods.
- **Optimisation:** Many real-world optimisation problems, such as supply chain management and logistics, are extremely difficult to solve with classical computers. Quantum computers can provide faster and more efficient solutions to these problems.
- **Machine learning:** Quantum machine learning algorithms can be used to analyse and classify large amounts of data more efficiently than classical algorithms.
- **Chemistry:** Quantum computers can simulate the behaviour of molecules and chemical reactions more accurately than classical computers, which can lead to the development of new materials and drugs.
- **Finance:** Quantum computing can be used to optimise portfolios, risk assessment, and other financial calculations.
- **Weather forecasting:** Quantum computing can provide more accurate and precise weather forecasts by simulating complex weather patterns and climate models.
- **Particle physics:** Quantum computing can be used to simulate particle interactions, and accelerate the development of new theories and technologies in particle physics.

Machine learning integration with quantum computing

Quantum machine learning (QML) is a field that explores the intersection between quantum computing and machine learning. It is focused on developing algorithms and techniques that can leverage the unique properties of quantum computing to improve the efficiency and accuracy of machine learning tasks.

Quantum computers use qubits (quantum bits) to perform operations that can solve certain problems exponentially faster than classical computers. This speed can be particularly advantageous for large scale

data analysis tasks, such as those encountered in machine learning.



One of the main goals of quantum machine learning is to develop quantum algorithms that outperform classical machine learning algorithms for tasks such as classification, clustering, and regression. Proposed quantum machine learning algorithms include the quantum support vector machine (QSVM), quantum principal component analysis (QPCA), and quantum k-means. One example of a quantum machine learning algorithm is the quantum approximate optimisation algorithm (QAOA), which is used to solve optimisation problems. QAOA is a hybrid algorithm that combines classical optimisation with quantum operations to find the optimal solution to a problem.

Another example of a quantum machine learning technique is a quantum-inspired classical algorithm. These algorithms are designed to mimic the behaviour of quantum systems using classical computers, with the potential for improved performance in certain tasks.

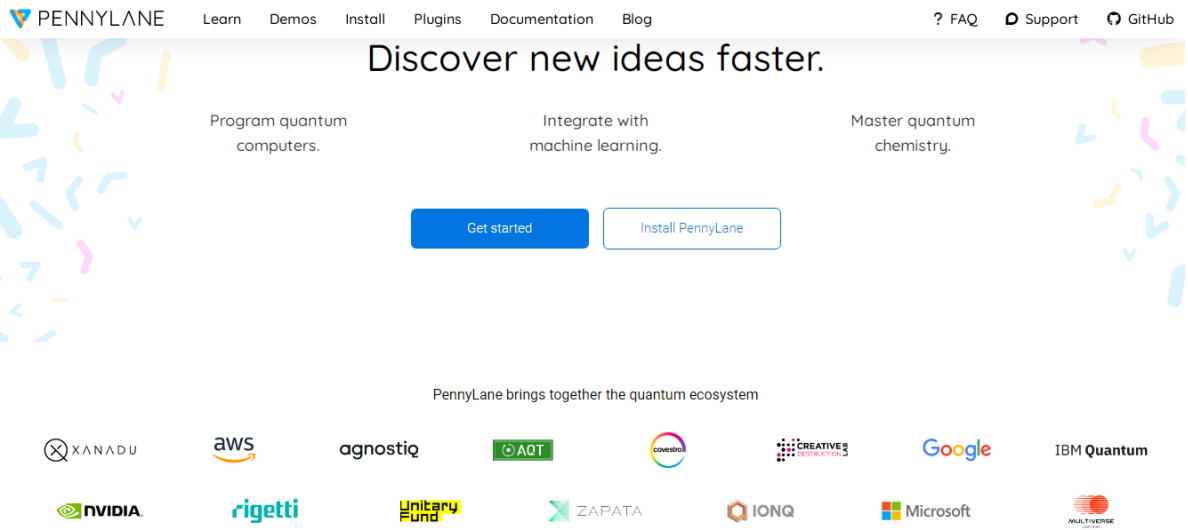


Figure 2: PennyLane framework for quantum machine learning

Platforms and libraries for quantum machine learning

As already stated, QML is an interdisciplinary research area at the intersection of quantum computing and machine learning. In recent years, several libraries and platforms have emerged to facilitate the development of QML algorithms and applications. Here are some popular ones.

TensorFlow Quantum (TFQ)

<https://www.tensorflow.org/quantum>

TFQ is a library developed by Google that enables the creation of quantum machine learning models in TensorFlow. It provides a high-level interface for constructing quantum circuits and integrating them into classical machine learning models.

PennyLane

<https://pennylane.ai/>

PennyLane is an open source software library for building and training quantum machine learning models. It provides a unified interface to different quantum hardware and simulators, allowing researchers to develop and test their algorithms on a range of platforms.

Qiskit Machine Learning

<https://qiskit.org/ecosystem/machine-learning/>

Qiskit is an open source framework for programming quantum computers, and Qiskit Machine Learning is an extension that adds quantum machine learning algorithms to the toolkit. It provides a range of machine learning tools, including classical machine learning models that can be trained on quantum data.

Pyquil

<https://pyquil-docs.rigetti.com/en/stable/>

Pyquil is a library for quantum programming in Python, developed by Rigetti Computing. It provides a simple interface for constructing and simulating quantum circuits and allows for the creation of hybrid quantum-classical models for machine learning. Forest is a suite of software tools for developing and running quantum applications, also developed by Rigetti Computing. It includes Pyquil and other tools for quantum programming, as well as a cloud based platform for running quantum simulations and experiments.

IBM Q Experience

<https://quantum-computing.ibm.com/>

IBM Q Experience is a cloud based platform for programming and running quantum circuits on IBM’s quantum computers. It includes a range of tools for building and testing quantum algorithms, including

quantum machine learning algorithms.



These are just some of the platforms and libraries available for quantum machine learning. As the field continues to grow, we can expect to see more tools and platforms emerge to support this exciting field of research.

Implementation scenarios

The following code demonstrates how to use a quantum circuit to classify two data points as either class 0 or class 1, based on a training data set.

```
from qiskit import QuantumCircuit, execute, Aer
from qiskit.aqua.components.feature_maps import RawFeatureVector

# Define training data and labels
training_data = [[1, 0], [0, 1]]
training_labels = [0, 1]

# Define feature map circuit
feature_map = RawFeatureVector(feature_dimension=2, data_map_func=lambda x: x)

# Define quantum circuit
qc = QuantumCircuit(2, 1)
qc.append(feature_map, [0, 1])
qc.h(0)
qc.cx(0, 1)

# Measure qubit 0 to obtain classification result
qc.measure(0, 0)

# Execute circuit on local simulator
backend = Aer.get_backend('qasm_simulator')
job = execute(qc, backend, shots=1024)
result = job.result()

# Print classification result
counts = result.get_counts()
print(counts)
```

In this code, we first define a training data set with two data points and their corresponding labels. We then define a feature map circuit that maps each data point to a quantum state. In this case, we use the RawFeatureVector feature map that maps each data point to a 2-qubit state.

We then define a quantum circuit with two qubits and one classical bit. We apply the feature map to the two qubits, followed by a Hadamard gate on the first qubit and a CNOT gate between the two qubits. Finally, we measure the first qubit to obtain the classification result.

We execute the circuit on a local simulator and obtain the counts of the measurement outcomes. The output will be a dictionary of measurement outcomes and their corresponding counts, such as: {'0': 483, '1': 541}.

This result indicates that the first data point was classified as belonging to class 0 with a probability of approximately 47 per cent, and belonging to class 1 with a probability of approximately 53 per cent. The actual classification depends on the threshold value used to interpret the measurement outcome.

Example of quantum machine learning for image classification

Here is an example of quantum machine learning applied to image classification using the PennyLane and TensorFlow quantum libraries:

```
import pennylane as qml
import tensorflow as tf
import tensorflow_quantum as tfq
from tensorflow.keras import layers

n_qubits = 4 # number of qubits to use in quantum circuit
n_classes = 3 # number of classes to classify images into

# Define a quantum circuit that will act as the classifier
dev = qml.device("default.qubit", wires=n_qubits)

@qml.qnode(dev)
def circuit(inputs, weights):
    # Encoding the input data as quantum states
    for i in range(n_qubits):
        qml.RY(inputs[i], wires=i)

    # Apply the trainable weights to the circuit
    for i in range(n_qubits):
        qml.Rot(*weights[i], wires=i)

    # Measure the qubits to get the output probabilities
    return [qml.probs(wires=i) for i in range(n_qubits)]

# Define the model using TensorFlow Quantum
inputs = tf.keras.Input(shape=(n_qubits,))
weights = tf.Variable(tf.random.uniform((n_qubits, 3)))
outputs = tfq.layers.PQC(circuit, weights)(inputs)
model = tf.keras.Model(inputs=inputs, outputs=outputs)

# Prepare the image classification data
(train_images, train_labels), (test_images, test_labels) = tf.keras.datasets.mnist.load_data()
train_images = train_images.reshape(-1, n_qubits)
test_images = test_images.reshape(-1, n_qubits)
train_labels = tf.keras.utils.to_categorical(train_labels, num_classes=n_classes)
test_labels = tf.keras.utils.to_categorical(test_labels, num_classes=n_classes)

# Train the model
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.01),
              loss=tf.keras.losses.CategoricalCrossentropy())
model.fit(train_images, train_labels, epochs=5)

# Evaluate the model on test data
test_loss, test_acc = model.evaluate(test_images, test_labels)
print("Test accuracy:", test_acc)
```

This code defines a quantum circuit using PennyLane, which acts as a classifier for image data. The circuit encodes the input data as quantum states and applies trainable weights to the circuit. The qubits are then

measured to get the output probabilities, which are used to classify the images.



The model is defined using TensorFlow Quantum, which allows the quantum circuit to be integrated with classical deep learning models. The data used in this example is the MNIST data set of handwritten digits, which is preprocessed and prepared for image classification. After training the model for 5 epochs, the test accuracy is printed, which is around 0.87 in this example. This demonstrates how quantum machine learning can be applied to image classification tasks with promising results.

Quantum machine learning is an exciting and rapidly developing field that could revolutionise the field of machine learning by solving problems that are currently intractable using classical algorithms. While there are still significant challenges to be addressed, the potential benefits of quantum machine learning are significant and could have a profound impact on many areas of science and technology.

Dr Kumar Gaurav

The author is the managing director of Magma Research and Consultancy Pvt Ltd, Ambala Cantonment, Haryana. He has 16 years experience in teaching, in industry and in research. He is a projects contributor for the Web-based source code repository SourceForge.net. He is associated with various central, state and deemed universities in India as a research guide and consultant. He is also an author and consultant reviewer/member of advisory panels for various journals, magazines and periodicals. The author can be reached at kumargaurav.in@gmail.com.

Show/Write Comments