Title: Quantum-Classical Hybrid Model for MNIST Classification Using TorchQuantum

OBJECTIVE:

- To classify MNIST handwritten digits.
- To train a quantum neural network to tell whether a handwritten digit is a 3 or a 6.

RELATED THEORY:

Quantum computing: Uses qubits instead of classical bits, which can exist in a superposition of 0 and 1 and be manipulated by quantum gates.

Quantum Neural Networks (QNNs): Combine classical preprocessing with quantum layers for pattern learning. Data is encoded into qubits, processed by quantum gates, and measured back into classical outputs. QNNs are theoretically capable of learning patterns faster or in higher-dimensional spaces than classical networks for certain problems.

MNIST Dataset

- MNIST stands for Modified National Institute of Standards and Technology.
- Standard dataset of handwritten digits (70,000 images of digits 0–9, each 28×28 pixels).
- It is one of the most widely used datasets in machine learning and computer vision for handwritten digit recognition.
- It is easy to experiment with quantum neural networks because we can reduce the input dimensions for qubits.

STEPS THE CODE PERFORMS

A. Classical preprocessing

- Uses **PyTorch** for tensor operations and **TorchQuantum** for quantum circuits.
- Input: Images of handwritten digits (from MNIST).
- Average pooling reduces the 28×28 image to a smaller 4×4 (or similar) size.
- Downsample images to reduce dimensions for quantum encoding.
- Random seeds ensure reproducibility.

B. Quantum Encoding:

- Each classical input value is encoded into qubits using **tq.GeneralEncoder**.
- This converts classical features (pixels) into quantum states that a quantum circuit can process.

C. Quantum Layer (QLayer)

- Quantum gates perform transformations on qubits.

- Consists of trainable quantum gates (RX, RY, RZ, CRX) and random layers to extract features and learn patterns.
- Learn a mapping from input qubit states to output probabilities of each class.
- **forward** method applies gates in a structured order and allows the network to learn complex patterns.

D. Measurement:

- After quantum operations, each qubit is measured in the Pauli-Z basis: **tq.MeasureAll(PauliZ)**
- Get classical numbers representing the model's prediction.

E. Training:

- Compare predictions with actual labels and adjust the quantum gates to improve accuracy.
- Uses Adam optimizer with learning rate 0.005 and weight decay 1e-4.
- Loss function: negative log likelihood **(F.nll_loss).**
- After each epoch, model parameters are updated, and training loss is printed.

F. Validation / Testing:

- Evaluate accuracy and loss after each epoch**.**
- Results are stored to plot accuracy vs epochs and loss vs epochs.
- Accuracy = correct predictions / total samples.
- Loss = NLL of predictions vs true labels

Main Function: Loads data, initializes model on GPU, sets optimizer and scheduler, runs training for 100 epochs, evaluates on test set, and optionally simulates with Qiskit.

TRAINING SETUP

- **Dataset:** MNIST digits 3 and 6 (subset for testing)
- **Batch Size:** 256
- **Device:** GPU (T4 on Colab)
- **Optimizer:** Adam (lr=5e-3, weight decay=1e-4)
- **Epochs:** 100

SIMULATION RESULTS

Epoch	Train Loss	Val Accuracy	Val Loss
1	0.5770	0.8166	0.5205
...	...	...	...
100	0.4035	0.8830	0.4127

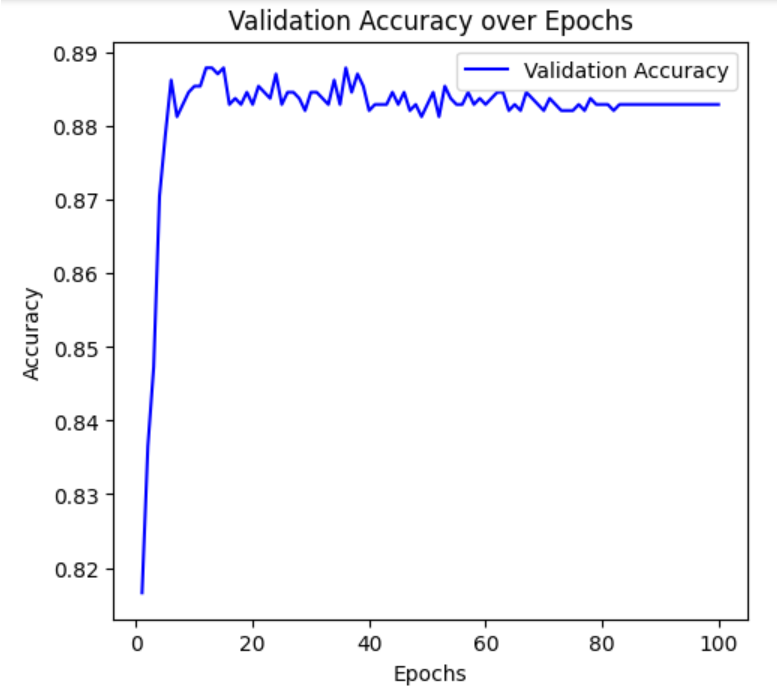

Figure 1: Validation accuracy over 100 epochs

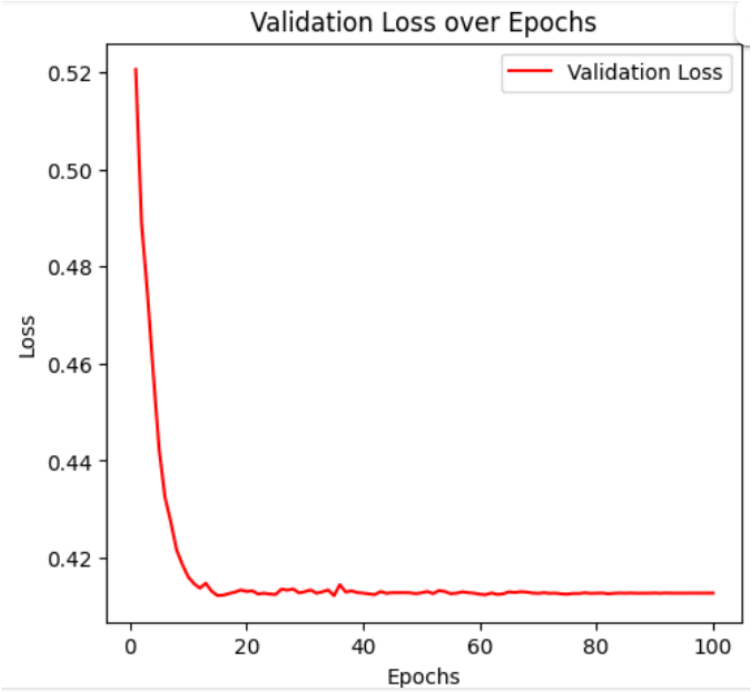

Figure 2: Train and validation loss over 100 epochs

Test Set:

- Accuracy: 0.867
- Loss: 0.459

OBSERVATIONS

- Validation accuracy gradually improved from 0.817 → 0.883 over 100 epochs.
- Validation loss decreased from 0.521 → 0.413, indicating stable convergence.
- Test set accuracy reached 0.867, demonstrating good generalization.

CONCLUSION

The quantum-classical hybrid model achieved ~87% accuracy after 100 epochs in classifying MNIST digits *3* and *6*. This shows that quantum-inspired learning can perform effectively on simple image recognition tasks and highlights its potential for future applications in quantum machine learning.