Journal of Scientific & Industrial Research
Vol. 84, January 2025, pp. 60-71
DOI: 10.56042/jsir.v84i1.14112

Hybrid Quantum Graph Neural Network for Brain Tumor MR Image Classification

S P Rajamohana[1]*, Vani Yelamali[2]*, Pallavi Soni[3], Manjunath Vanahalli[2] & Prabu Prasad[2]

[1]Department of Computer science, School of Engineering & Technology, Pondicherry University, Karaikal 609 605, Puducherry, India

[2]Department of Computer Science and Engineering, Indian Institute of Information Technology Dharwad 580 009, Karnataka, India

[3]Quantum Computing Researcher, Kwantum G Research Labs Pvt Ltd, Bangalore 560 064, Karnataka, India

Received 01 October 2024; revised 28 Nov 2024; accepted 01 December 2024

Brain tumor rank as the tenth leading cause of mortality among both adults and children. Early detection and treatment significantly enhance survival rates. Recent advancements in deep learning have demonstrated promise in identifying and classifying brain tumors using Magnetic Resonance Imaging (MRI) scans. This paper presents a novel approach that integrates classical and hybrid quantum-inspired graph neural networks for tumor classification. Classical Graph Convolutional Neural Networks (GCNN) analyse complex relationships within medical imaging data, while hybrid Quantum Graph Neural Networks (QGNN) improves performance by leveraging principles of quantum computing. Previous studies highlight the challenges posed by the diverse nature of brain imaging data. This study compares various classification approaches, emphasizing architectures, training techniques, and performance metrics. The objective is to train and evaluate models for identifying brain tumors from MRI scans. The hybrid QGNN demonstrates accuracy and loss metrics comparable to advanced classical GCNN, with accuracy improving from 0.41(41%) to 0.64(64%) during training and from 0.31(31%) to 0.52(52%) in validation datasets, thereby showcasing its effectiveness in distinguishing between normal and tumor images.

Keywords: Brain tumor MRI classification, Graph convolutional neural network, Quantum graph neural network

Introduction

The abnormal proliferation or clustering of cells within or adjacent to the brain is referred to as a brain tumor, also known as a Central Nervous System (CNS) tumor. Tumors are categorized into two types: benign and malignant. Benign tumors grow slowly and do not invade surrounding tissues, whereas malignant tumors are characterized by their aggressive spread to other locations. According to the World Health Organization, brain tumors rank as the tenth leading cause of death among both adults and children. In 2021, the American Cancer Society projected that 78,980 individuals would be diagnosed with brain tumors, which included 24,530 cases of malignant tumors and 55,150 cases of benign tumors. Among these, 13,840 cases occurred in males and 10,690 in females. Consequently, the diagnosis, prognosis, and treatment of brain tumors heavily rely on the automated segmentation and classification of medical images. The likelihood of a patient surviving a brain tumor increases when treatment is initiated promptly following an early diagnosis.[1]

*Authors for Correspondence
E-mail: vani.23phdcs02@iiitdwd.ac.in,
monamohanasp@pondiuni.ac.in

Díaz–Pernas *et al.*[2] developed a fully automated model for brain tumor segmentation and classification using a Deep Convolutional Neural Network with a multiscale approach. Unlike previous research, there model processes input images at three spatial scales through distinct pathways, inspired by the Human Visual System. It analyzes MRI images of meningioma, glioma, and pituitary tumors across sagittal, coronal, and axial views without pre-processing to remove skull or vertebral components. They evaluated the method on a publicly available dataset of 3,064 slices from 233 patients, achieving a tumor classification accuracy of 0.973, surpassing other methods using the same dataset.

Konar *et al.*[3] developed the Quantum Fully Self-Supervised Neural Network (QFS-Net), a self-supervised shallow learning model that employs a three-level qutrit-inspired quantum information system for the automated segmentation of brain MRI scans. The QFS-Net is designed with a layered qutrit architecture linked by parametric Hadamard gates, allowing for efficient self-organized counter propagation of quantum states without supervision. When tested on the Cancer Imaging Archive (TCIA) dataset, the QFS-Net surpassed leading supervised

models, such as U-Net and URes-Net, as well as the self-supervised QIS-Net in tumor detection, achieving high Dice similarity coefficients and accuracy with minimal human input. Furthermore, the model showed resilience when evaluated on natural grayscale images from the Berkeley Segmentation Dataset.

Amin *et al.*[4] proposed novel model for brain tumor detection that utilizes transfer learning and a quantum variational classifier (QVR) to differentiate between meningioma, no tumor, and pituitary tumor. Deep features were extracted from the InceptionV3 model and subsequently inserted into the QVR. The performance of the classification method was evaluated on using a dataset and two publicly available datasets. The model achieved an accuracy of 99.44% on no tumor, 99.44% on meningioma, on 98.03% pituitary. In the 2020-BRATS Challenge, the proposed method achieved 90.91% accuracy for HGG and LGG slices. A modified Seg-Network classified the total infected tumor region, achieving global accuracies of 0.982 on the Kaggle dataset, 0.999 on private images, and 0.997 in the challenge.

Vidyarthi *et al.*[5] conducted an analysis of real–time brain tumor images from 110 patients diagnosed with high–grade malignant brain tumors, including Central Neurocytoma (CNC), Glioblastoma Multiforme (GBM), Gliomas (GLI), Intraventricular Malignant Mass (IVMM), and Metastasis (MTS). This study employs a feature set from six domains to capture latent information in a designated region. Features are extracted using the Cumulative Variance Method (CVM) and applied to train models with K-Nearest Neighbour (KNN), multi-class Support Vector Machine (mSVM), and Neural Network (NN) methodologies for multi-class classification accuracy. The mean accuracies are 88.43% for KNN, 92.5% for mSVM, and 95.86% for NN. Compared to existing algorithms like Independent Component Analysis (ICA) and Genetic Algorithm (GA), the proposed method improves accuracy by approximately 2% for KNN, 3% for SVM, and 4% for NN. The NN classifier demonstrates superior performance, achieving 95.86% accuracy with diverse features.

Zain Eldin *et al.*[6] implemented the Brain Tumor Classification Model based on CNN (BCM-CNN). This study employs an adaptive dynamic sine-cosine fitness grey wolf optimizer (ADSCFGWO) for hyperparameter optimization. Using the Inception-ResnetV2 architecture, it enhances brain tumor diagnosis with binary outputs (0: Normal, 1: Tumor). The model optimizes two types of hyperparameters: those defining the network structure and those for training. The ADSCFGWO algorithm effectively combines the strengths of the sine-cosine and grey wolf algorithms. BCM-CNN achieved 99.98% accuracy on the BRaTS 2021 Task 1 dataset, demonstrating significant performance improvement through hyperparameter optimization.

Mahmud *et al.*[7] developed a convolutional neural network (CNN) for detecting brain tumors in magnetic resonance (MR) images. The study compared this model with established architectures like ResNet-50, VGG16, and Inception V3, using metrics such as accuracy, recall, loss, and area under the curve (AUC). The CNN achieved an accuracy of 93.3%, an AUC of 98.43%, a recall of 91.19%, and a loss of 0.25, demonstrating its reliability for early brain tumor detection compared to other models.

Choudhuri *et al.*[8] created a Quantum Classical Convolutional Neural Network (QCCNN) for binary classification of brain MR images, focusing on tumor identification. By encoding data into quantum states, the model allows for rapid information extraction, achieving accuracy rates of 97.5% to 98.72% across datasets like Brats 2013 and Harvard Medical School. Future advancements in quantum computing are expected to enhance this approach's performance. The model has been evaluated on datasets such as Brats 2013, Harvard Medical School, and a private dataset, using standard metrics to assess robustness. The QCCNN model shows accuracy rates of 97.5% to 98.72%, confirming its effectiveness in brain tumor detection and classification.

Darwish *et al.*[9] introduced a clustering methodology using the Quantum Inspired Dragonfly Algorithm (QDA) to enhance initial contour points and improve segmentation. This model balances exploitation and exploration, overcoming the limitations of traditional Dragonfly Algorithm (DA) methods, such as slow convergence and local optima. It outperformed leading techniques for brain tumor segmentation with 3D MRI images from the BraTS 2019 dataset.

Sharma *et al.*[10] created a deep neural network model for image and video analysis that integrates fully connected layers within a graph structure. This architecture effectively extracts discriminative features. Evaluated on the ImageNet dataset, which includes over 14 million images across 1,000 categories, the proposed Graph Neural Network (GNN) achieved an accuracy of 96.63%, outperforming current state-of-the-art methods.

Medical diagnosis increasingly relies on Machine Learning (ML) and Deep Learning (DL), as numerous algorithms have demonstrated strong performance and minimal error in the diagnosis and classification of brain tumors. Over the past decade, it has been observed that Neural Networks excel in processing structured data, such as text and images. These Neural Networks are utilized to implement Graph Neural Networks (GNNs). In a GNN, each node may possess a set of features that describe it, while the edges can be either undirected (two–way) or directed (one–way). Graph Neural Networks are specifically designed to operate on graph–structured data, where nodes represent entities and edges represent relationships. Classical GNNs have achieved significant success across various domains, including social network analysis, molecular biology, and recommendation systems. Quantum Graph Neural Networks (GNNs) represent an emerging field that aims to leverage quantum properties such as superposition and entanglement to perform computations potentially superior to those of classical counterparts.[11] The integration of the Quantum Graph Convolutional Layer into the neural network architecture involves the combination of quantum layers with classical neural network layers.[12]

Despite advancements in brain tumor classification using deep learning and quantum-inspired techniques, significant gaps remain.[2,7] Most research relies on classical neural networks or limited quantum approaches, often overlooking hybrid models that combine both paradigms. There is also a lack of integration between quantum principles and graph neural networks (GNNs), limiting the modeling of complex relationships in medical imaging data. Although accuracies are high, methods often lack robustness across multiple datasets, restricting clinical applicability. Many models are trained on limited databases, resulting in poor generalizability. Additionally, the focus on optimizing hyperparameters or feature extraction often overshadows the need for innovative architectures that capture the complexity of MRI images. Most existing models do not fully utilize graph-based representations for spatial and relational data. The Classical GCNN models are lacking in reducing overfitting problem in the models, as shown in the paper the prediction accuracy and loss throughout the process has improved in Hybrid QGNN. Quantum computing utilizes a concept called quantum parallelism, where the data can process simultaneously at once utilizing the concept of superposition. These gaps underscore the need for a hybrid quantum-inspired GNN approach to improve computational efficiency and diagnostic accuracy in brain tumor classification.

Materials and Methods

Dataset Description

The dataset utilized for this research is the Brain MR Imaging dataset, which is available on Kaggle.[13] This dataset is specifically designed for the training and evaluation of models aimed at identifying brain tumors from Magnetic Resonance (MR) images. It comprises a total of 253 brain MR images, classified into two categories, as shown in Fig. 1: patients with tumors, designated as Class 1, and patients without tumors, designated as Class 0. Each image is appropriately labelled to indicate the presence or absence of a brain tumor in the respective patient. The image files are formatted in PNG, and although their sizes vary, they generally maintain a high resolution, which is essential for accurate tumor detection.

- Total images: 253
- Categories:
  - With tumor (1): 155 images
  - Without tumor (0): 98 images

Data Pre–processing

The initial phase of data pre-processing involves loading brain tumor images from the dataset. The dataset is organized into two primary directories: one containing images of brain tumors and the other containing normal images (without brain tumor). Each image is retrieved from its respective directory, ensuring that all available data is utilized. This loading process includes navigating the directory

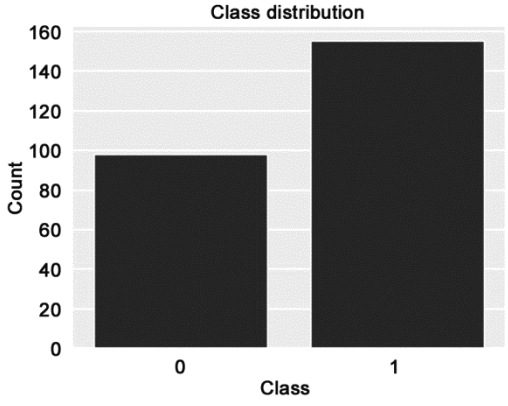

Fig. 1 — Brain tumor image classes

structure to locate and access each image file. The images are then stored in arrays, with appropriate labels assigned to indicate the presence or absence of a brain tumor. This systematic approach ensures that the dataset is efficiently organized and prepared for the subsequent pre- processing step.

Once the images are loaded, they undergo normalization and resizing, as illustrated in Fig. 2. In Fig. 2 (a) displays the original images containing brain tumors alongside their resized in (b) and normalized in (c). In Fig. 2 (d), (e), (f) the original images without brain tumor, along with resized and normalized version are given. Normalization is a critical step that involves converting the pixel values of each image to a common scale, typically ranging from 0 to 1. This process helps reduce computational complexity and ensures that the model can process the images effectively. The images are resized to a specific dimension, in this case, 64 × 64 pixels, to standardize their proportions. This uniformity in size is essential for inputting the images into the machine learning pipeline, as it guarantees consistency and comparability across all data. Additionally, converting the images to grayscale further reduces dimensionality and directs the model's focus toward essential features relevant for classification, thereby enhancing the efficiency of the processing pipeline.

Principle Component Analysis (PCA)

Principal Component Analysis (PCA) is utilized to reduce the dimensionality of the dataset from 54 to 5 dimensions.[14] This technique minimizes the number of features to align with the number of qubits. The dataset is partitioned in an 80:20 ratio for training and testing purposes. Each image has a resolution of 64 × 64 pixels. To visualize the dataset of brain tumor images as graphs, the 64 × 64-pixel images, which comprise 4,096 features, are flattened using PCA into a vector space. This vector space is represented as nodes, while the connections (edges) between these nodes are depicted as a tensor using PyTorch Geometric in the model.

Classical Graph Convolutional Neural Network

In classical Graph Convolutional Neural Networks (GCNN), images are initially converted into graph data and subsequently processed according to the architecture depicted in Fig. 3. This conversion enhances connectivity and facilitates the extraction of attributes from the data. Initially, the data is in the form of grayscale images; during pre-processing, it is transformed into graphical data, after which the process follows the traditional approach of training, testing, and evaluation.

Model Architecture

Graph Construction

To compute the graphical data, the model requires two grids: the first represents the nodes and their associated features, while the second represents the edges connecting the nodes, along with a corresponding edge index for the final classification.

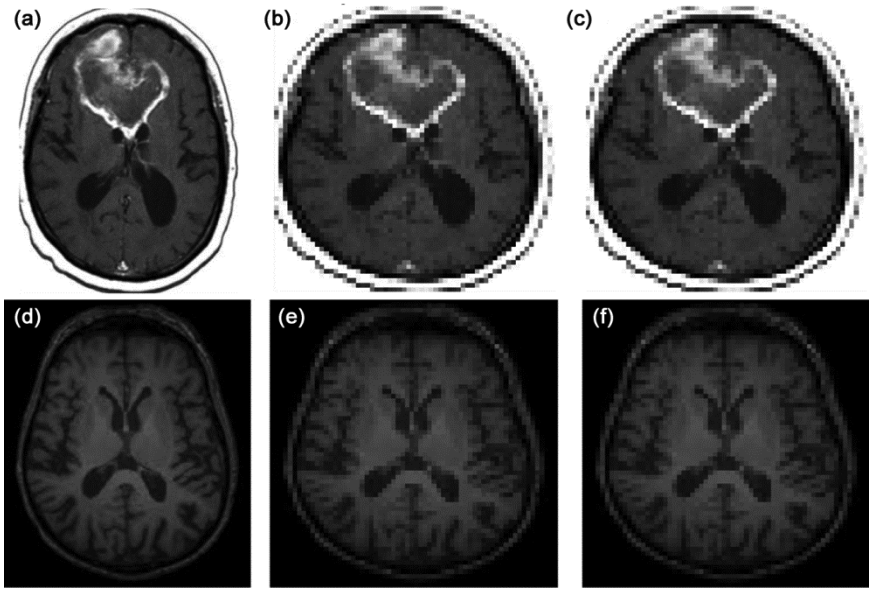

Fig. 2— Image of brain with and without tumor: (a) with tumor-original, (b) with tumor-resized, (c) with tumor-normalized, (d) without tumor-original, (e) without tumor-resized, (f) without tumor-normalized

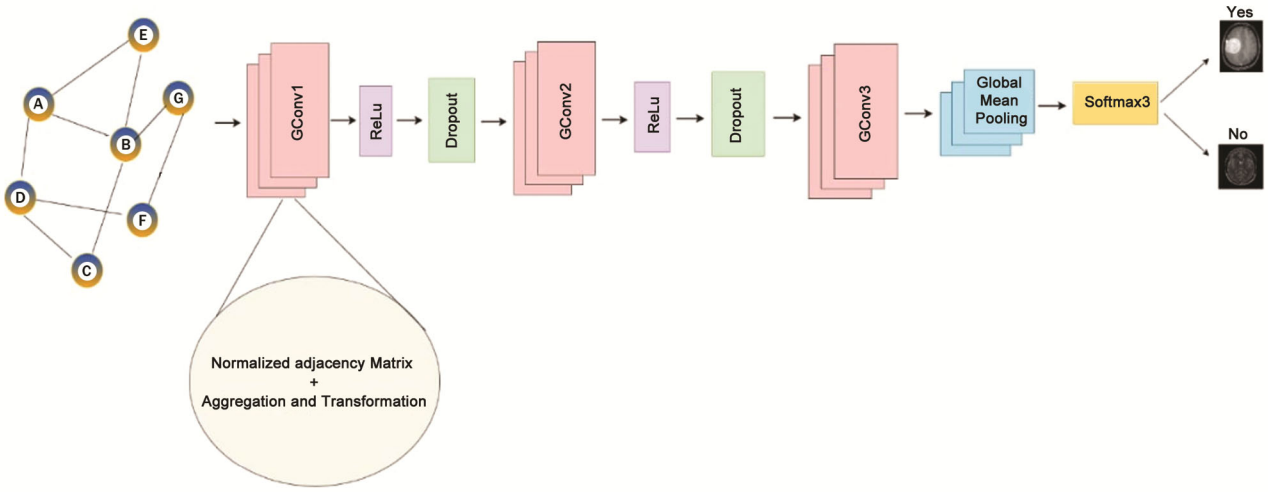

Fig. 3 — Architecture of classical graph convolutional neural network

- Input Layer: In this layer, the PCA-reduced graphical data is fed into the model.
- GCNConv Layers: Traditionally, convolutional layers are employed to update feature information and the connections between neighbouring nodes as pixels. In contrast, Graph Convolutional Neural Networks (GCNs) update features in their respective vector or tensor spaces based on their connections in a graph. This process also includes the addition of self-loops, normalization, aggregation, and linear transformation using a weight matrix.
- Dropout: This layer is employed to prevent overfitting during training. The concept involves randomly setting a subset of neurons to zero during each iteration, which encourages the model to learn more effectively and prevents it from becoming stagnant throughout the training process.
- Global Mean Pooling: It aggregates the node features to create a single graph representation.
- Activation Function: SoftMax activation is used to normalize the node features.

Training and Optimization

Loss Function: The binary cross-entropy loss function is used to measure the dissimilarity between the predicted probability distribution and the actual binary labels of a dataset.

Evaluation: To evaluate the model's efficiency, accuracy and loss are utilized to assess underfitting and overfitting. Additionally, the classification report and confusion matrix are employed to analyse predictive performance.

Proposed Hybrid Quantum Graph Neural Network:

In the proposed PCA-transformed data, a quantum layer is integrated into the architecture. A hidden layer, referred to as a parameterized circuit, is incorporated into the neural network to merge quantum and classical neural networks. This circuit utilizes a classical input vector to configure the rotation angles of the quantum gates. The output from the preceding neural network layer serves as input to this parameterized circuit. The measurement statistics from the parameterized circuit are collected and used as inputs for the subsequent layer of the neural network. This process is repeated until the final output layer is reached. The QGNN layer processes these quantum states using a combination of quantum gates designed to capture complex data relationships. Following the quantum layer, the architecture includes classical neural network layers. These classical layers typically consist of dense (fully connected) layers that further process the outputs from the quantum layer. The dense layers employ activation functions, such as ReLU, to introduce nonlinearity and enhance the network's ability to learn intricate patterns. The final layer of the architecture is configured for binary classification, utilizing a sigmoid activation function to produce a probability that indicates the presence or absence of a brain tumor, as illustrated in Fig. 4.

Quantum Device Initialization

Quantum device initialization is the process of configuring a quantum computing environment to perform computations. In this study, we utilized a quantum simulator known as the 'default. qubit'

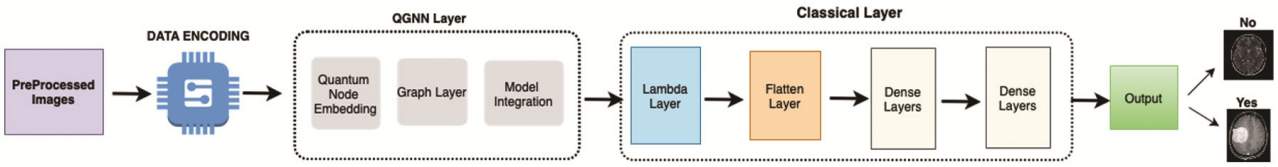

Fig. 4 — Architecture of hybrid quantum graph neural network

device, provided by the PennyLane library. This process involves specifying the number of qubits required for the computations. In this instance, we initialize a device with a number of qubits that corresponds to the principal components obtained from Principal Component Analysis (PCA). This initialization step is crucial as it defines the quantum resources and sets the stage for implementing quantum circuits that will process the image data.

Data Encoding

Angle encoding enables the representation of classical data in the quantum domain, paving the way for quantum machine learning algorithms. By encoding classical information into quantum states, we can leverage the parallelism and computational power of quantum computing to process and manipulate data more efficiently.[15] To encode the classical information to quantum, the rotation gate Ry(θ) rotates the qubit state around the y–axis of the Bloch sphere by an angle θ,[16] which is determined during the training process. The parameter θ enables the quantum circuit to learn and adapt based on the data, effectively parameterizing the quantum neural network. Ry(θ) rotations are used according to the number of qubits shown in Fig. 5. In the data encoding, the classical images of 64×64 size in grayscale are transformed into the quantum state using Ry(θ) rotations, which will store the information in the Bloch sphere as rotations along the y-axis. After the rotations, it returns the expected value using the Pauli–Z operator. This encoding translates classical data into a quantum representation utilizing parameterized quantum circuits, which constitute the foundation of the QGNN. Each data point is represented as a quantum state, where the amplitude and phase of the quantum bits (qubits) correspond to the features derived from PCA.

Graph Construction

The quantum encoding facilitates the construction of a quantum graph where each node corresponds to a qubit and its state represents the information from the image. This graph– based representation is essential for

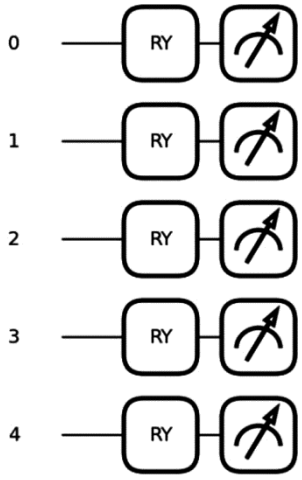

Fig. 5 — Data Encoding

leveraging the quantum computational capabilities of the QGNN, allowing it to process and learn from complex data patterns more efficiently than classical methods. By structuring the data as a graph, the QGNN can exploit quantum entanglement and superposition, potentially leading to superior performance in the classification of brain tumor images.

Quantum Circuit Design

The layout of the quantum circuit is a crucial element in quantum image processing, as shown in shown in Fig. 6. The quantum circuit is engineered to encode conventional data into quantum states and execute quantum operations that facilitate the Quantum Graph Neural Network (QGNN) in learning from the data.

Quantum Operations

Quantum operations are the precise actions executed by the quantum circuit on the qubits to manipulate the data. These operations encompass a series of quantum gates that alter the states of qubits to derive significant patterns from the data. In the QGNN, quantum operations entail the application of parameterized rotations (utilizing gates such as RY and Rot) and entangling gates (like CNOT) to construct intricate quantum states that embody the fundamental structure of the data. These procedures

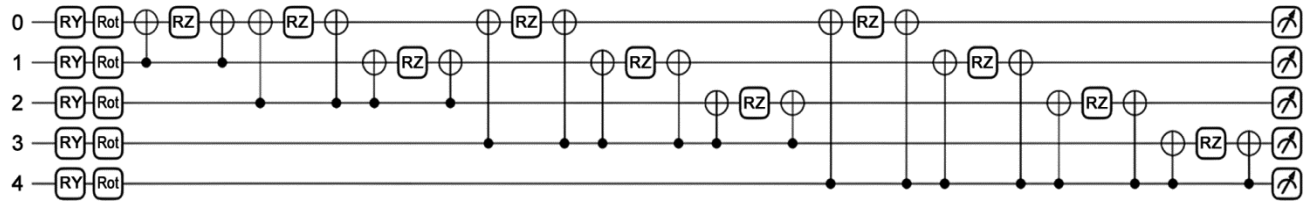

Fig. 6 — Quantum circuit

allow the QGNN to utilize quantum phenomena such as superposition and entanglement, hence improving its capacity to learn and classify brain tumor images. The meticulous coordination of these quantum actions is essential for attaining precise and effective quantum processing of images.

Quantum Node Embeddings

In the QGNN layer, the function includes a Rot gate, which applies graph features to each qubit in all three directions. These Rot gates give rotations to each individual qubit according to the weights applied; these weights are also considered as input features that represent the nodes and update the individual feature representation in the Bloch sphere. After that, the combination of CNOT and RY gates is used for feature extractions, and explicitly the combination is used for building a relationship between nodes to compute the output state. In Quantum Graph Neural Networks (QGNN), quantum gates and circuits are used for transforming the classical images into quantum states, and then this quantum information is used to transform and process the information in the form of nodes according to the number of qubits used.

Algorithm 1 Classical to Quantum Encoding

Input: Features
Output: Encoded data
Function circuit(x):
Allocate n–qubits in the quantum state
for i=0 to n–qubits–1:
Apply Ry(f(x[i]), qubits[i]) // f(x[i]) *defines the rotation angle mapping*
End for
Measure each qubit in the z–basis and store expectation values
measurement results = []
for i = 0 to n qubits − 1 do
measurement results. append
end for
return measurement results
end function

Mathematical Representation

Feature Encoding: Each feature vector x is encoded into a quantum state using rotation gates RY as shown in Eq. (1) If x_i is a feature of the vector x, it is encoded as follows:

$$\mathrm{RY}(\theta) = e^{i\frac{\theta}{2}}$$

$$Y=\begin{pmatrix} cos\theta/2 & -sin\theta/2 \\ sin\theta/2 & cos\theta/2 \end{pmatrix} \ldots (1)$$

where , Y is the Pauli–Y matrix

$$\mathrm{Y} = \begin{bmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{bmatrix}$$

When applied these rotations to the qubits is as shown below

$$|\psi\rangle = \mathrm{RY}(\theta_1) \otimes \mathrm{RY}(\theta_2) \otimes \mathrm{RY}(\theta_3) \otimes \mathrm{RY}(\theta_4) \otimes \mathrm{RY}(\theta_5)\,|0\rangle^{\otimes 5} \quad \ldots (2)$$

Measurement: The expectation values of the Pauli–Z operator are measured as shown below

$$\langle Z\rangle_i = \langle\psi|Z_i|\psi\rangle \quad \ldots (3)$$

where, Z is the Pauli–Z matrix:

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Hybrid Quantum Graph Neural Network Layer

The Hybrid Quantum Graph Neural Network Layer enhances the traditional graph convolutional neural network function within a quantum framework. It involves applying parameterized quantum gates to extract features from the graph representation.

Mathematical Representation:

Rotation Gates: Each qubit q_i is subjected to a rotation operation parameterized by weights w as shown in Eq. (4):

$$\mathrm{Rot}(\theta, \varphi, \lambda) = \mathrm{R_X}(\theta)\mathrm{RY}(\varphi)\,\mathrm{R_Z}(\lambda) \quad \ldots (4)$$

where, $\mathrm{R_Z}$ and RY are rotation matrices.

Entanglement: CNOT gates are applied to introduce entanglement between qubits[17] as shown in Eq. (5):

$$\mathrm{CNOT}_{ij} = |0\rangle\langle 0| \otimes \mathrm{I} + |1\rangle\langle 1| \otimes \mathrm{X} \quad \ldots (5)$$

where, X is the Pauli–X matrix.

Expectation Values: Following the application of rotations and entanglement, the expectation values of the Pauli–Z operator are recorded as indicated in Eq. (6):

$$\langle Z\rangle_i = \langle\psi|Z_i|\psi\rangle \quad \ldots (6)$$

here, Z is the Pauli–Z matrix:

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Model Integration:

Integrating the Quantum Graph Neural Network Layer into the overall neural network model involves combining quantum layers with classical neural network layers. This hybrid approach leverages the strengths of both paradigms to enhance model performance.

Mathematical Representation:

Input Layer: Accepts the encoded quantum states $x \in R^n$ where, n is the number of qubits.

Quantum Graph Neural Network Layer:

Applies the operations described above to produce quantum node embeddings $h^{(l)}$ as shown in Eq. (7):

$$h^{(l)} = \text{QGNNLayer}(h^{(l-1)}, w) \quad \ldots (7)$$

Reshape Layer: Converts the output to a suitable shape for further processing.

Flatten Layer: Flattens the multi–dimensional tensor to a one–dimensional tensor as shown:

$$h_{flat} = \text{Flatten}(h) \quad \ldots (8)$$

Dense Layers: Perform classical neural network operations as shown in Eq. (9):

$$h_{dense1} = \sigma\,(W_{dense1} h_{flat} + b_{dense1}) \quad \ldots (9)$$

where, σ is the ReLU activation function.

Output Layer: Employs a sigmoid activation function for binary classification, as demonstrated in Eq. (10):

$$\hat{y} = \sigma\,(W_{outputh dense2} + b_{output}) \quad \ldots (10)$$

Training Procedure

The training process for the QGNN encompasses several essential elements to enhance the model's efficacy. The PCA–transformed data is first encoded into quantum states via the quantum layer. The model is trained utilizing a supervised learning methodology, wherein the input data (quantum states) and associated labels (tumor or no tumor) are input into the quantum neural network. Subsequently, the classical component engages in the training process, employing an optimizer, such as Stochastic Gradient Descent (SGD) with momentum, to minimize the binary cross–entropy loss function. This loss function quantifies the disparity between projected probability and actual labels, directing the optimizer to modify the model's parameters to enhance accuracy. The training consists of several iterations across 100 epochs, during which the model parameters are adjusted in each epoch according to the gradients derived from the loss function. Batch processing is employed to manage subsets of training data simultaneously, hence improving the efficiency and stability of the training procedure.

Algorithm 2 QGNN

```
Input: Inputs, n qubits, dev
Output: Quantum graph neural network results
Function QGNN (inputs, n qubits, dev):
    Initialize QGNN with n qubits and device
    Allocate weight kernel of shape (nqubits, 3)
    Function qgnn circuit func(inputs, weights):
        Define quantum node (qnode) with device dev
        for i = 0 to n qubits − 1 do
            Apply RY (inputs [0, i], wires = i)
        end for
        Apply qgnn layer(weights)
        Initialize results = []
        for i = 0 to n qubits − 1 do
            Append qml.expval(qml.P auliZ(i)) to results
        end for
        return results
    end function
    Function qgnn layer(weights):
        for i = 0 to n qubits − 1 do
            Apply qml.Rot(weights[i, 0], weights[i, 1], weights
            [i, 2], wires = i)
        end for
        for i = 0 to n qubits − 1 do
            for j = 0 to i − 1 do
                Apply qml.CNOT(wires = [i,j])
                Apply qml.RZ(weights[j, 0], wires = j)
                Apply qml.CNOT(wires = [i,j])
            end for
        end for
    end function
    Define qgnn results
tf.py function (func=qgnn circuit func,
inp= [inputs, kernel], Tout=tf.float32)
        return qgnn results
end function
```

Model Evaluation

Model evaluation is an essential process for assessing the performance and generalization ability of the trained QGNN. The assessment entails evaluating the model using a validation set comprising data that was excluded from the training phase. Essential performance indicators, including accuracy, loss, precision, recall, and the F1–score, are calculated to assess the model's efficacy. The confusion matrix visualizes the classification performance of a model, emphasizing true positives, false positives, true negatives, and false negatives. This comprehensive assessment aids in detecting potential overfitting or underfitting problems and offers insights on areas for further model enhancement.

Confusion Matrix

The confusion matrix is a 2 × 2 matrix for binary classification which is used to visualize how many samples are correctly predicted out of the total samples shown in Table 1.[18]

Performance Metrics

To analyse the performance of Hybrid QGNN, it is compared with Classical GCNN through different performance metrics as shown in Eqs (11–15). The Classical GCNN is compared with the proposed Hybrid QGNN methodology.

$$\text{Accuracy} = \frac{\text{Number of Correct Predictios}}{\text{Total Number of Predictions}} \quad \ldots (11)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \quad \ldots (12)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad \ldots (13)$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision+Recall}} \quad \ldots (14)$$

Here, TP – True Positive, FN – False Negative, and FP – False Positive.

Measurement

Upon processing quantum information for feature extraction via the QGNN layer, the output is generated in the form of tensors. The information is subsequently input into the classical segment of the model architecture, where it is reshaped and flattened to provide compatibility for post-processing in the dense layers. The model architecture comprises two dense layers; the first employs ReLU as the activation function, while the second utilizes the sigmoid function for processing. Subsequent to the dense layers, the information is represented in classical bit string format, which is subsequently processed using the SGD optimizer and employs binary cross–entropy as its loss function. Ultimately, the trained model provides the accuracy and loss metrics for both the training and validation datasets. The results are utilized to graph accuracy against validation and loss against validation loss to assess overfitting in the dataset. The model's performance is evaluated using a classification report and confusion matrix, which are detailed in the results and analysis section to show the presence or absence of a brain tumor. This systematic method guarantees that the dataset is effectively arranged and prepared for ensuing pre-processing stages.

Table 1 — Total samples

	Actual class	
Predicted class	Positive	Negative
Positive	TP	FN
Negative	FP	TN

Results and Discussion

The loss and accuracy for both the quantum and classical models are calculated for 100 epochs for the training and validation data. The accuracy and loss for quantum GNN model is presented in Fig. 7(c) & (d). At the left plot x-axis (epoch) represents the number of training iterations and y-axis (accuracy) indicates the classification accuracy, with higher values being better. The training accuracy (blue line) rapidly improves within the first 20 epochs, stabilizing around 0.64. This indicates that the model quickly learns to fit the training data. The Validation Accuracy (orange line) starts slower and plateaus near 0.53 after a few epochs. The validation accuracy shows no improvement after the initial training phase, suggesting that the model's performance on unseen data has stagnated. On the right Plot, loss vs. validation loss the x-axis (Epoch) represents the number of training iterations and y-axis (loss) measures the error or mismatch between predicted and actual values, with lower values being better. The training loss (blue line) rapidly decreases within the first 20 epochs and then stabilizes around 0.65, indicates effective minimization of loss on the training set. Validation Loss (orange line) Initially decreases but flattens and even slightly increases after around 10 epochs. The decrease in loss demonstrates the models strong error reduction capabilities. Similarly, an increase in accuracy confirms the model's competence in correctly classifying brain tumor images.

However, the classical GCNN model as shown in the Fig. 7(a) & (b) the training accuracy (blue line) remains constant at a value around 0.4 and the validation accuracy remains constant at a value around 0.32 which is shown in the left plot accuracy vs validation accuracy. This shows that the model's accuracy is not improving over time indicating poor model architecture and insufficient learning rate. The right plot loss vs validation loss shows consistent decrease in the training loss which indicates that the model is reducing its error on the training data. For the validation loss also, there is a decrease but remains significantly lower than the training loss. This shows that the model is overfitted, as the validation accuracy does not improve despite decreasing validation loss.

Our comprehensive evaluation of the hybrid QGNN algorithm for brain tumor MR image classification

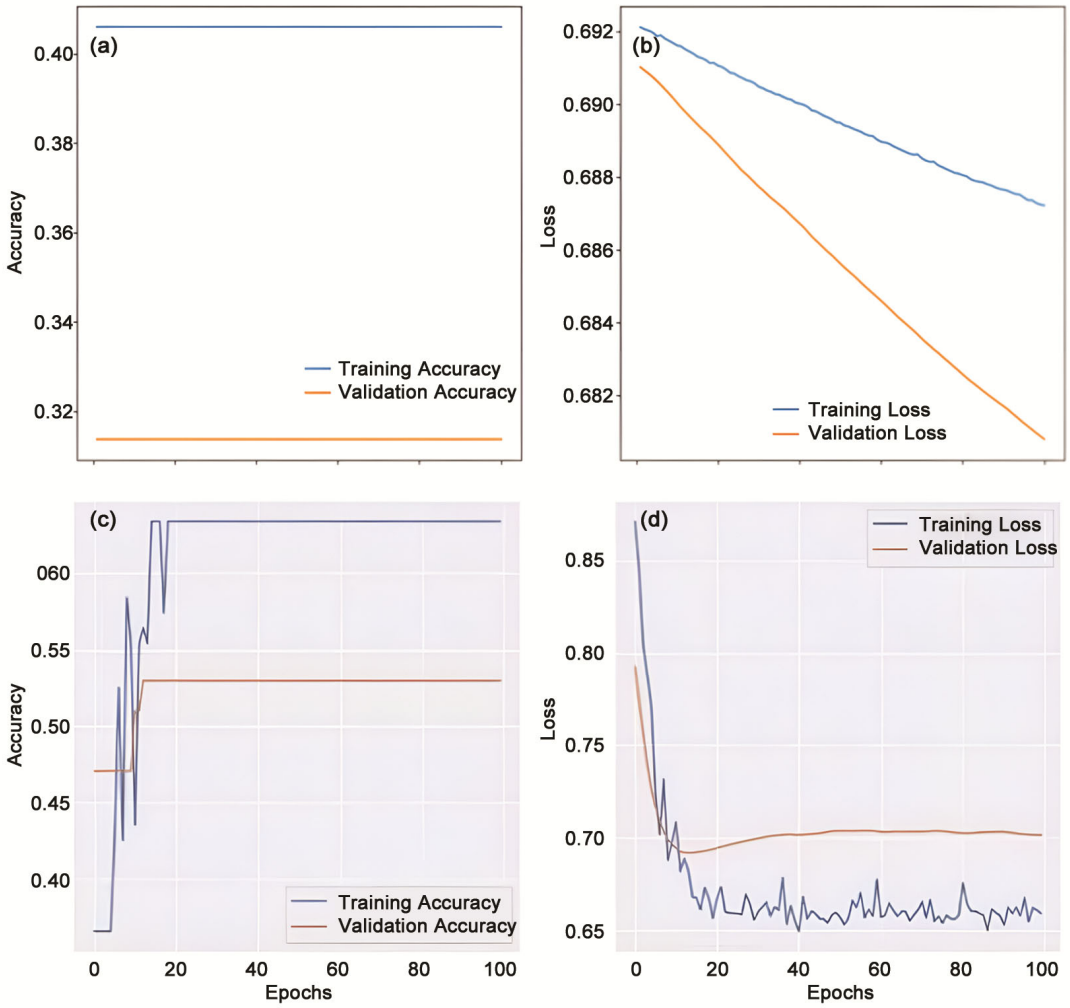

Fig. 7 — Accuracy and Loss : (a) Accuracy of classical GCNN model, (b) Loss in classical GCNN model, (c) Accuracy of quantum GNN model, (d) Loss in quantum GNN model

demonstrates the superior performance across multiple metrics. The comparative analysis of quantum GNN model with the classical GCNN is shown in Table 2, which demonstrates training accuracy, training loss, validation accuracy, and validation loss. The hybrid QGNN outperforms the classical GCNN by 23% for training data and 21% for the validation data.

To compare both of these models more comprehensively, other performance metrics, that is, confusion matrix and classification report are used. The confusion matrix for QGNN and classical GCNN are presented in Fig. 8.

The performance of the model with respect to accuracy, recall, precision, and F1 score for quantum and classical models are presented in Table 3. The hybrid quantum GNN results summarized in Table 3 illustrate an accuracy of 0.53 (53%), precision of 0.26 (26%), recall of 0.50 (50%), and F1 score of 0.35 (35%). The classical GCNN results illustrate an accuracy of 0.31 (31%), precision of 0.16 (16%), recall of 0.50 (50%), and F1 score of 0.34 (34%).

Table 2 — Performance metrics for Classical GCNN & Hybrid QGNN

Metric	Classical GCNN	Hybrid QGNN
Training accuracy	0.4159	0.643663356
Training loss	0.6872	0.659687936
Validation accuracy	0.3137	0.529411793
Validation loss	0.6908	0.693463008

In light of the classification report, the precision and F1 score have also improved for different classes, where 0 labels normal images and 1 labels brain tumor images. In the context of the confusion matrix, from Fig. 8, it is clearly visible that the hybrid QGNN model is predicting more efficiently compared to the classical GCNN.

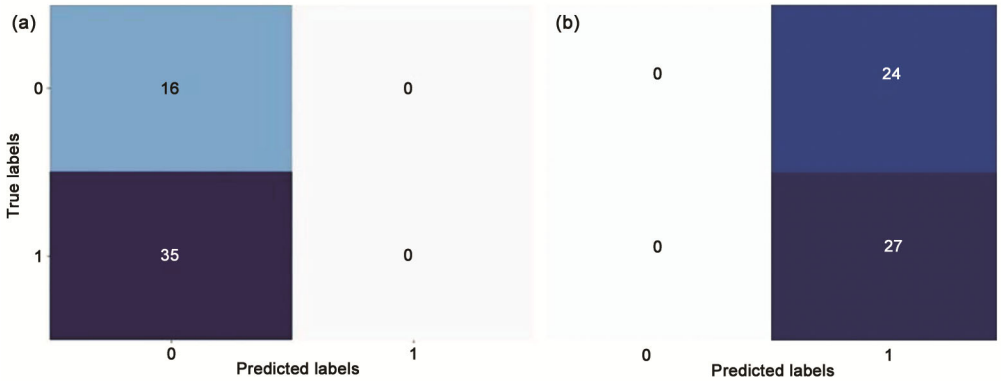

Fig. 8 —Confusion matrix for: (a) Classical GCNN model, (b) Hybrid Quantum GNN model

Table 3 — Classification report for both the models

	Precision	Recall	F1–score	support
Classification report for Hybrid Quantum GNN				
Without tumor	0.00	0.00	0.00	24
With tumor	0.53	1.00	0.69	27
Accuracy			**0.53**	51
Macro SVG	0.26	0.50	0.35	51
Weighted Avg	0.28	0.53	0.37	51
Classification report for Classical GCNN				
Without tumor	0.31	1.00	0.48	16
With tumor	0.00	0.00	0.00	35
Accuracy			**0.31**	51
Macro SVG	0.16	0.50	0.34	51
Weighted Avg	0.10	0.31	0.15	51

Conclusions

Classical GCNN and hybrid QGNN represent two distinct yet complementary paradigms for learning from graph–structured data. While classical GCNNs have achieved notable success, hybrid QGNNs hold the promise of unprecedented capabilities through the unique properties of quantum computing. A hybrid quantum graph neural network architecture is introduced to efficiently and accurately diagnose benign and malignant brain illnesses. The dataset utilized for this work is Brain MRI for brain tumor classification, accessible on Kaggle. The hybrid QGNN model is executed on a quantum simulator called default qubit provided by the PennyLane library with 5 qubits. In terms of accuracy and loss, the hybrid QGNN outperforms classical GCNN. According to comparisons between the two models, in the training and validation dataset, the training accuracy improved from 0.41 (41%) to 0.64 (64%), and validation accuracy improved from 0.31 (31%) to 0.52 (52%). This shows the efficiency of the model for classifying normal and brain tumor images. Continued research in this area may yield powerful tools for solving complex problems across various domains.

References

1 Deepak S & Ameer P, Automated categorization of brain tumor from mri using CNN features and SVM, *J Ambient Intell Humaniz Comput*, **12(8)** (2021), 8357–8369

2 Díaz–Pernas F J, Martínez–Zarzuela M, Antón–Rodríguez M & González–Ortega D, A deep learning approach for brain tumor classification and segmentation using a multiscale convolutional neural network *Healthcare*, **9**(2021) 153

3 Konar D, Bhattacharyya S, Panigrahi B K & Behrman E C, Qutrit–inspired fully self–supervised shallow quantum learning network for brain tumor segmentation, *IEEE Trans Neural Netw Learn Syst*, **33(11)** (2021) 6331–6345

4 Amin J, Anjum M A, Sharif M, Jabeen S, Kadry S & Moreno Ger P, A new model for brain tumor detection using ensemble transfer learning and quantum variational classifier, *Comput Intell*, **2022(1)** (2022) 3236305.

5 Vidyarthi A, Agarwal R, Gupta D, Sharma R, Draheim D & Tiwari P, Machine learning assisted methodology for multiclass classification of malignant brain tumors, *IEEE Access*, **10** (2022) 50624–50640

6 Zain Eldin H, Gamel S A, El–Kenawy E–S M, Alharbi A H, Khafaga D S, Ibrahim A & Talaat F M, Brain tumor detection and classification using deep learning and sine–cosine fitness grey wolf optimization, *Bioengineering*, **10(1)** (2022) 18

7 Mahmud M I, Mamun M & Abdelgawad A, A deep analysis of brain tumor detection from mr images using deep learning networks, *Algorithms*, **16(4)** (2023) 176

8 Choudhuri R & Halder A, Brain MRI tumor classification using quantum classical convolutional neural net architecture, *Neural Comput Appl*, **35(6)** (2023) 4467–4478

9 Darwish S M, Abu Shaheen L J & Elzoghabi A A, A new medical analytical framework for automated detection of MRI brain tumor using evolutionary quantum inspired level set technique, *Bioengineering*, **10(7)** (2023) 819.

10 Sharma A, Tselykh A, Bozhenyuk A & Kim B G, Image and video analysis using graph neural network for internet of medical things and computer vision applications, *CAAI Trans Intell Technol*, (2024)

11 Ravinder M, Saluja G, Allabun S, Alqahtani M S, Abbas M, Othman M & Soufiene B O, Enhanced brain tumor classification using graph convolutional neural network architecture, *Sci Rep*, **13(1)** (2023) 14938

12 Liao Y, Zhang X–M & Ferrie C, Graph neural networks on quantum computers, *arXiv preprint arXiv:2405.17060*, (2024)

13 Chakrabarty, "Kaggle," [Online], Available: Https://Www.Kaggle.Com/Datasets/Navoneel/Brain–Mri–Images–for–Brain–Tumor–Detection.,

14 Babu Vimala B, Srinivasan S, Mathivanan S K, Mahalakshmi, Jayagopal P & Dalu G T, Detection and classification of brain tumor using hybrid deep learning models, *Sci Rep*, **13(1)** (2023) 23029

15 Shoaib M R, Zhao J, Emara H M, Mubarak A F, Omer O A, Abd El–Samie F E & Esmaiel H, improving brain tumor classification: an approach integrating pre–trained CNN models and machine learning algorithms, *Heliyon*, (2024)

16 Parmar S J, Parmar V R, Verma J, Roy S & Bhattacharya P, Quantum computing: exploring superposition and entanglement for cutting–edge applications, *16th Int Conf Sec Info Net (SIN)*, (2023) 1–6

17 Rath M & Date H, Quantum data encoding: a comparative analysis of classical–to–quantum mapping techniques and their impact on machine learning accuracy, *EPJ Quantum Technol*, **11(1)** (2024) 72

18 Chandra S, Saxena S, Kumar S, Chaube M K & Bodhey N K, A novel framework for brain disease classification using quantum convolutional neural network, *IEEE Int Women in Engg (WIE) Conf Elec Comp Engg (WIECON–ECE)*, (2022) 346–351