Journal of Theoretical and Applied Information Technology
31st July 2025. Vol.103. No.14

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

EXPLORING QUANTUM MACHINE LEARNING ALGORITHMS FOR ENHANCED CHEMICAL TOXICITY CLASSIFICATION

KALIDINDI VENKATESWARA RAO[1], KUNJAMNAGESWARA RAO[2], GOKURUBOYINA SITARATNAM[3]

[1]Research Scholar, Department of Computer Science & Systems Engineering, Andhra University, Visakhapatnam, Andhra Pradesh, India
[2]Professor, Department of Computer Science & Systems Engineering, Andhra University, Visakhapatnam, Andhra Pradesh, India
[3]Research Scientist, Institute of Bioinformatics and Computational Biology (Recognized as SIRO), Visakhapatnam, India
E-mail: [1]kvrao.rs@andhrauniversity.edu.in , [2]kunjamnag@gmail.com, [3] sitagokuruboyina@gmail.com

ABSTRACT

Classical computational approaches are steadily improving in their ability to predict molecular properties based on their sequences or structure, which is essential in drug discovery, irrespective of its many challenges. The major challenge is to use computational approaches to identify the properties, such as toxicity, solubility, etc. (ADMET), of the drug targets without costing much time. Machine learning and deep learning methods are employed for predicting ADMET properties. The proposed approach combines quantum computing with machine learning for the binary task of toxicity prediction based on the SMILES data. This Quantum Machine Learning (QML) approach involves a novel five-step procedure that converts SMILES to their respective molecular fingerprints, which are in turn dimensionally reduced, consequently forming four qubits. These qubits are inputs to the QML models, such as the Quantum SVC (QSVC), Variational Quantum Circuits (VQC), and Quantum AutoEncoders (QAE). The models are trained and optimized by using different optimizers and then evaluated based on the accuracy metric. The QAE model outperformed the remaining model by achieving an accuracy of 99%.

Keywords: *ADMET, SMILES, QML, QAE, QSVC, VQC*

1. INTRODUCTION

In this era of computer-aided drug discovery, determining the side effects of drugs remains a major challenge, particularly because of the complexity of biological systems. Drug safety concerns are a major reason for the discontinuation of medication in the preoperative and postoperative periods [1]. In recent years, many clinical trials have failed, often due to unexpected toxicities. As such, a risk assessment tool has proven to be very useful, particularly in the preliminary phases of drug discovery, which helps in selecting compounds that minimize risk after drug testing. In this context, it is quite beneficial to employ machine learning models for toxicity classification across various toxicity databases. With the aid of chemical structural data, these models identify molecular patterns and predict biological properties, which makes early toxicity testing faster and more efficient. Recently, QML has shown a promising approach to enhance these predictions by aiding quantum computing's ability to process complex data patterns at high speed. Quantum computing excels in handling higher-dimensional features paces compared with classical computing, with quantum kernel functions being especially powerful in managing these spaces. For certain types of classification problems, quantum kernels have demonstrated the ability to speedup over their classical counterparts exponentially [2]. Several QML variants exist, including the QSVC, VQC, and quantum nearest neighbor (Qk-NN). The choice of classification depends on the type of data and the processing process [3].

QML demonstrates that quantum computing and machine learning can be successfully combined. In QML, there are four major classes out of which we use classical data sets that are converted to quantum states using quantum feature maps or quantum simulators. These quantum states are then processed using quantum circuits, and the outputs are fed into machine learning algorithms for both analysis and classification [4]. This hybrid approach uses the

power of quantum and classical computers to solve problems more efficiently.

2. PROPOSED METHODOLOGY

A QML workflow was developed as a five-step model for toxicity prediction based on SMILES data. Initializing the procedure, SMILES strings are converted into molecular finger prints, a numerical representation of the molecule's structural features. Secondly, to reduce the complexity of molecular fingerprints, dimensionality reduction techniques, such as PCA, are employed. Third, quantum embedding, where the reduced data is encoded into a quantum state suitable for quantum processing. Fourth, the aforementioned data is processed using different QML models, such as QSVC, VQC, and QAE, to learn and predict toxicity. The final step involves testing the model and evaluating its performance.

2.1. Preprocess the SMILES

In this initial step, the RDK it package was used to create a molecular object from the SMILES data. Canonical SMILES strings were then generated from SMILES to standardize the atom order and ensure consistency. This process includes handling stereo chemistry and the use of lower-case notation to indicate aromacity, ring closure, hydrogen suppression, and different bond types. Additionally, tautomer sand resonance structures were considered to ensure uniform representation across different molecular forms. This step is crucial to maintain uniform length of arrays for SMILES strings

2.2. Encoding

Canonical SMILES strings are encoded into molecular finger prints, which are binary vectors generated to indicate whether certain molecular features or fragments are present in a molecule. Each feature is indicated by a binary digit (0 or 1) in the fingerprint. These fingerprints are directly related to the discovery and characterization of Quantitative Structure-Activity Relationship (QSAR) analysis [5]. In the proposed study, Morgan fingerprints and MACCS keys were considered. These are used based on the model efficiency of working with fingerprint data of particular type.

2.3. Quantum Embeddings

Quantum algorithms require embedding classical data into quantum states, which are representations of data in the quantum realm, often as a superposition of the basic states. Different methods of embedding quantum states include basic encoding, amplitude encoding, and angle encoding. In the proposed model, amplitude encoding and angle encoding are chosen based on the model's requirements. Amplitude encoding is used for two of three models and angle embedding for one model. The choice of encoding method impacts the efficiency and accuracy of quantum computations, depending on the nature of the dataset and the quantum circuit constraints.

2.4. Quantum Model

Quantum circuits are built by using universal quantum gates, which manipulate qubits and their states according to principles of quantum mechanics. These gates, including Pauli gates (X, Y, Z) and the Hadamard gate, are used to design quantum circuits adjusted to specific requirements [6-7]. Quantum states can exist in superpositions, and measurements collapse these states into one of the basis states for each qubit, producing classical bits as the outputs. To develop a QML model, the designing and training involve selecting an appropriate quantum algorithm that matches the problem. Examples include Quantum Support Vector Machines (QSVM), which use quantum kernels to separate data in high-dimensional space, and Quantum Neural Networks (QNN), which mimic classical neural networks. To minimize the cost function, the training involves adjusting the parameters of a quantum circuit. During the quantum measurement phase, measurements are repeated multiple times to build a probability distribution, providing insights into quantum computation outcomes.

2.5. Evaluation

Finally, the model was evaluated by using 20% of the unseen data from the Clintox Dataset based on the accuracy metric. The evaluation results provided insights into the model's generalization capability in predicting toxicity. A comparison with baseline models demonstrated the effectiveness of the proposed approach. This step also examines the overfitting and underfitting of the models. To provide a thorough analysis, the model's performance was also evaluated using additional important metrics like precision, recall, and F1-score. The results indicated that the proposed method outperformed traditional machine learning models in distinguishing toxic and non-toxic compounds. Furthermore, visualizing the model's predictions through confusion matrices helped in understanding misclassification patterns and areas for improvement.

Journal of Theoretical and Applied Information Technology
31st July 2025. Vol.103. No.14

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

3. PREPROCESSING SMILES

Pre-processing of SMILES is essential for reducing redundancy and ensuring consistency. During pre-processing, the SMILES strings were canonicalized, standardized, and validated. They are then transformed into molecular fingerprints. The Molecular Fingerprints of a same length are used to train the models.

3.1. Canonicalization of SMILES

In this step, the SMILES strings are converted into a standardized and unique representation of the molecule. The canonical form of SMILES standardizes the order of atoms and uses lowercase notation to indicate aromacity, thereby ensuring a consistent structure. It also accurately preserves the stereochemistry [8]. The rules for canonicalization are as follows [9]:

3.1.1. Atom ordering and Handling Stereo-chemistry: Atoms in the molecule are consistently ordered based on their connectivity and atomic properties. A unique order is assigned to each atom to ensure that the SMILES string is generated in a consistent sequence. The stereochemistry (3D arrangement) of the molecule was preserved and encoded using standardized symbols. The canonical form ensures that stereo-centers are consistently represented.

3.1.2. Aromaticity perception and Branching:

Aromatic atoms and bonds are represented by lowercase letters and specific notations. Aromaticity must be detected and consistently applied across molecules. Ring Closures: Ring closures are consistently represented by assigning unique identifiers to atoms that close a ring. The same rings are always closed in the same order. Branches in the molecule are consistently ordered and are represented in parentheses in the SMILES string.

3.1.3. Canonicalization Example:

Consider the molecule 2-butanol. This molecule has the following two possible SMILES strings.

i. **SMILES 1:** "CC(C)COH" →2−butanol
ii. **SMILES 2:** "CC(O)C(C)C" →2−butanol

After canonicalization, SMILES strings SMILES 1 and SMILES 2 are converted to the canonical SMILES "CC(C)CO". This unique representation is now consistent, ensuring that the molecule is identified in the same manner across different datasets or processing steps.

3.2. Standardization and Validation of SMILES

Standardization of SMILES includes steps such as removal of salts or solvents, neutralization, and handling of tautomeric forms. After these steps, the SMILES strings are converted into a consistent format that adheres to specific rules or conventions, ensuring that all molecules are uniformly represented [10]. This process helps in reducing redundancy in molecular datasets and improves the reliability of machine learning models that are trained on SMILES representations. Validation of SMILES was performed to ensure chemical validity, prevent errors in analysis, and maintain data quality control. Structural validation, stereochemistry validation, SMILES syntax validation, detection, and handling invalid SMILES are the steps involved in SMILES validation [11].

4. ENCODING SMILES

4.1. SMILES conversion to Molecular-Fingerprints

In this step, smile strings, which are in textual form, are converted to molecular fingerprints, which are numerical vectors that capture the structural features of the molecule. Molecular Fingerprints: These fixed-length binary or integer vectors, where each bit or value represents the presence or absence of a particular substructure, atom pair, or pattern within a molecule. Different types of molecular fingerprints capture different aspects of a molecule's structure [12]. Common types of fingerprints include the following:

i. Topological fingerprints (e.g., Morgan or ECFP): Encodes atom connectivity.
ii. Path-based finger prints (e.g., daylight): Encode paths for atoms and bonds.
iii. MACCS keys: A set of predefined structural keys.

There are various types of molecular fingerprints, each with its own strength, and are chosen according to specific requirements. For instance, organ fingerprints are frequently used in similarity search. In the proposed work, MACCS (Molecular ACCess System) key fingerprints are used due to their simplicity and interpretability. MACCS keys are a set of predefined structural fragments or features, typically 166. Each key corresponds to a specific chemical feature, such as the presence of a particular atom, bond type, or functional group [13]. For example, consider the case of benzene. The

Journal of Theoretical and Applied Information Technology
31st July 2025. Vol.103. No.14

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

MACCS key fingerprint for benzene was a binary vector with a length of 166. Each bit in this vector corresponds to a MACCS key. The conversion process is illustrated in Figure 1.

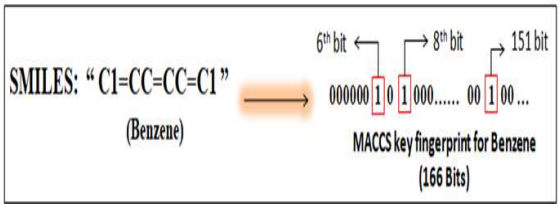

Figure 1: SMILES conversion to Molecular Fingerprints

In this example:

i. Bit 6 → 1, indicating the presence of an aromatic ring.
ii. Bit 8 –> 1 indicates the presence of a double bond.
iii. Bit 151 − > 1, indicating the presence of a carbon atom.
iv. Other bits would be zero if these particular features were not present in benzene.

4.2. Normalization of MACCS Fingerprint Vectors

Normalization is necessary when SMILES strings are embedded in the quantum states. It is important to obtain consistent data [14]. In the proposed model, the generated MACCS fingerprint vectors are normalized using vector normalization.

Vector Normalization:

i. The normalization step calculates the Euclidean norm of the MACCS fingerprint vector, which is essentially the square root of the sum of the squares of the vector elements.
ii. For a binary vector, the norm is the square root of the sum of squares of the 1s in vector. Because the vector is binary, the sum of squares is simply the number of 1's.
iii. After computing the norm, each element in the vector is divided by this norm to convert the binary vector into a vector of real numbers. The resulting normalized vector has a length of one, which ensures that all vectors are on the same scale [15].

After Normalization, normalized vectors of equal lengths are suitable for quantum circuits.

4.3. Dimensionality Reduction

In the proposed model, dimensionality reduction was applied to the normalized vectors using Principal Component Analysis (PCA), and the number of dimensions was reduced to four. Dimensionality reduction was used for the following purposes:

4.3.1 Simplification

MACCS fingerprints are 166-dimensional vectors, which can be complex and computationally expensive to process, especially when using quantum machine learning algorithms.

4.3.2 Noise Reduction

High-dimensional data can contain a lot of noise or irrelevant information that may not contribute significantly to the predictive power of the model. Reducing the number of dimensions can help filter out this noise.

4.3.3 Working with Principal Component Analysis (PCA)

PCA is a technique that reduces the dimensionality of data by identifying the principal components along which the variance of the data is maximized. It transforms the original features into principal components that are uncorrelated and ordered by the amount of variance explained [16]. PCA helps in extracting the most important features that capture the majority of the information in high-dimensional data, while ignoring less important ones. In the proposed model PCA plays crucial role in reducing the number of dimensions of training data.

4.3.4 Number of Dimensions

Embedding data into quantum states requires the encoding of features in quantum circuits. The complexity of these circuits increases with an increase in the number of dimensions. Reducing to four dimensions simplifies the quantum circuit and makes the model more efficient and feasible for running on current quantum hardware.

5. QUANTUM EMBEDDINGS

To use quantum algorithms, classical data must be converted into quantum states. A quantum state is the representation of data in the quantum realm, often a superposition of basic states. Therefore, classical data are mapped to quantum states; this process is known as quantum embedding. Different types of quantum embedding provide various methods for representing and manipulating data within a quantum space. In this study, amplitude

Journal of Theoretical and Applied Information Technology
31st July 2025. Vol.103. No.14

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

and variational encoding were employed as quantum embeddings [17].

5.1. Amplitude Encoding

The amplitude of quantum states refers to the coefficients of the basis in the quantum state vector. This determines the measurement of the probabilities of the quantum states in each basis state. Understanding and manipulating these amplitudes are fundamental for quantum computation and information processing. By encoding data into the amplitudes of quantum states, quantum computers can process multiple data points simultaneously, which is called quantum parallelism. This approach is well suited for these numerical vectors, which enables quantum computers and algorithms to process and learn molecular data efficiently [18].

5.1.1. Amplitude Encoding of SMILES

To understand amplitude encoding, consider a simplified example in which SMILES strings have been preprocessed and converted into a numerical representation, such as a molecular fingerprint or a feature vector. In this case, a benzene molecule is used as an example, and its SMILES string is transformed into a 4-dimensional feature vector. The benzene SMILES is represented as a feature vector [0.5, 0.2, 0.3, 0.1]. The feature vector is normalized to ensure that the sum of the squares of its components equals 1, as required for the quantum states. The normalization of the vector is shown in equation (1) and (2).

Normalized vector:

$$Norm = \sqrt{0.52 + 0.22 + 0.32 + 0.12} = \sqrt{0.25 + 0.04 + 0.09 + 0.01}$$

$$= \sqrt{0.39} \approx 0.624 \quad (1)$$

The normalized vector becomes

$$[0.5/0.624, 0.2/0.624, 0.3/0.624, 0.1/0.624] \approx [0.801, 0.321, 0.481, 0.16] \quad (2)$$

5.1.2. Encoding into Quantum State

The normalized vector was then used to create a quantum state. In amplitude encoding, this vector is mapped directly onto the amplitudes of the quantum state as shown in equation (3).

$$\textbf{Quantum state:}|\psi\rangle = 0.801|00\rangle + 0.321|01\rangle + 0.481|10\rangle + 0.160|11\rangle \quad (3)$$

Each amplitude corresponds to a possible quantum state of the qubits (e.g., $|00\rangle$, $|01\rangle$, $|10\rangle$, $|11\rangle$ for a 2-qubit system).

5.2. Variational Encoding

In variational encoding, a normalized vector is used as the input parameter for a quantum circuit. These parameters are tuned within a quantum algorithm, often using a variational quantum algorithm (VQA) to minimize the cost function related to the problem being solved. Considering the previous example of benzene, parameters from the normalized vector [0.801, 0.321, 0.481, 0.160] could be used as rotation angles in quantum gates (e.g., RX, RY, and RZ gates) within a quantum circuit. The quantum state encoding can be expressed as shown in equation (4).

$$|\psi\rangle = RY(0.801) \bullet RZ(0.321) \bullet RX(0.481) \bullet RY(0.160) \bullet |0\rangle \quad (4)$$

Where RY, RZ, and RX are the rotation gates applied to the initial state $|0\rangle$. The quantum state was then processed using the quantum circuit, with the parameters adjusted during the training process to optimize the circuit's performance for a specific task, such as classification or regression.

5.3. ZZFeatureMap

ZZFeatureMap is a type of quantum feature map used to encode classical data into quantum states using parameterized quantum circuits. Consider a benzene molecule as an example to understand how the feature map works. For a 4-dimensional feature vector, ZZFeatureMap is set as

i. Feature Dimension: 4
ii. Repetitions: 2 (The feature map was applied twice to increase expressivity).
iii. Entanglement: linear (qubits are entangled in a linear chain).

A normalized feature vector [0.801, 0.321, 0.481, 0.160] was used to parallelize the rotations in the ZZFeatureMap. ZZFeatureMap applies parameterized rotations and entanglements to prepare a quantum state that encodes the feature vector. The quantum state after applying the ZZFeatureMap will be a superposition of the basis states, where the amplitudes are influenced by the encoded features, and the exact form of the quantum state will depend on the details of the feature map and the specific quantum circuit implementation.

Journal of Theoretical and Applied Information Technology
31st July 2025. Vol.103. No.14

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

6. QUANTUM MODELS

Quantum models are advanced tools that use quantum computing for data analyses and machine learning. These models use quantum mechanical principles such as superposition and entanglement to handle complex problems more efficiently than classical models. VQC, QSVC, and QAE are quantum models that are trained and assessed using the ClinTox dataset. Each model and the model training process are covered in the following subsections.

6.1. Implementation of VQC

The five-step proposed framework for implementing VQC is covered in this subsection. Variational Circuits are designed by using different quantum gates as per the requirement. The choice of quantum gates and their arrangement plays a crucial role in optimizing the circuit's performance. Additionally, parameterized quantum circuits are trained using classical optimization techniques to minimize a predefined loss function and achieve better predictive accuracy.

6.1.1. SMILES preprocessing: The SMILES strings are converted into a numerical representation, such as molecular fingerprints (e.g., MACCS keys). In this step, the chemical structure is converted into a form that can be processed using a quantum algorithm. A 166-bit fingerprint is generated for each SMILES string in this step.

6.1.2. DataEncoding: Molecular fingerprints are then encoded into quantum states using a feature map. In this study, ZZFeatureMap was used to map the data into a quantum circuit. This feature map captures interactions between molecular features by applying entangling gates. The choice of ZZFeatureMap ensures that relevant molecular relationships are preserved while reducing the dimensionality of the input space.

6.1.3. Variational Circuit: A parameterized quantum circuit is designed and applied to the quantum state, which might involve layers of quantum gates, such as rotations (RX, RY, RZ), and entanglement gates, such as CNOT [19]. The parameters of these gates were initially randomly set. For instance, consider equation (5):

$$|\psi\rangle = RY(\theta 1) \cdot RZ(\theta 2) \cdot RX(\theta 3) \cdot Entanglement \cdot |Encoded\ State\rangle \quad (5)$$

An ansatz (a quantum circuit used in variational algorithms) is created using the real amplitudes circuit. The ansatz consists of multiple layers of rotation gates (RX and RY) and entanglement layers. In this code, the ansatz has three repetitions of these layers, making it capable of capturing the complex patterns in the data.

6.1.4 Training the VQC: The VQC was trained using a labeled dataset of regular fingerprints (e.g., toxic vs. non-toxic compounds). A quantum circuit is executed, and the output is compared with the true labels. The classical optimizer adjusts the parameters to improve the classification accuracy. The variational parameters in the ansatz were optimized using the COBYLA optimizer. This optimizer iteratively adjusts parameters to minimize the objective function (classification error).

6.1.5 Prediction: After training, the VQC can classify a new compound's SMILES-derived fingerprint as toxic or non-toxic based on the output of the quantum circuit. The classification is performed by measuring the qubits and interpreting the probability distribution of the outcomes. The model's performance is evaluated using standard metrics like accuracy, precision, recall, and F1-score to ensure reliable predictions.

6.2. Quantum Support Vector Classifier (QSVC)

Support vector classifiers (SVC), which use quantum kernels, are known as QSVC. SVC is a supervised machine learning algorithm used for classification. This algorithm works by finding the optimal hyperplane that separates different classes in the feature space. To deal with nonlinear separable data, these algorithms use kernels, which transform them into higher-dimensional spaces where a linear separator can be found [20]. Classical SVC is computationally expensive in the case of high-dimensional feature spaces. By embedding data into quantum states, QSVC can accelerate the process of offending the optimal separator and handling high-dimensional spaces [21].

6.2.1. Classical SVM Foundation

The aim is to determine the optimal hyperplane that maximizes the margin between two classes. Mathematically, it is formulated as shown in equation (6).

$$min_{\{w,b\}} \frac{1}{2}|w|_2^2$$
$$\text{Subject to: } yi(w * xi + b) \geq 1, \forall i \quad (6)$$

where 'w'is the weight vector, 'b' is the bias, and xi is the feature vector. In cases where the data are not linearly separable, the SVM uses a kernel

Journal of Theoretical and Applied Information Technology
31st July 2025. Vol.103. No.14

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

function $K(x_i, x_j)$ to implicitly map the data into a higher-dimensional space where it becomes linearly separable.

6.2.2. Quantum Feature Map

In QSVC, classical data are mapped into quantum states using a quantum feature map represented as $\phi(x)$. The quantum state $|\phi(x)\rangle$ encodes the classical feature vector x in the quantum state [22]. ZZFeatureMap is used to embed classical features into a quantum state. This feature map is parameterized and has an entanglement strategy specified as "linear" meaning that qubits are entangled in a linear configuration. reps=2 indicates that the feature map is repeated twice, allowing for a richer quantum representation of the data.

6.2.3. Quantum Kernel Computation

The quantum kernel is computed as the inner product of quantum states corresponding to different data points as shown in equation (7):

$$K_{Q(x_i,x_j)} = \left|\langle \phi(x_i), \phi(x_j)\rangle\right|^2 \quad (7)$$

This method measures the similarity between data points in the quantum feature space. A quantum kernel, the FidelityQuantumKernel, was constructed using the ComputeUncompute fidelity method. This kernel measures the similarity between quantum states corresponding to different data points. The kernel is then used by QSVC to distinguish between different classes (e.g., toxic vs. non-toxic molecules) based on these quantum state similarities.

6.2.4. Support Vector Classifier

The QSVC then uses this quantum kernel instead of the classical kernel to perform classification [23]. The decision function of the kernel is shown in equation (8).

$$f(x) = sign\left(\sum_{\{i=1\}}^{N} \alpha_i y_i K_{Q(x_i,x)} + b\right) \quad (8)$$

Here, αi is the coefficient of the support vector and yi is the class label. The Figure 2 presented a pipeline for predicting toxic and non-toxic compounds using the VQC and QSVC models.

6.3. Quantum Auto Encoders (QAE)

Classical autoencoders are variants of ANNs used to learn from efficient data for feature extraction. Autoencoders consist of two major components: encoders and decoder. Encoders are used for dimensionality reduction of the input data, while decoders are used to reconstruct the original data from the encoded form.

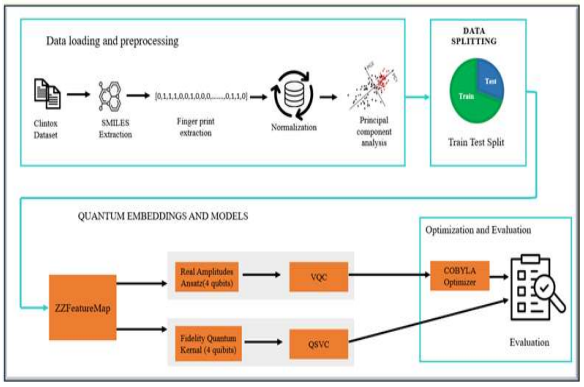

Figure 2. Pipeline for Toxicity Prediction using QSVC and VQC Models

The main purpose of using autoencoders is to capture important features while minimizing information loss. QAE works similarly to classical autoencoders, the only difference is that a classical autoencoder works on classical bits, whereas a QAE works on qubits.QAE consist of two main components [24]:

i. **Encoder:** By applying quantum gates, the encoder compresses the input quantum state into a small quantum state that preserves the most significant information while reducing its dimensionality.

ii. **Decoder:** The decoder attempts to reconstruct the original quantum state from the compressed state to match the input as closely as possible

6.3.1. Implementation of QAE

The PennyLane library was used to implement the QAE. The quantum device was initialized using PennyLane's default qubit simulator with four qubits. The SMILES strings were converted into MACCS fingerprints, which were binary vectors. These vectors were normalized to ensure that they fit the quantum framework. These normalized vectors were then encoded into quantum states using a parameterized quantum circuit. To increase the accuracy of the quantum autoencoder and reduce the reconstruction error, the circuit was optimized using a gradient-based optimizer.

6.3.2. Quantum Embedding

Normalized vectors are embedded into quantum states. In the proposed model Angular embedding was used to map the classical data to its quantum state space. Angular embedding is chosen because it allows direct and interpretable mapping of high-dimensional classical inputs to their quantum states

efficiently. In this method, each classical data point is encoded into the angles of quantum gates, typically rotation gates (such as RX, RY, and or RZ) applied to the qubits in a quantum circuit. For example, if a feature vector $x = [x_1,x_2,...,x_n]$ exists, each feature x_i can be encoded as a rotation angle for a qubit. The qubits are rotated by these angles, effectively embedding the classical data into a quantum state.

6.3.3. Quantum Encoder

This is designed to compress the quantum state to a lower-dimensional form, which involves several layers of quantum gates that entangle qubits to capture complex relations between features in quantum representation. After angle embedding, a series of entangling layers was applied using StronglyEntanglingLayers to compress the data. The output of the encoder is a set of expected values for the Pauli-Z operators on each qubit. The Pauli-Z operator is a quantum mechanical operator represented in the form of a matrix, and is one of the three Pauli matrices [25]. The Pauli-Z operator is defined as shown in equation (9)

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad (9)$$

This operator acts on a single qubit, and its effect on the basis states $|0\rangle$ and $|1\rangle$ is as shown in equation (10):

$$Z|0\rangle = |0\rangle$$
$$Z|1\rangle = -|1\rangle \quad (10)$$

The Pauli-Z operator flips the phase of the $|1\rangle$ state while leaving the $|0\rangle$ state unchanged.

6.3.4. Quantum Decoder

This is designed to reconstruct the quantum state from a lower dimensional form. Using the same AngleEmbedding and StronglyEntanglingLayers templates, the decoder circuit attempts to reconstruct the original data from the received encoded quantum states. An autoencoder function was used to tie the encoder and decoder together. By comparing the output quantum state with the initial input state, the loss function was created to quantify the reconstruction error. In order to reduce this error and improve the model's capacity to capture crucial features, the optimization process entailed adjusting the circuit parameters. In order to make sure the quantum autoencoder successfully learned to compress and reconstruct molecular fingerprints, several training iterations were conducted.

6.3.5. Loss Function

It used to measure the difference between the original and reconstructed outputs. It sums the Mean Squared Error (MSE) over all the training examples. A lower MSE value indicates that the model effectively captures the underlying patterns in the data. This metric helps in evaluating how well the autoencoder reduces dimensionality while maintaining the most important information. The model learns to produce reconstructions that closely resemble the original inputs by minimizing the MSE. To avoid overfitting and enhance the autoencoder's generalization, regularization techniques can also be used. Furthermore, monitoring the MSE over training epochs sheds light on the learning process's stability and convergence.

6.3.6. Training

Training is performed using the Broyden-Fletcher-Goldfarb Shanno (BFGS) optimization method, which minimizes the loss function by updating the weights of the encoder and decoder. BFGS is an iterative optimization algorithm used to solve unconstrained optimization problems, where the goal is to minimize or maximize the objective function. After training, the model was evaluated on both training and test datasets to compute the reconstruction error.

7. RESULTS

In this work, QML algorithms were applied to the Clintox dataset, which contains SMILES representations of chemical compounds as input data and toxicity labels based on FDA approval and clinical trial results as target labels. QSVC, VQC, and QAE are quantum models used for the binary classification task of predicting toxicity. The QML models were trained to categorize toxic and non-toxic compounds using SMILES as input. Evaluation Metrics to evaluate the performance of the QML models, several standard evaluation metrics, such as accuracy, precision, recall, and F1-score, have been used. However, this study focuses primarily on accuracy.

Training and testing accuracies are two distinct categories of accuracy that have been examined. The ability of the model to learn from the training set of data was measured based on its training accuracy. The testing accuracy measures the model's ability to generalize the prediction results on unseen data.

QSVC: The utilization of quantum kernels allows QSVC to take advantage of the quantum

states' ability to encode higher-dimensional features in a more compact and entangled form, allowing for better separability of complex data.

Results:

i. Training accuracy: 0.95 or 95%

ii. Testing accuracy: 0.94 or 94%

Analysis: The QSVC's performance on the training data suggests that the model has a solid understanding of the underlying data patterns. The small gap between the training and testing accuracies points to a relatively balanced model with minor overfitting. Testing accuracy demonstrated that the quantum kernel was able to effectively classify toxic and non-toxic compounds, but there were still a few misclassifications, indicating the potential for further improvement.

VQC: The VQC model uses parameterized quantum circuits (PQCs) for classification. The model's parameters are optimized to learn decision boundaries in a similar way to classical neural networks, but with quantum-enhanced capabilities. Figure 4 shows the objective function value against iterations during the training of the VQC. In the early iterations, the objective function increases slightly before decreasing. This suggests that the optimizer initially explored different parts of the parameter space, possibly encountering local maxima before converging to a better solution. The objective function showed notable oscillations between iterations 5 and 25. This could indicate that the optimizer is navigating a complex loss landscape with multiple local minima and maxima, causing fluctuations in the objective value. These oscillations are common in quantum variational circuits owing to the nonconvex nature of the optimization problem. After approximately the 25th iteration, the objective function starts to stabilize and exhibits smaller variations, hovering around a lower value. This suggests that VQC converges and fine-tunes its parameters

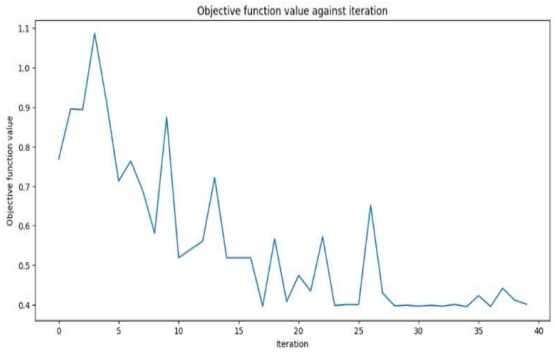

Figure 4. VQC Objective Function Convergence over Iterations

Results:

i. Training accuracy: 0.94 or 94%

ii. Testing accuracy: 0.93 or 93%

Analysis: The VQC achieved a training accuracy of 94%, showing that it learns to classify the training examples with high precision. This accuracy reflects the ability of VQC to fit the training data well through its parameterized quantum circuits. The small reduction in accuracy from training to testing suggests that the VQC is effective in learning the classification task but may benefit from improvements in circuit design or optimization techniques to further improve the performance.

QAE: The ability of the QAE to compress and encode information into a lower-dimensional latent space allows it to learn more efficiently and identify the most important features for toxicity classification.

Results:

i. Training accuracy: 0.99 or 99%

ii. Testing accuracy: 0.98 or 98%.

Analysis: The high accuracy of QAE on both the training and test sets suggests that the model generalizes well and is highly effective at denoising and reconstructing molecular data. This result highlights the power of autoencoders in quantum machine learning, particularly for tasks involving complex data, such as molecular toxicity. The Table 1 shows a comparison of the accuracies of QSVC, VQC, and QAE. The accuracies are the average performance achieved from two binary classification tasks of toxicity, with target labels for FDA-approved drugs and clinical trial outcomes considered separately. Among the models, the QAE per formed the best, achieving an accuracy of 98%.

Table 1: Accuracies comparison of QML Algorithms for Toxicity Classification on the ClinTox Dataset

Method	Accuracy
QSVC	0.94±0.01
VQC	0.93±0.01
QAE	0.98±0.005

8. CONCLUSION

The proposed framework introduces a structured approach for applying Quantum Machine Learning (QML) to SMILES data by leveraging specialized encoding techniques such as amplitude and variational encoding. These methods enable the

transformation of molecular data into quantum states, facilitating efficient quantum computation for toxicity prediction. By evaluating multiple QML algorithms—including Variational Quantum Classifiers (VQC), Quantum Support Vector Classifiers (QSVC), and Quantum Autoencoders (QAE)—the framework provides comparative insights into their predictive capabilities. This demonstrates the potential of quantum algorithms to enhance accuracy and uncover new patterns in molecular datasets, offering valuable contributions to computational drug discovery and materials science.

While the framework successfully highlights the applicability of QML in chemistry, it also brings attention to current limitations in the field. The reliance on noisy intermediate-scale quantum (NISQ) devices and the computational overhead introduced by encoding methods can limit real-world applicability at this stage. Additionally, performance evaluations conducted on simulators may not fully reflect behavior on actual quantum hardware. Despite these challenges, the work represents a promising step toward integrating quantum computing into molecular modeling and sets the stage for more scalable and hardware-optimized solutions in the near future.

REFERENCES:

[1] S. K. Niazi and Z. Mariam, "Computer-Aided Drug Design and Drug Discovery: A Prospective Analysis," Pharmaceuticals (Basel), vol. 17, no. 1, p. 22, Dec. 2023, doi: 10.3390/ph17010022.

[2] S. Karthikeyan, M. Akila, D. Sumathi, and T. Poongodi, Quantum Machine Learning: A Modern Approach, 2024, doi: 10.1201/9781003429654.

[3] M. Avramouli, I. Savvas, A. Vasilaki, G. Garani, and A. Xenakis, "Quantum machine learning in drug discovery: Current state and challenges," in Proceedings of the ACM International Conference on AI in Drug Discovery (AIDD), New York, NY, USA, 2023. doi: 10.1145/3575879.3576024.

[4] D. Peral-García, J. Cruz-Benito, and F. J. García-Peñalvo, "Systematic literature review: Quantum machine learning and its applications," Computer Science Review, vol. 51, p. 100619, 2024, doi: 10.1016/j.cosrev.2024.100619.

[5] Yi Ding, Minchun Chen, Chao Guo, Peng Zhang, Jingwen Wang, Molecular fingerprint-based machine learning assisted QSAR model development for prediction of ionic liquid properties, Journal of Molecular Liquids, Volume 326, 2021, 115212, ISSN 0167-7322, https://doi.org/10.1016/j.molliq.2020.115212.

[6] Simões RDM, Huber P, Meier N, Smailov N, Füchslin RM, Stockinger K. Experimental evaluation of quantum machine learning algorithms. IEEE Access. 2023;11:6197–208.

[7] H. Drias and F. Yalaoui, Proceedings of the conference series: QSAC: International Symposium on Quantum Sciences: Applications and Challenges, 2023. doi: 10.1007/978-3-031-59318-5.

[8] L. Wang, et al., "Chemical structure representation based on SMILES and its application in deep learning," Chemosphere, 2023, vol. 310, p. 136879.

[9] O'Boyle, N. M., Banck, M., James, C. A., Morley, C., Vandermeersch, T., & Hutchison, G. R. (2011). Open Babel: An open chemical toolbox. Journal of Cheminformatics, 3(1), 33. DOI: https://doi.org/10.1186/1758-2946-3-33

[10] Khan, M. I., et al., "Chemical representation using SMILES: A review." Journal of Molecular Graphics and Modelling, 2020, 101, 107723.

[11] D. Bonnard, et al., "Validation of molecular representations: A case study of SMILES and InChI," Journal of Cheminformatics, 2023, vol. 15, no. 1, p. 50.

[12] D. Rogers, M. Hahn, Extended-connectivity fingerprints, Journal of Chemical Information and Modeling 50 (5) (2010) 742–754. doi:10.1021/ci100050t.

[13] H. Kuwahara, X. Gao, Analysis of the effects of related fingerprints on molecular similarity using an eigenvalue entropy approach, Journal of Cheminformatics 13 (1) (2021) 27. doi: 10.1186/s13321-021-00506-2.

[14] B Zagidullin, Z Wang, Y Guan, E Pitkänen, J Tang, Comparative analysis of molecular fingerprints in prediction of drug combination effects, Briefings in Bioinformatics, Volume 22, Issue 6, November 2021, bbab291, https://doi.org/10.1093/bib/bbab291.

[15] Boldini, D., Ballabio, D., Consonni, V. et al. Effectiveness of molecular fingerprints for exploring the chemical space of natural products. J Cheminform 16, 35 (2024). https://doi.org/10.1186/s13321-024-00830-3.

[16] S. Mishra, U. Sarkar, S. Taraphder, S. Datta, D. Swain, R. Saikhom, S. Panda, M. Laishram,

Principal component analysis, International Journal of Livestock Research (2017) 1doi: 10.5455/ijlr.20170415115235.

[17] S. Lloyd, M. Schuld, A. Ijaz, J. Izaac, and N. Killoran, "Quantum embeddings for machine learning," arXiv:2001.03622 [quant-ph], 2020. [Online]. Available: https://arxiv.org/abs/2001.03622.

[18] J. Gonzalez-Conde, T. W. Watts, P. Rodriguez-Grasa, and M. Sanz, "Efficient quantum amplitude encoding of polynomial functions," Quantum, vol. 8, p. 1297, Mar. 2024. http://dx.doi.org/10.22331/q-2024-03-21-1297.

[19] N. Abdullayev, L. Aliyeva and J. Hasanov, "Comparative Study of Quantum to Classical Machine Learning Algorithms for Breast Cancer Classification," 2023 IEEE 17th International Conference on Application of Information and Communication Technologies (AICT), Baku, Azerbaijan, 2023, pp. 1-6, doi: 10.1109/AICT59525.2023.10313161.

[20] D. F. Locher, L. Cardarelli, and M. Müller, "Quantum Error Correction with Quantum Autoencoders," Quantum, vol. 7, p. 942, Mar. 2023. https://doi.org/10.22331/q-2023-03-09-942.

[21] S. N. Pushpak and S. Jain, "An Implementation of Quantum Machine Learning Technique to Determine Insurance Claim Fraud," 2022 10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 2022, pp. 1-5, doi: 10.1109/ICRITO56286.2022.9964828.

[22] L. Huynh, J. Hong, A. Mian, H. Suzuki, Y. Wu, and S. Camtepe, "Quantum-Inspired Machine Learning: A Survey," arXiv preprint, arXiv:2308.11269, 2023. [Online]. Available: https://arxiv.org/abs/2308.11269

[23] Y. Yu, M. Li, L. Liu, Y. Li, and J. Wang, “Clinical big data and deep learning: Applications, challenges, and future outlooks,” Big Data Min. Anal., vol. 2, no. 4, pp. 288–305, 2019, doi: 10.26599/BDMA.2019.9020007.

[24] Muhammad Aasim, Ramazan Katırcı, Alpaslan Şevket Acar, Seyid Amjad Ali,A comparative and practical approach using quantum machine learning (QML) and support vector classifier (SVC) for Light emitting diodes mediated in vitro micropropagation of black mulberry (Morus nigra L.),Industrial Crops and Products,Volume 213,2024,118397,ISSN 0926-6690,https://doi.org/10.1016/j.indcrop.2024.118397.

[25] L. Fidkowski, J. Haah, and M. B. Hastings, "Exactly solvable model for a beyond-cohomology symmetry-protected topological phase," Phys. Rev. B, vol. 101, no. 15, p. 155124, Apr. 2020.