Quantum and Classical Graph Convolutional Neural Networks for Protein Ligand Dissociation Constant Prediction

Azamat Salamatov
Dept. of Computer Science
University of Cincinnati
Cincinnati, Ohio, USA
salamaat@mail.uc.edu

Gowtham Atluri
Dept. of Computer Science
University of Cincinnati
Cincinnati, Ohio, USA
atlurigm@ucmail.uc.edu

***Abstract*—How long a drug stays bound to its target - the residence time - is now recognized as a stronger in vivo efficacy driver than binding affinity alone. Yet current machine learning (ML) models for dissociation kinetics (koff) ignore two critical sources of structure information: (1) the change in protein–ligand geometry across time, and (2) the ability to represent complex spatial interactions with fewer parameters. We extend a state-of-the-art spatial GNN for protein–ligand complexes with two innovations. First, we introduce a 2–timestep GCN+GRU model that learns structure changes before and after short molecular dynamics simulations. Second, we compress the model head using variational quantum circuits, preserving expressivity while removing 66% of parameters. On the full PDBbind-koff-2020 benchmark, temporal integration improves predictive accuracy, and quantum compression matches the full classical model's accuracy at a fraction of its size. Our results show that time and quantum structure are underexplored, high-leverage axes for advancing kinetic ML models used in drug design.**

***Index Terms*—Quantum Deep Learning, Ligand Dissociation Constant, Molecular Dynamics, Spatial Data, Drug Discovery**

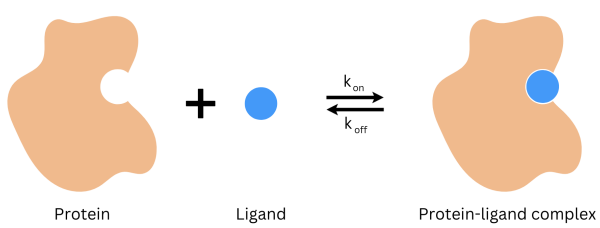

Fig. 1. Protein-ligand binding reaction pathway with the association rate constant (k_{on}) and the dissociation rate constant (k_{off})

I. Introduction

In structure-based drug design, understanding the dynamics of protein-drug interactions is crucial, particularly from both thermodynamic and kinetic perspectives. Thermodynamics (e.g., equilibrium constants: K_i, K_d, IC_{50}, etc.) describes the stability of the drug-protein complex at equilibrium, while kinetics (e.g., association and dissociation rates: k_{on}, k_{off}[1]) provides insights into how quickly the drug binds and unbinds from the target [23], [24]. Even drugs with similar binding strengths can differ significantly in their kinetic behaviors, which can influence their therapeutic effects [25]. Drugs that dissociate quickly generally require higher doses due to shorter action times, whereas those with slower dissociation rates stay bound longer, allowing for lower doses, enhanced selectivity, and potentially fewer side effects. These kinetic properties can therefore offer valuable guidance in optimizing drug efficacy and safety [25], [26], [27].

Currently, most studies assess the strength of drug-target protein interactions using thermodynamic properties, such as the equilibrium binding constant (K_i). These measurements are typically obtained from *in vitro* binding assays conducted in closed systems, where drug and protein concentrations remain constant. However, this setup does not reflect the dynamic, constantly changing conditions within the body. In the open, *in vivo* environment, drugs that bind or dissociate slowly may not reach equilibrium within the standard experimental timeframe, making it challenging to accurately capture

[1]In the Fig. 1, the association rate constant k_{on} defines the speed at which a ligand binds to its target protein, often measured in units of $\text{M}^{-1}\text{s}^{-1}$. A high k_{on} value suggests a ligand rapidly associates with the receptor, which can be crucial for fast-acting drugs. Conversely, k_{off} represents the dissociation rate constant, or how quickly the ligand unbinds from the receptor, with units of s^{-1}. A low k_{off} value indicates the ligand remains bound for a longer duration, contributing to prolonged drug efficacy.

their interaction characteristics under physiological conditions [23], [27].

Kinetic data on drug-target protein interactions are increasingly recognized as a more accurate indicator of a drug's *in vivo* efficacy compared to thermodynamic data [26]. Recent studies have shown that the pharmacological activity of drugs in living systems is more strongly correlated with kinetic properties, such as binding and dissociation rates, than with thermodynamic equilibrium constants [25]–[29]. For instance, research by the Heitman group demonstrated that the intracellular efficacy of A2A adenosine receptor agonists correlates closely with their residence time (with an R^2 of 0.90), while showing only a weak correlation with the equilibrium inhibition constant K_i (with an R^2 of 0.13) [28]. This highlights the importance of considering binding kinetics, alongside affinity, to enhance drug development outcomes and improve the likelihood of therapeutic success [27].

II. Related Work

Researchers are increasingly focused on creating computational approaches to predict dissociation rate constants (k_{off}), aiming to complement experimental methods. Traditional approaches often rely on computationally intensive kinetic simulations to study drug dissociation pathways [30]. Recently, some studies have turned to quantitative structure-activity relationship (QSAR) models to improve predictive performance for dissociation kinetics [31], [32].

Several QSKR models have been reported in the literature, typically focused on specific drug targets. In 2016, Mei Hu's group at Chongqing University developed a QSKR model for 37 HIV-1 protein inhibitors, using the 3D molecular force field Volsurf descriptor along with various physicochemical properties [31]. They employed partial least squares (PLS) and support vector machine (SVM) algorithms, achieving a correlation coefficient (R) of 0.772 on the test set for predicting dissociation rate constants. Similarly, in 2018, Wade et al. applied COMBINE analysis and PLS to construct a QSKR model for 66 HSP90 protein inhibitors, with a correlation coefficient (R^2) of 0.86 on the test set [32]. These studies, along with others [5]–[14], have shown that QSKR models are feasible, although their limited dataset sizes and single-target focus restrict their broader applicability in drug development.

In 2020, Su et al. developed a more general-purpose QSKR model using a random forest algorithm on the newly curated PDBbind-k_{off}-2020 dataset [34]. Their model, trained on 406 protein-ligand complexes, achieved a correlation coefficient (R) of 0.623 on the test set, demonstrating the potential of structure-based machine learning models for predicting k_{off}.

That same year, Amangeldiuly et al. introduced a QSKR model using structural descriptors, specifically atom-type contact counts within a binding site radius, to predict unbinding kinetics [27]. This model was trained on a dataset of 501 protein-ligand complexes with experimentally measured k_{off} values. Using random forest algorithms, they achieved a correlation coefficient (R^2) of 0.64, comparable to MD-based methods but with significantly reduced computational demands.

In 2023, Zhao et al. developed machine learning models for predicting k_{off} values, employing eight algorithms, including Bayesian Neural Network (BNN), random forest (RF), support vector machine (SVM), and extreme gradient boosting (XGBoost) [36]. Using descriptors based on van der Waals and electrostatic interaction energies, they conducted case studies on HSP90 and RIP1 kinase inhibitors, achieving state-of-the-art accuracy with the BNN model, which obtained correlation coefficients of $R^2 = 0.947$ and $R^2 = 0.745$ for the HSP90 and RIP1 datasets, respectively. This study demonstrates the effectiveness of machine learning, particularly BNN, for k_{off} prediction, suggesting these models are well-suited for high-throughput screening.

In 2024, Akhunzada et al. proposed a machine learning approach focusing on dynamic interaction fingerprints (IFPs) along the ligand unbinding pathway [35]. Using a dataset of 85 protein-ligand complexes specific to heat shock protein (HSP90) from the PDBbind-k_{off}-2020 database, they generated dynamic IFPs via steered molecular dynamics (SMD) simulations. By incorporating these dynamic IFPs into a random forest (RF) regression model, they achieved a correlation coefficient (R^2) of 0.80, outperforming static IFP-based models with an R^2 of 0.60, thereby highlighting the value of dynamic interaction data in capturing unbinding kinetics.

Our Contribution: While previous studies have advanced the modeling of k_{off} values, challenges persist in capturing the mutual information between proteins and ligands and in identifying key atoms and residues involved in these interactions. To address these challenges, we introduce two novel approaches for predicting the ligand dissociation constant (k_{off}): a classical Graph Convolutional Network (GCN) and a quantum-enhanced GCN (QGCN). To our knowledge, this is the first application of both graph neural networks and quantum neural networks for predicting the ligand dissociation constant (k_{off}). Our models utilize spatial data from 3D structural representations of protein-ligand complexes in the PDBbind-k_{off}-2020 dataset [34], represented as graphs, to capture intricate interaction patterns. Additionally, we incorporate a GCN architecture adapted from [22], optimized using AutoDock Vina [33] distance terms, and integrate a bitransport information mechanism with physics-based distance terms to enhance prediction

accuracy. Unlike other methods, our approach not only effectively captures mutual information between protein-ligand pairs but also highlights essential atoms in the ligands and residues in the proteins. These innovations present a promising framework for advancing structure-based drug design.

III. Dataset

A. Dataset Collection and Data Mining Approach

In this study, the PDBbind-k_{off}-2020 dataset [34] was used to benchmark protein–ligand dissociation rate constant (k_{off}) prediction. This dataset is a collection of protein–ligand complexes for which experimentally measured k_{off} values are available. To prepare the dataset, we carefully filtered the protein-ligand pairs based on the following criteria: (i) all measurements had to be specific values rather than approximations or ranges; (ii) ligand structures had to match between the PDBbind-k_{off}-2020 dataset and the chemical component dictionaries in the PDB databank; (iii) proteins were required to have only one Cα atom per residue, (iv) it was possible to read and parse the files. Initially, the dataset contained 680 protein-ligand complexes, from which 423 were retained. After filtering, we split the dataset into training, validation, and test sets in 70:15:15 ratio, respectively.

3D spatial features of an individual protein and ligand were read and parsed from PDB and MOL2 files, respectively [4], [5], [18].

B. Feature Representation

1) Protein Representation:

Node Feature Matrix: In our model, protein 3D structures are represented as graphs, where nodes correspond to 20 different amino acid residues, and edges connect nonconsecutive amino acid residues whose Cα atoms are within an 8 Å distance. Each node in this graph has a 41-dimensional feature vector that combines primary sequence information, normalized position, and evolutionary relationships:
(1) *Primary sequence information*: A 20-dimensional one-hot encoding is used to represent the amino acid types in the protein sequence; (2) *Normalized position*: The position of each amino acid in the sequence is scaled by the total sequence length (number of residues in this sequence); (3) *Evolutionary relationships*: Using the BLOSUM62 matrix [37], which captures evolutionary similarities among amino acids, a 20-dimensional vector from the matrix represents each amino acid type.

Together, these features yield a 41-dimensional vector for each node in the protein graph.

Adjacency Matrix: The adjacency matrix represents connections between amino acid residues as nodes, with edges indicating non-covalent interactions between these residues. An edge is created if the Euclidean distance between the C_α atoms of two amino acids is less than 8 $\mathring{A}$. This distance map is converted into an adjacency matrix by assigning a value of 1 for edges (distances $\leq$ 8 $\mathring{A}$) and 0 otherwise.

2) Ligand Representation: In the ligand graph, nodes represent atoms, and edges denote chemical bonds, generated using RDKit [38]. The initial node features for ligands include: (1) *Atom type*: 10-dimensional one-hot encoding of atom types (H, C, N, O, F, P, S, Cl, Br, I), only these 10 atoms appear in the whole dataset; (2) *Atom degree*: 5-dimensional one-hot encoding based on atom connectivity; (3) *Atom explicit valence*: 7-dimensional one-hot encoding for the explicit valence of each atom; (4) *Atom implicit valence*: 2-dimensional one-hot encoding for the implicit valence; (5) *Aromaticity*: A 2-dimensional feature indicating whether the atom is aromatic.

These features result in an 26-dimensional vector per atom. For edges, a 6-dimensional feature vector represents six bond types, including single, double, triple, aromatic bonds, as well as conjugated and ring bonds.

3) Interaction Representation: An interaction-based subgraph captures the pocket-ligand interaction features. Nodes in this subgraph include protein amino acid residues and ligand atoms: - Amino acid residues nodes: Each node has a 41-dimensional feature vector, as defined in the protein representation. - Ligand atom nodes: Each node is represented by the 26-dimensional feature vector described in the ligand representation.

Edges in this subgraph are created if the Euclidean distance between a heavy atom from the ligand and an amino acid Cα atom is less than 8 Å. The edge feature is defined as $\text{edge}_{\text{inter}} = \frac{8\ \text{Å} - \text{distance}}{8\ \text{Å}}$ for distances under 8 Å, and is set to 0 otherwise.

C. Vina Distance Optimization Terms

In our study, the Vina distance optimization terms incorporate the same weighted scoring terms as used in AutoDock Vina [33]. These terms are composed of five conformation-dependent terms and one conformation-independent term. The five conformation-dependent terms represent intermolecular contributions, including three terms for steric interactions, one term for hydrophobic bonding, and one for hydrogen bonding. The detailed setup of Vina terms is available in Appendix B.

All code implementation were done on Python programming language. PyTorch deep learning library [41], and Deep Graph Library (DGL) [42] were used to construct graphs, and train the Graph Convolutional Networks (GCN). PennyLane [44] and Qiskit [40] quantum machine learning libraries were used to construct quantum neural networks [43].

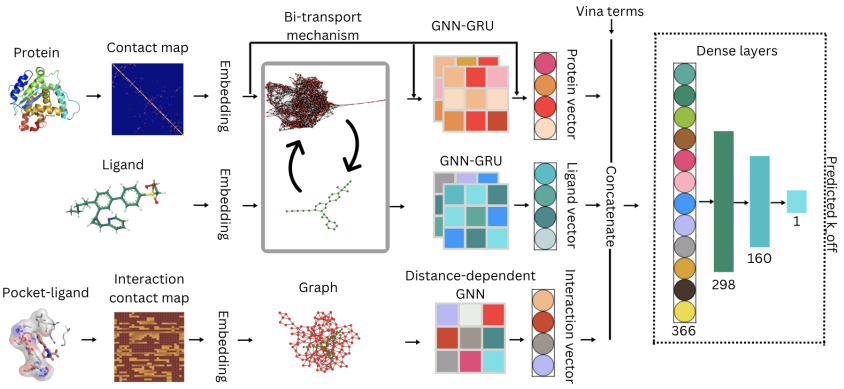

Fig. 2. Three main GNN blocks were developed to extract features from the protein structure, ligand structure, and pocket-ligand interaction structure, as proposed in GraphscoreDTA [22]. In the protein and ligand blocks, a bitransport information mechanism was introduced to facilitate communication between the protein and ligand representations. Separate GNNs were then employed to capture structural features of the protein and ligand, integrating multihead attention, GRUs, and skip connections within the networks. For the pocket-ligand interaction block, distance-dependent GNNs were used to model the pocket-ligand structure. Finally, the features learned from all three blocks, along with Vina distance optimization terms, were combined and passed through three dense layers to predict binding affinity.

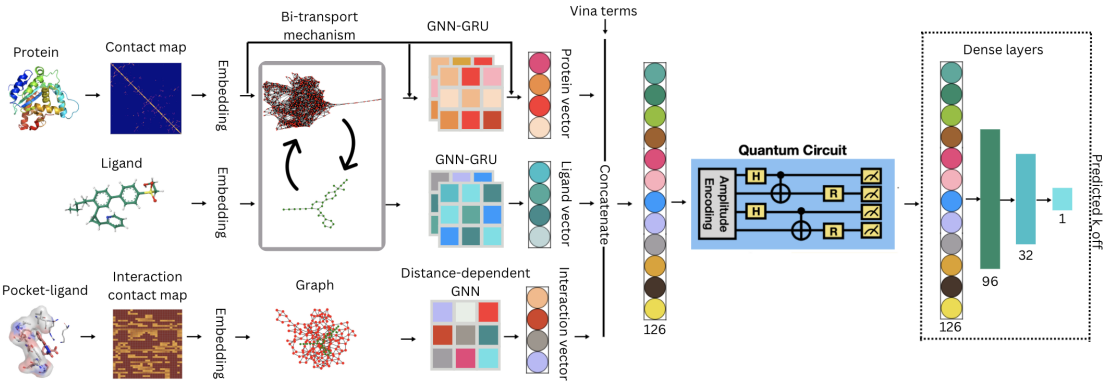

Fig. 3. The Quantum GCN architecture. The output vector of size 126 is trained on the Variational Quantum Circuit and followed by classical post-processing step: fully connected dense layers. The quantum circuit in this figure is simplified for demonstration purposes. The complete variation quantum circuits are provided in Fig. 15 and Fig. 16.

IV. Quantum Data Encoding

In quantum computing, data from classical systems must be encoded into quantum states. This section discusses two encoding schemes used in our work: *Amplitude Encoding* and *Variational Encoding*. We demonstrate each encoding with an example. Note: the basics of quantum computing are covered in Appendix A.

A. Amplitude Encoding

Amplitude encoding maps a classical data vector into the amplitudes of a quantum state. The visual representation of the quantum circuit for amplitude encoding is provided in Fig. 15. Given a normalized classical vector $\mathbf{x} = [x_0, x_1, \ldots, x_{N-1}]$, it is encoded as a quantum state:

$$|\Psi\rangle = \sum_{i=0}^{N-1} \alpha_i |i\rangle, \quad \text{where } \alpha_i = \frac{x_i}{\|\mathbf{x}\|}, \ \|\mathbf{x}\| = \sqrt{\sum_{i=0}^{N-1} x_i^2}.$$

This encoding reduces the number of qubits needed for large datasets, as an N-dimensional vector requires only $\log_2(N)$ qubits.

B. Variational Encoding

Variational encoding maps classical data into quantum states by using the values in the data array as

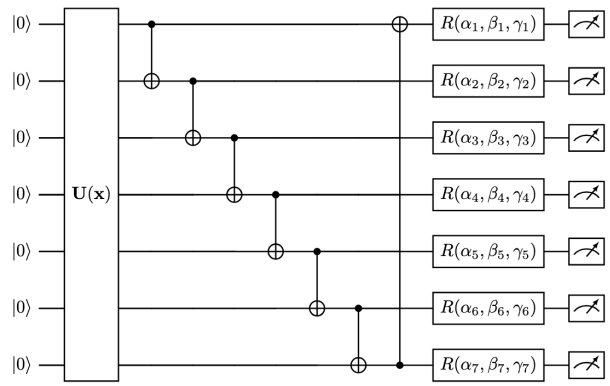

Fig. 4. Variational quantum circuit architecture for the classifier with amplitude encoding. The initial VQC block encodes the vector produced by the graph convolution operation. An N-dimensional vector is transformed into a quantum state using $\log_2(N)$ qubits. In this study, $N = 10$. The $U(\mathbf{x})$ function represents the quantum routine for amplitude encoding. Trainable parameters, denoted by α_i, β_i, and γ_i, are optimized during training.

rotation angles for quantum gates, see Fig. 16 for the visual representation of the quantum circuit. For a given classical vector $\mathbf{x} = [x_1, x_2, \ldots, x_n]$, the encoding uses parameterized single-qubit gates, such as R_y and R_z, to prepare the quantum state:

$$U(\mathbf{x}) = R_z(\arctan(x_1^2))R_y(\arctan(x_1)) \otimes \cdots \otimes R_z(\arctan(x_n^2))R_y(\arctan(x_n)).$$

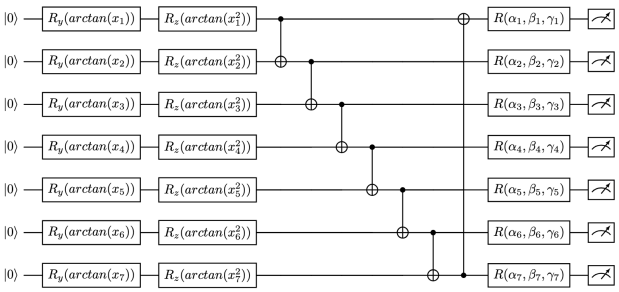

Fig. 5. Variational quantum circuit block with variational encoding. This encoding block employs R_y and R_z rotation gates, where the rotation angles are parameterized by $\arctan(x_i)$ and $\arctan(x_i^2)$ for each qubit. The parameters α_i, β_i, and γ_i are trainable and optimized during the learning process.

V. Model Construction

In this work, we employed (1) a graph convolutional neural network (GCN) architecture from [22] incorporating Vina distance optimization terms to predict protein-ligand dissociation constant (k_{off}), (2) a concatenation of two GCNs with described architecture processed by a Gated Recurrent Unit (GRU) layer and trained on protein-ligand structures before and after 2-ns Molecular Dynamics (MD) simulation, (3) a Quantum GCN with reduced number of parameters and augmented by Quantum Layer - Variation Quantum Circuit.

A. GCN Architecture

As illustrated in Fig. 2, the architecture begins with embedding layers that convert node and edge vectors into fixed-dimensional representations. The model is composed of three primary GNN blocks designed to learn features from protein structures, ligand graphs, and pocket-ligand interaction structures, respectively. The details of the GNN setup are as follows.

First, a bitransport information mechanism was introduced to facilitate information exchange between the protein and ligand representations. Then, the amino acid and atom features were processed through two iterations of protein and compound GNNs, producing updated feature representations. In the second iteration, a skip connection was added between the embedding vector and the GNN input, followed by another skip connection linking the embedding vector to the GNN output. In the interaction graph, the amino acid and atom features were processed by a single GNN layer to update the interaction features, and a skip connection was applied between the embedding vector and the GNN output.

Next, a multihead attention mechanism was incorporated into both the protein and ligand GNNs to adjust the importance of individual atoms within the ligand and specific residues within the protein. Additionally, gated recurrent units (GRUs) were employed to control the information flow between the general node and the super node within the ligand and protein graphs. To enhance predictive performance, Vina distance terms, capturing intermolecular interactions and ligand flexibility, were included in the model. As discussed in III-C, these terms were derived from AutoDock Vina, a widely used tool for molecular docking and virtual screening, and include three steric interaction terms (gauss1, gauss2, and repulsion), a hydrophobic interaction term, and a hydrogen bond term. Ligand flexibility is represented by the number of rotatable bonds among the ligand's heavy atoms.

Finally, the features generated by the three GNN blocks, along with the Vina distance optimization terms, were concatenated and passed through three dense layers. The first two dense layers consist of 390 and 320 nodes, each followed by a PReLU activation function, while the third layer has 160 nodes and leads to the output. To fine-tune the model, the validation set was

used for parameter tuning, and the model was trained for 100 epochs. The version of the model achieving the best validation performance was selected. The chosen hyperparameters are detailed in Table I.

1) Bitransport information mechanism: In this study, we utilized a bitransport information mechanism (introduced in [22], here adapted by us for k_{off} prediction) for information exchange between protein and ligand feature representations. This mechanism relies on multihead attention and position-aware attention to capture the distinct contributions of each component.

For the ligand feature updates, multihead attention was applied to transform the protein feature matrix $X_p \in \mathbb{R}^{n\times c}$ into a unified global descriptor $G_p \in \mathbb{R}^{1\times c}$. Here, we used $h = 8$ attention heads to aggregate information effectively. The global representation G_p for the protein is defined as:

$$G_p = w_g \left(\frac{1}{h} \sum_i g_i \right) \in \mathbb{R}^{1\times c},$$

where w_g is a learnable parameter, c represents the feature dimension, and g_i denotes each attention head output, defined by:

$$g_i = Te_i \in \mathbb{R}^{s\times 1},$$

with T derived by applying a transformation on X_p as follows:

$$T = w_p X_p = [t_1, t_2, \ldots, t_n] \in \mathbb{R}^{s\times n},$$

where w_p is another learnable parameter. Additionally, multihead attention employs a matrix B, constructed by:

$$B = w_t X_p = [b_1^T, b_2^T, \ldots, b_n^T]^T \in \mathbb{R}^{h\times n},$$

where each attention score e_j^i for position j is given by:

$$e_j^i = \frac{\exp(b_j^i)}{\sum_s \exp(b_s^i)},$$

where s represents the feature dimension, and n is the total number of amino acids in the protein.

Position-aware attention then updates each ligand atom's features. The initial ligand features $X_l \in \mathbb{R}^{m\times c}$ are modified as follows:

$$X_l^1 = X_l + X_l \odot a_l^T.$$

Here, position-aware attention for ligand features is computed by:

$$a_l = \text{softmax}\left((w_l X_l)^T w_g G_p \right),$$

where w_l and w_g are learnable parameters, and m is the number of ligand atoms. This approach is repeated in the protein to get updated amino acid residue features.

2) Multihead Attention Mechanism: In the ligand atom graph, in addition to the standard atom nodes, a virtual "super node" was introduced to gather information from the actual atom nodes. This multihead attention mechanism was applied to assess the contribution of each individual atom node. For transferring information from general atom nodes to the super node, a K-head attention mechanism was implemented as follows:

$$I_{v2s} = F\left(w_{a2s} \left[\sum_{i=1}^{n} T_{a,i}^1 V_i, \sum_{i=1}^{n} T_{a,i}^2 V_i, \ldots, \sum_{i=1}^{n} T_{a,i}^K V_i \right] \right)$$

where F denotes the $\tanh$ activation function, n is the number of ligand atoms, and w_{a2s} represents a learnable weight. In this implementation, $K = 4$ attention heads were used.

Each $T_{a,i}^k$ is computed using a softmax function, which assigns weights based on the specific attention parameters:

$$T_{a,i}^k = \text{softmax}\left(w_{at}^k c_{v,i}^k \right), \quad k = 1, 2, 3,$$

where i represents the index of ligand atoms, and w_{at}^k is a learnable parameter.

The values $c_{v,i}^k$ are calculated by combining features from the atom nodes and the super node using the following formula:

$$c_{v,i}^k = F\left(w_{aat}^k V_i \right) * F\left(w_{sat}^k s \right), \quad k = 1, 2, 3,$$

where w_{aat}^k and w_{sat}^k are learnable weights, V_i represents the atom node features, and s represents the features of the super node.

This process is similarly applied in the protein graph, where information is transferred and updated using the same multihead attention approach.

Here is a paraphrased version of the text with mathematical notation included:

3) Gated Recurrent Units: To model dependencies across different time scales, a Gated Recurrent Unit (GRU) [45] was incorporated into the architecture. This GRU mechanism helps capture the temporal dependencies among nodes in both individual atom and super nodes. Two GRUs were employed to manage the proportion of information shared between individual atoms and the super node.

The feature update for the super node can be defined as:

$$s' = \text{GRU}(s, \text{tran}_{v2s}),$$

where tran_{v2s} is the transformed information from atom nodes to the super node, computed as:

$$\text{tran}_{v2s} = g_s * I_{v2s} + (1 - g_s) * I_{s2s},$$

with g_s being the gating mechanism determined by:

$$g_s = \text{sigmoid}(w_{s2s} I_{s2s} + w_{v2s} I_{v2s}),$$

where w_{s2s} and w_{v2s} are learnable parameters.

The update of features for each individual atom node V_i is calculated as:

$$V_i' = \text{GRU}(V_i, \text{tran}_{s2v}),$$

where tran_{s2v} is the information transferred from the super node back to the individual atom nodes, defined by:

$$\text{tran}_{s2v} = g_v * I_{s2v} + (1 - g_v) * I_{v2v},$$

and the gating mechanism g_v is calculated as:

$$g_v = \text{sigmoid}(w_{v2v} I_{v2v} + w_{s2v} I_{s2v}),$$

where w_{v2v} and w_{s2v} are learnable weights.

In this setup, tran_{v2s} denotes the transformed information moving from atom nodes to the super node, while tran_{s2v} represents the flow of processed information from the super node back to the atom nodes. The terms I_{v2s}, I_{s2s}, I_{v2v}, and I_{s2v} refer to different types of interaction information between nodes.

4) Graph Neural Network: The graph neural network (GNN) was developed to capture the ligand graph features, incorporating two types of nodes—the individual atom nodes and a virtual super node—as well as a single type of edge encoding. Five distinct types of information were utilized to compute and update the node features within this graph: (i) information from neighboring atoms; (ii) data from the super node; (iii) transformed data transferred from individual atoms to the super node; (iv) transformed information passed from the super node back to individual atoms; and (v) edge-related information. The update process for the super node features was implemented using the $\tanh$ activation function (see Fig. 6-A).

To transfer information from individual atoms to the super node, a multihead attention mechanism was employed (as shown Fig. 7). Similarly, information from the super node back to individual atoms was updated using the $\tanh$ activation function (see Fig. 6-B). The aggregation of information from neighboring atoms was achieved through one-step neighbor information collection along graph edges (see Fig. 8).

To refine the feature representations, two GRUs were applied separately to update features of individual atoms and the super node. This GNN structure iterated twice to update the ligand node features fully. The protein graph features were updated through a similar GNN process, differing only in that edge features were excluded from the graph updates.

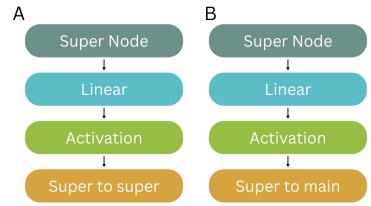

Fig. 6. Information transfer of super node

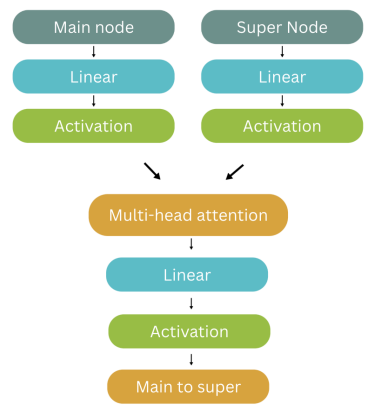

Fig. 7. Information transfer from main node to super node

5) Pocket–Ligand Distance-Dependent Graph Neural Network: In the pocket-ligand interaction graph, a radial pooling mechanism based on distance was employed to capture the interaction edges. Each interaction edge is defined as:

$$f(r_{ij}) = \left(\exp\left(\frac{-(r_{ij} - r_s)^2}{\sigma^2} \right) \right) f_c(r_{ij}),$$

where r_{ij} represents the interaction distance between a protein residue and a ligand atom, and r_s and σ are learnable parameters. The term $f_c(r_{ij})$ is a cutoff function, defined as:

$$f_c(r_{ij}) = \begin{cases} \frac{1}{2} \cos\left(\frac{\pi r_{ij}}{R_C} \right), & 0 < r_{ij} \leq 8 \text{ Å}, \\ 0, & r_{ij} > 8 \text{ Å}, \end{cases}$$

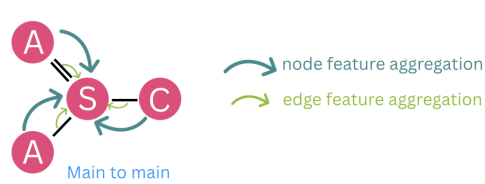

Fig. 8. Atom information gathered from its neighbor atoms

TABLE I. Hyperparameters for Different Model Blocks

Blocks	**Hyperparameters**	**Settings for GCN**	**Settings of Quantum GCN**
	Epoch	80	50
	Batch size	8	4
	Optimizer	AdamW	AdamW, RMSProp
	Learning rate	1.2e-4	1.2e-3
	Weight decay	1e-6	1e-5
Ligand GNN block	Iteration	2	2
	Graph node feature size	120	40
	Hidden size (node)	120	40
	Graph edge feature size	8	6
	Attention heads	4	4
	Activation function	LeakyReLU, Tanh, Sigmoid	LeakyReLU, Tanh, Sigmoid
Protein GNN block	Iteration	2	2
	Graph node feature size	120	40
	Hidden size (node)	120	40
	Attention heads	4	4
	Activation function	LeakyReLU, Tanh, Sigmoid	LeakyReLU, Tanh, Sigmoid
Pocket-ligand interaction distance-based GNN block	Iteration	1	1
	Graph node feature size	120	40
	Distance cutoff	8 Å	8 Å
	Depth	3	3
	Activation function	ReLU	ReLU
Fully connected neural network block	Number of neurons	[366, 298, 160]	[126, 96, 32]
	Activation function	PReLU	PReLU

where R_C is set to 8 Å.

Following this, the graph node features are updated by aggregating information from neighboring nodes along different edges, as well as from self-node contributions. Ultimately, three iterations of distance-based radial pooling within the pocket-ligand interaction graph neural network are performed to enhance and update the node features.

B. 2-GCN-GRU Architecture

In order to capture the temporal relationship between protein-ligand complexes we applied the described GCN to 2 datasets, where both of them have the same protein-ligand complexes but have different 3D structures. The first dataset captures the initial structures of complexes (before the chemical reaction). The second dataset contains the structures of the same complexes after 2-ns Molecular Dynamics (MD) simulation (after the chemical reaction).

Both of these datasets were passed through two GCNs with same architecuture but distinct parameters. Their outputs of size 126 and 126 were concatenated into an array of size 252 and fed into a new Gated Recurrent Unit (GRU) layer. The introduced GRU layer is expected to capture the temporal relationship between the 2 timesteps of the complexes.

The output of the GRU is processed in a similar manner by passing through fully connected layers.

C. Quantum GCN Architecture

The proposed Quantum GCN framework consists of three main components: (1) the described Graph Convolutional Network (GCN), (2) a Variational Quantum Circuit (VQC), and (3) a classical post-processing stage, see Fig 3. The idea of introducing the quantum layers - VQCs - is to reduce the number of parameters in the GCN, while preserving its accuracy using quantum advantage. Thus, we were able to reduce the complexity of the graph neural network, which had input size of 366, into a much smaller neural network of size 126. Refer to Table I for the details of GCN compression.

Initially, the original complex structures are processed through the reduced GCN, which outputs a feature vector of size 126. Then, the vector of size 126 is then padded with two additional zeros to reach a size of 128, aligning it with the requirement of 2^n for amplitude encoding.

Next, the padded vector is encoded into a quantum state using amplitude encoding (as described in Section A-A) to minimize the number of qubits required. The resulting quantum state is subsequently processed through a series of variational operations. Specifically, the architecture includes two VQCs separated by a tanh activation function. The first VQC performs both the amplitude encoding and variational transformations, Fig. 15. The Pauli-Z (Z-gate) expectation values from the first VQC are then re-encoded using the variational encoding scheme and passed to the second VQC for

further variational transformations, Fig. 16.

Finally, the measured expectation values from the second VQC block are fed into a fully connected neural network, which processes the data to output the predicted k_{off} value.

1) Optimization of Quantum Circuits: For gradient-based optimization, the parameter-shift rule [46], [47] is employed to compute the gradients of quantum functions. This approach has been highly effective in VQC-based quantum machine learning tasks [48]–[54]. For a quantum function $f(x;\theta)$ with an observable $\hat{P}$, the gradient is calculated as:

$$\begin{aligned} f(x;\theta) &= \left\langle 0 \middle| U_0^\dagger(x) U_i^\dagger(\theta_i) \hat{P} U_i(\theta_i) U_0(x) \middle| 0 \right\rangle \\ &= \left\langle x \middle| U_i^\dagger(\theta_i) \hat{P} U_i(\theta_i) \middle| x \right\rangle, \end{aligned}$$

where x is the classical input from the GCN, $U_0(x)$ prepares the quantum state based on the classical data, $U_i(\theta_i)$ represents the single-qubit rotation generated by the Pauli operators X, Y, and Z, and i is the index of the circuit parameter being evaluated. The gradient of the function f with respect to the parameter θ_i is given by:

$$\frac{\partial f(x;\theta)}{\partial \theta_i} = \frac{1}{2}\left[f\left(x;\theta_i + \frac{\pi}{2}\right) - f\left(x;\theta_i - \frac{\pi}{2}\right) \right].$$

With the ability to calculate quantum gradients, various classical optimization algorithms can be applied. For this study, the RMSProp optimizer is used, which is a variant of gradient descent with an adaptive learning rate. The circuit parameters θ are updated as follows:

$$\begin{aligned} E[g^2]_t &= \alpha E[g^2]_{t-1} + (1-\alpha) g_t^2, \\ \theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t, \end{aligned}$$

where g_t is the gradient at step t and $E[g^2]_t$ is the weighted moving average of squared gradients, with the initial value $E[g^2]_0 = g_0^2$. The hyperparameters are set as follows: learning rate $\eta = 0.01$, smoothing constant $\alpha = 0.99$, and $\epsilon = 10^{-8}$.

VI. Evaluation

A. Evaluation Metrics

We used the training and validation loss graphs to show the convergence of our models. Most of the approaches converge at epochs 40-50. For the demonstration of Quantum Advantage, we compared the loss of Quantum GCN which used the reduced number of parameters, with a similar reduced analog without the quantum layers. It is clearly seen that QGCN has lower loss (approx. 0.0023) compared to the reduced GCN with loss (approx. 0.0047)

We used the following metrics to assess the predictive accuracy of our models: root mean squared error

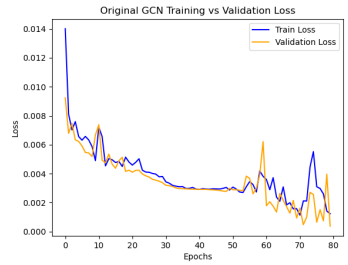

Fig. 9. Original GCN Training vs Validation Loss

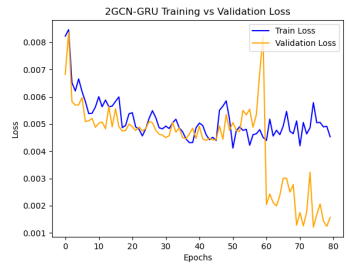

Fig. 10. 2GCN-GRU Training vs Validation Loss

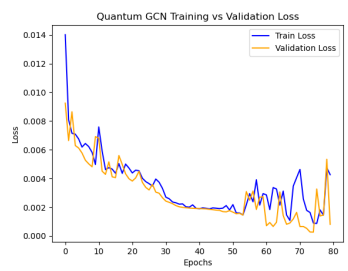

Fig. 11. Quantum GCN Training vs Validation Loss

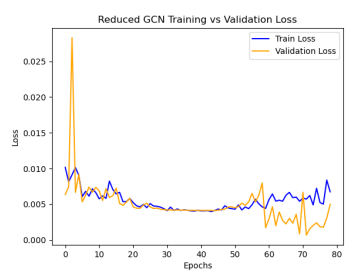

Fig. 12. Reduced GCN Training vs Validation Loss

(RMSE) and mean absolute error (MAE). The formulas for each metric are as follows:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=0}^{n-1}(y_i - \hat{y}_i)^2}; \quad \text{MAE} = \frac{1}{n}\sum_{i=0}^{n-1}|y_i - \hat{y}_i|$$

where $\hat{y}_i$ is the predicted k_{off} value for the i-th sample, y_i is the actual k_{off} value, n represents the number of samples.

According to Table II the original GCN and Quantum GCN have comparable and even higher MAE and RMSE metrics for prediction k_{off} values. This indicates the supremacy of the Graph Neural Networks for spacial representation of proteins and ligand. In addition, the Quantum GCN even with lower number of parameters is approximately as precise as the original GCN initially proposed in [22], but upgraded to be used for k_{off} predition.

TABLE II. Performance comparison of various models based on RMSE and MAE. Bold values highlight the best performance in each metric.

Author	**RMSE**	**MAE**
GCN	**0.242**	**0.114**
GCN + 2 timestep GRU	0.313	0.191
Quantum GCN	**0.276**	**0.115**
Su et al.	0.85	–
Amangeldiuly et al.	0.82	0.59
Zhao et al.	**0.220**	0.184
Akhunzada et al.	0.38	0.29

Although the Quantum GCN achieves comparable accuracy to the full classical GCN while using fewer parameters, our findings also highlight model-specific limitations. Despite the implementatoin of temporal structure, the 2GCN-GRU model, showed higher RMSE and MAE, suggesting that two timesteps of MD (0 ns and 2 ns) may not contain sufficiently informative structural changes for k_{off} prediction. The Quantum GCN, on the other hand, generalizes nearly as well as the high-capacity classical baseline, but still slightly underperforms the model of Zhao et al. in RMSE. This indicates that while quantum compression effectively reduces model size, further improvements in encoding schemes, circuit depth, and hybrid architectures may be needed to consistently outperform classical counterparts. Overall, these results provide evidence that spatial GNN representations are the dominant factor in k_{off} prediction, and that quantum layers can be integrated without degrading predictive accuracy.

VII. Conclusion and Future Discussion

In this work, we introduced three architectures for predicting protein–ligand dissociation constants (k_{off}): a classical GCN model, a temporal 2GCN-GRU model, and a Quantum GCN that incorporates Variational Quantum Circuits to reduce the number of trainable parameters. Across the full PDBbind-koff-2020 benchmark, our results demonstrate that spatial GNNs remain highly effective for modeling protein–ligand interactions, outperforming previous QSKR approaches by a large margin. The Quantum GCN matches the accuracy of the full classical model while using a substantially smaller parameter set, highlighting the promise of quantum-based compression in structure-based learning.

Looking forward, several research directions emerge. First, temporal models may benefit from longer or better-selected MD trajectories, as two timesteps appear insufficient to capture relevant unbinding dynamics. Second, improving quantum encodings—such as data-reuploading, entangling layers, or QSVT-inspired blocks—may further enhance accuracy and robustness. Third, integrating physically motivated features such as path sampling metrics, interaction energies, or dynamic fingerprints could strengthen the model's ability to capture long-timescale kinetic effects. Finally, scaling these models to larger datasets and benchmarking them on prospective, experimentally measured k_{off} values will be essential for assessing their translational impact in drug discovery.

Code and Data Availability

The code used in this study is publicly available at https://github.com/asalamatov/Koff_Q_GCN. The dataset used in this study is also publicly available from the PDBbind database and can be accessed at https://www.pdbbind-plus.org.cn/.

Conflict of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

References

[1] Tonge, Peter J. "Drug–target kinetics in drug discovery." ACS chemical neuroscience 9.1 (2018): 29-39.

[2] Schuetz, Doris A., et al. "Kinetics for Drug Discovery: an industry-driven effort to target drug residence time." Drug discovery today 22.6 (2017): 896-911.

[3] IJzerman, Adriaan P., and Dong Guo. "Drug–target association kinetics in drug discovery." Trends in biochemical sciences 44.10 (2019): 861-871.

[4] Jakub Poziemski, Artur Yurkevych, Paweł Siedlecki, "Assessment of molecular dynamics time series descriptors in protein-ligand affinity prediction," unpublished

[5] Sohraby, Farzin, and Ariane Nunes-Alves. "Advances in computational methods for ligand binding kinetics." Trends in Biochemical Sciences 48.5 (2023): 437-449.

[6] Ray, Dhiman, Sharon Emily Stone, and Ioan Andricioaei. "Markovian weighted ensemble milestoning (M-WEM): Long-time kinetics from short trajectories." Journal of Chemical Theory and Computation 18.1 (2021): 79-95.

[7] Wolf, Steffen, et al. "Multisecond ligand dissociation dynamics from atomistic simulations." Nature communications 11.1 (2020): 2918.

[8] Brotzakis, Z. Faidon, Vittorio Limongelli, and Michele Parrinello. "Accelerating the calculation of protein–ligand binding free energy and residence times using dynamically optimized collective variables." Journal of chemical theory and computation 15.1 (2018): 743-750.

[9] Jagger, Benjamin R., Anupam A. Ojha, and Rommie E. Amaro. "Predicting ligand binding kinetics using a Markovian milestoning with voronoi tessellations multiscale approach." Journal of Chemical Theory and Computation 16.8 (2020): 5348-5357.

[10] Votapka, Lane W., et al. "SEEKR2: Versatile multiscale milestoning utilizing the OpenMM molecular dynamics engine." Journal of chemical information and modeling 62.13 (2022): 3253-3262.

[11] Miao, Yinglong, Apurba Bhattarai, and Jinan Wang. "Ligand Gaussian accelerated molecular dynamics (LiGaMD): Characterization of ligand binding thermodynamics and kinetics." Journal of chemical theory and computation 16.9 (2020): 5526-5547.

[12] Dickson, Alex, and Charles L. Brooks III. "WExplore: hierarchical exploration of high-dimensional spaces using the weighted ensemble algorithm." The Journal of Physical Chemistry B 118.13 (2014): 3532-3542.

[13] Donyapour, Nazanin, Nicole M. Roussey, and Alex Dickson. "REVO: Resampling of ensembles by variation optimization." The Journal of chemical physics 150.24 (2019).

[14] Dandekar, Bhupendra R., and Jagannath Mondal. "Capturing protein–ligand recognition pathways in coarse-grained simulation." The Journal of Physical Chemistry Letters 11.13 (2020): 5302-5311.

[15] Schütt, Kristof, et al. "Schnet: A continuous-filter convolutional neural network for modeling quantum interactions." Advances in neural information processing systems 30 (2017).

[16] Thölke, Philipp, and Gianni De Fabritiis. "Torchmd-net: equivariant transformers for neural network based molecular potentials." arXiv preprint arXiv:2202.02541 (2022).

[17] Simeon, Guillem, and Gianni De Fabritiis. "Tensornet: Cartesian tensor representations for efficient learning of molecular potentials." Advances in Neural Information Processing Systems 36 (2024).

[18] Schapin, Nikolai, et al. "On Machine Learning Approaches for Protein-Ligand Binding Affinity Prediction." arXiv preprint arXiv:2407.19073 (2024).

[19] Dong, Lina, et al. "Prediction of Protein-Ligand Binding Affinity by a Hybrid Quantum-Classical Deep Learning Algorithm." Advanced Quantum Technologies 6.9 (2023): 2300107.

[20] Domingo, L., et al. "A hybrid quantum-classical fusion neural network to improve protein-ligand binding affinity predictions for drug discovery." arXiv preprint arXiv:2309.03919 (2023).

[21] Domingo, Laia, et al. "Hybrid quantum-classical convolutional neural networks to improve molecular protein binding affinity predictions." arXiv preprint arXiv:2301.06331 (2023).

[22] Wang, K., Zhou, R., Tang, J., Li, M. (2023). GraphscoreDTA: optimized graph neural network for protein–ligand binding affinity prediction. Bioinformatics, 39(6), btad340.

[23] Copeland, Robert A., David L. Pompliano, and Thomas D. Meek. "Drug–target residence time and its implications for lead optimization." Nature reviews Drug discovery 5.9 (2006): 730-739.

[24] Tummino, Peter J., and Robert A. Copeland. "Residence time of receptor- ligand complexes and its effect on biological function." Biochemistry 47, no. 20 (2008): 5481-5492.

[25] Schuetz, Doris A., et al. "Kinetics for Drug Discovery: an industry-driven effort to target drug residence time." Drug discovery today 22.6 (2017): 896-911.

[26] Pan, Albert C., David W. Borhani, Ron O. Dror, and David E. Shaw. "Molecular determinants of drug–receptor binding kinetics." Drug discovery today 18, no. 13-14 (2013): 667-673.

[27] Su, Minyi, Huisi Liu, Haixia Lin, and Renxiao Wang. "Machine-Learning Model for Predicting the Rate Constant of Protein-Ligand Dissociation." Acta Physico-Chimica Sinica 36, no. 1 (2020): 1907006-0.

[28] Guo, Dong, Thea Mulder-Krieger, Adriaan P. IJzerman, and Laura H. Heitman. "Functional efficacy of adenosine A2A receptor agonists is positively correlated to their receptor residence time." British journal of pharmacology 166, no. 6 (2012): 1846-1859.

[29] Folmer, Rutger HA. "Drug target residence time: a misleading concept." Drug Discovery Today 23, no. 1 (2018): 12-16.

[30] Bruce, Neil J., Gaurav K. Ganotra, Daria B. Kokh, S. Kashif Sadiq, and Rebecca C. Wade. "New approaches for computing ligand–receptor binding kinetics." Current opinion in structural biology 49 (2018): 1-10.

[31] Qu, Sujun, Shuheng Huang, Xianchao Pan, Li Yang, and Hu Mei. "Constructing interconsistent, reasonable, and predictive models for both the kinetic and thermodynamic properties of HIV-1

protease inhibitors." Journal of Chemical Information and Modeling 56, no. 10 (2016): 2061-2068.

[32] Ganotra, Gaurav K., and Rebecca C. Wade. "Prediction of drug–target binding kinetics by comparative binding energy analysis." ACS medicinal chemistry letters 9, no. 11 (2018): 1134-1139.

[33] Trott, Oleg, and Arthur J. Olson. "AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading." Journal of computational chemistry 31, no. 2 (2010): 455-461.

[34] Liu, Huisi, Minyi Su, Hai-Xia Lin, Renxiao Wang, and Yan Li. "Public data set of protein–ligand dissociation kinetic constants for quantitative structure–kinetics relationship studies." ACS omega 7, no. 22 (2022): 18985-18996.

[35] Akhunzada, Muhammad Jan, Hyun Jung Yoon, Abdennour Braka, Indrajit Deb, and Sangwook Wu. "Using Dynamic Interaction Fingerprints to Derive Baseline Machine Learning Model for the Prediction of Protein-Ligand Dissociation Rate Constant." (2024).

[36] Zhao, Yujing, Qilei Liu, Jian Du, Qingwei Meng, and Lei Zhang. "Machine learning methods for developments of binding kinetic models in predicting protein-ligand dissociation rate constants." Smart Molecules 1, no. 3 (2023): e20230012.

[37] Henikoff, Steven, and Jorja G. Henikoff. "Amino acid substitution matrices from protein blocks." Proceedings of the National Academy of Sciences 89, no. 22 (1992): 10915-10919.

[38] Landrum, Greg. "Rdkit documentation." Release 1, no. 1-79 (2013): 4.

[39] Bondi, A. van. "van der Waals Volumes and Radii." The Journal of physical chemistry 68, no. 3 (1964): 441-451.

[40] Javadi-Abhari, Ali, Matthew Treinish, Kevin Krsulich, Christopher J. Wood, Jake Lishman, Julien Gacon, Simon Martiel et al. "Quantum computing with Qiskit." arXiv preprint arXiv:2405.08810 (2024).

[41] Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen et al. "Pytorch: An imperative style, high-performance deep learning library." Advances in neural information processing systems 32 (2019).

[42] Wang, Minjie, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou et al. "Deep graph library: A graph-centric, highly-performant package for graph neural networks." arXiv preprint arXiv:1909.01315 (2019).

[43] Domingo, Laia, Marko Djukic, Christine Johnson, and Florentino Borondo. "Binding affinity predictions with hybrid quantum-classical convolutional neural networks." Scientific Reports 13, no. 1 (2023): 17951.

[44] Bergholm, Ville, J. Izaac, M. Schuld, C. Gogolin, S. Ahmed, V. Ajith, M. S. Alam et al. "Pennylane: Automatic differentiation of hybrid quantum-classical computations. arXiv 2018." arXiv preprint arXiv:1811.04968 (2018).

[45] Chung, Junyoung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. "Empirical evaluation of gated recurrent neural networks on sequence modeling." arXiv preprint arXiv:1412.3555 (2014).

[46] Schuld, Maria, et al. "Evaluating analytic gradients on quantum hardware." Physical Review A 99.3 (2019): 032331.

[47] Bergholm, Ville, et al. "Pennylane: Automatic differentiation of hybrid quantum-classical computations." arXiv preprint arXiv:1811.04968 (2018).

[48] Mari, Andrea, Thomas R. Bromley, Josh Izaac, Maria Schuld, and Nathan Killoran. "Transfer learning in hybrid classical-quantum neural networks." Quantum 4 (2020): 340.

[49] S. Y.-C. Chen, C.-M. Huang, C.-W. Hsing, and Y.-J. Kao, "Hybrid quantum-classical classifier based on tensor network and variational quantum circuit," arXiv preprint arXiv:2011.14651, 2020

[50] S. Y.-C. Chen, S. Yoo, and Y.-L. L. Fang, "Quantum long short-term memory," arXiv preprint arXiv:2009.01783, 2020.

[51] S. Y.-C. Chen, T.-C. Wei, C. Zhang, H. Yu, and S. Yoo, "Quantum convolutional neural networks for high energy physics data analysis," arXiv preprint arXiv:2012.12177, 2020.

[52] S. Y.-C. Chen, C.-H. H. Yang, J. Qi, P.-Y. Chen, X. Ma, and H.-S. Goan, "Variational quantum circuits for deep reinforcement learning," IEEE Access, vol. 8, pp. 141007–141024, 2020.

[53] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, "Quantum circuit learning," Physical Review A, vol. 98, no. 3, p. 032309, 2018

[54] M. Schuld and F. Petruccione, "Information encoding," in Supervised Learning with Quantum Computers, pp. 139–171, Cham: Springer International Publishing, 2018.