IJHSR
ijhighschoolresearch.org
■ RESEARCH ARTICLE

Comparison of Classical and Hybrid Quantum Models in Secondary Structure Protein Prediction

Quaid Shubin

Windward School, 11350 Palms Blvd, Los Angeles, CA 90066, U.S.A; quaid.shubin@gmail.com

ABSTRACT: Protein structure prediction (PSP) is one of the most challenging and vital problems in modern biochemistry, with far-reaching implications for biological research and medical applications, particularly in drug design and disease treatment. This paper introduces how PSP has been tackled in recent decades, from wet lab experiments to new models like AlphaFold, as well as their computational limitations. We then present our perspective on how quantum computing can address the computational challenges in PSP, and investigate classical and quantum approaches for solving a subset of the PSP problem: a binary secondary structure classifier for protein primary sequences. We evaluate traditional feedforward neural networks against quantum neural network hybrid models using the CATH dataset. We found that the quantum-classical hybrid model outperformed the classical model on a 5-trial average, achieving 84.5% accuracy versus 79.75% with 13 input features. These results highlight the significant challenges in transitioning to tertiary structure predictions, but also the potential and need for further research in the integration of quantum methods to obtain more accurate predictions in larger, more diverse protein datasets.

KEYWORDS: Biochemistry, Structural Biochemistry, Quantum Computing, Quantum Neural Networks, Machine Learning, Protein Structure Prediction.

■ Introduction

The self-assembly of cells and structures within biological organisms is one of the most studied aspects of biology and is done through proteins. A protein's function and ability to bind to other molecules is determined by its folded three-dimensional structure, which results from its primary amino acid sequence, the polypeptide. The process by which the polypeptide determines the protein structure is one of the biggest questions in biochemistry and is known as the "protein structure prediction problem."[1] Understanding how proteins fold is fundamental to the understanding of many biological processes, including understanding the causes of certain diseases and how to design therapeutics. Abnormal protein structures, or misfolded proteins, are linked to numerous disorders, including Alzheimer's, Parkinson's disease, Huntington's disease, and cystic fibrosis. Many of the drugs used to treat various diseases target such misfolded proteins, so understanding how the folded protein is shaped, along with back-engineering amino acid sequences from desired structures, holds the key to the design and implementation of drugs that can alter protein function.[2] Generally, knowing how proteins fold can allow for the development of any specific protein that can perform any desired function.

The field of protein prediction was based on the idea that all the information needed for protein folding is contained in the energy landscape of a protein polypeptide (Anfinsen's thermodynamic hypothesis).[3] This also forms the reasoning behind methods of protein structure prediction, where possible shapes (or conformations) of a protein are scored based on their energy level and then sorted from lowest to highest energy levels, with the lowest energy structure being the most stable (and thus the most likely to be observed in nature).[4] The specific problem of finding the structure of a protein, given its amino acid bases, regardless of finding the optimal path to that structure, is called protein structure prediction (PSP).[5] The difficulty of PSP results from the exponentially expanding number of possible conformations a protein can adopt as its length increases. Along with the exponential search space, prediction algorithms also face problems due to the multitude of overlapping biochemical interactions that form a protein's energy landscape.

Traditionally, the discovery of new drugs has been accomplished through time-consuming and expensive wet lab experiments like nuclear magnetic resonance (NMR),[6] X-ray crystallography,[7] and cryo-electron microscopy.[8] Realizing their shortcomings, researchers turned to deep learning to solve PSP. One of the leading deep learning methods is template-based models. Algorithms like Deepmind's AlphaFold2, for instance, are making progress by using the large database knowledge of the Protein Data Bank.[9] However, due to their architecture, database models are limited to the types of proteins in their training data and struggle to maintain high accuracy when tested on novel proteins.

Alternatives to these database frameworks are models that attempt to simulate the physics of protein folding. But simulations also have their own set of problems, including sufficiently accurate computational models (forcefields) and the needed computational power when scaling protein size.[10] To address the computational size of this problem, we propose turning to a computational system that could potentially handle the size of the problem: quantum computing. With breakthroughs and research in quantum computing, and the beginning of theoretical research illustrating the application of integrating quantum ap-

DOI: 10.36838/v7i3.87

proaches in studying protein prediction,[11] we believe that the integration of quantum computing could lead to more accurate predictions, especially for novel and large protein structures.[12]

In this paper, we compare a feedforward neural network (FFNN) to a hybrid quantum-classical system, which uses a quantum neural network (QNN) coupled to a classical output layer, in solving a simplified version of the protein-structure prediction problem: a binary classifier. This classifier classifies amino acid sequences into an alpha-helix (α) or a beta sheet (β). Compared with the problem of predicting full tertiary structures (the standard and most computationally challenging version of protein folding), secondary structure prediction is an intermediate step in the folding process, though still integral to understanding full tertiary structure. Research has shown that the secondary structure is closely tied to the protein's overall conformation and that incorporating secondary structure prediction alongside residue-residue contact information can improve tertiary structure prediction accuracy.[13] Additionally, coupling the two predictions can enhance the accuracy of secondary structure predictions.[14] Through these trials, we illustrate that quantum-classical hybrid models have the potential for predicting protein structures and hope to inspire future research into quantum integration in PSP.

This work reviews the history and analysis of protein structure prediction to illustrate the limitations of database and physics-based classical machine learning methods. It then introduces quantum computing fundamentals and quantum neural networks to address how quantum computing can combat the limitations of classical models and provide its unique advantages in PSP. The paper showcases the classical and quantum models we used for our comparison and the CATH dataset we used to train and test our models. Our results demonstrate that the hybrid quantum-classical model achieved slightly higher accuracy on a 5-trial average, with a 4.75% improvement over the classical model when using 13 input features (qubits). The findings suggest potential advantages of incorporating quantum approaches into predictions and protein folding while also highlighting the need for further research in scaling quantum methods for broader applications in PSP.

Protein Structure Prediction in Classical Machine Learning:

NP-Hard Problem:

Any folded protein can be described by its amino acid sequence, and if we knew the sequence of a protein's amino acids, we could, in theory, predict the protein's shape. Yet protein prediction is still barely understood. Simply, the longer the protein, the harder it is to predict its native structure. This difficulty is quantified through PSP classification as **NP-hard** in computational complexity theory.[15] Computational complexity theory classifies problems based on how much time and input size they require to be solved. Polynomial-time algorithms can be executed in a computational time that increases at a polynomial rate as the input size grows. Such problems are known as **P** (or P-time). The class **NP** (nondeterministic polynomial time) refers to problems where the proposed 'solutions' can be verified in polynomial time, yet they may not be solvable in polynomial time. The class of NP-hard problems includes those that are at least as hard as the hardest problems in NP and often take exponentially more time to solve as the size of the input increases. The protein folding problem is NP-hard because the number of possible shapes that a protein can take increases exponentially with its length because any one amino acid can and will interact with the other amino acids in the chain.[16-17]

The Energy Landscape of Proteins:

The theory of energy landscape started with Anfinsen's thermodynamics hypothesis, stating that all information needed for protein folding was encapsulated within the primary sequence of amino acids and that the protein found its native state by finding the state of lowest free energy.[18] This allows a protein's final structure to be predicted through its sequence alone if the energy landscape could be accurately modeled and traversed.[3] Levinthal's Paradox states that it would be longer than the age of the universe for an average-length protein to find its native state. Yet, proteins fold in microseconds.[19-20] Energy landscape theory answers this.[21] It describes protein folding as a trip through an energy landscape funnel (Figure 1). The unfolded protein at the top of the funnel has high energy, entropy, and possible conformations and is then driven by energetically favorable interactions as it folds. The protein follows these interactions until it reaches its native folded state at the bottom of the funnel.[22]

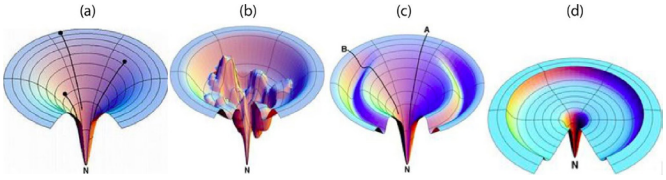

Figure 1: This graphic was created and published in the public domain by Ken A. Dill. Original caption: The illustrations of proposed energy landscapes that each demonstrate the degree of freedom a protein possesses in terms of configurations and the multidimensional routes that a protein can take to achieve its final configuration. From left to right for the proposed funnel-shaped energy landscapes: the idealized smooth funnel, the rugged funnel, the Moat funnel, and the Champagne Glass funnel (Dill & Chan, 1997)

In addition, it was later found that the energy landscape was rugged (Figure 1b). This ruggedness results from the interactions between amino acids, including hydrogen bonding, van der Waals forces, electrostatic interactions, and hydrophobic effects. The balance of favorable (lowering the energy) and unfavorable (raising the energy) interactions creates local variations in the energy landscape, such that each point on the funnel corresponds to a specific protein conformation, and small changes in the amino acid positions can lead to significant changes in the energy topology. From the perspective of the prediction models, the valleys represent local energy minima (conformations where the protein is in a metastable but not native state). Prediction algorithms can get trapped in these energy barriers, mistakenly taking them for global minima. Proteins solve this problem by using chaperones, which help them to escape local minima and flatten the energy landscape. Adding chaperone effects to the prediction algorithms

adds yet another layer of complexity since it now becomes necessary to model not just the protein but also its interaction with outside molecules.

Database Approaches to PSP::

Database PSP approaches are based on the assumption that a similar protein sequence is likely to fold into a similar structure. These models rely on the huge amount of experimentally determined protein structures available in databases, and then for an unknown protein sequence, the model deduces the most likely structure through a comparison between the unknown sequence and experimentally known structures. The biannual Critical Assessment of Techniques for Protein Structure Prediction **(CASP)** competition was established in 1994 to evaluate these methods. Currently, most PSP models base their predictions on comparative or homology modeling. Homology modeling is performed by comparing the predicted sequence against the sequences in the PDB. If a homologous structure is found, the coordinates of a model's backbone atoms in the temple are mapped to the atoms of the backbone in the new sequence.[23] The accuracy of these models is related to how closely the target and the template sequence match. As a result, the accuracy drops drastically for dissimilar sequences, and the GDT-TS falls below 50 for more difficult targets (Figure 2).[24]

The iterative database model **AlphaFold2** (released at CASP14, 2020) is the highest-accuracy PSP model.[9] It uses a deep learning transformer architecture with a refinement process where the computed residue distances and angles between residues are used to re-assess the amino acid sequences iteratively.[9] Accuracy increases in the CASP models from 1994 to 2020 are shown in Figure 2 below, with AlphaFold2 obtaining a prediction accuracy above 90 for even some difficult targets. Database models can still give errors, however. The most sophisticated database models can still be challenged by proteins with disordered regions, complex structures, membrane proteins, and overall novel protein structures. Furthermore, database models do not capture dynamic aspects of protein functions and lose substantial accuracy on proteins that are significantly different from known structures.

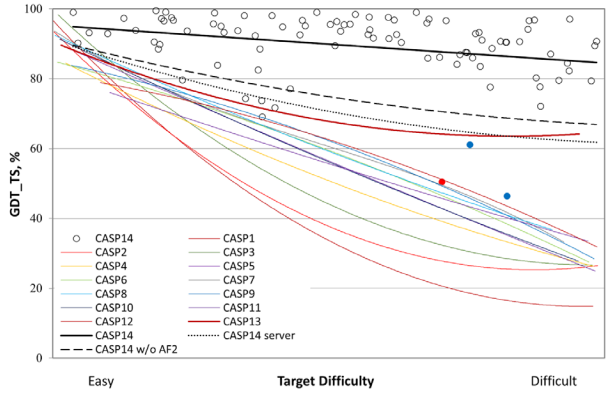

Figure 2: Combined results of all the CASP competitions. The dotted line is predictions made by automated servers. The dashed line includes all predictions assisted by humans besides the highest performing group; the black line represents the predictions by Group 427, or AlphaFold 2. This plot uses the GDT_TS score, in which 100 is a perfect result, and 0 is a meaningless prediction. GDT-TS refers to the global distance test: a quantitative measure of structural similarity between two structures.

Physics-Based Approaches to PSP:

The other prevalent method for protein prediction comes from kinematically simulating how proteins go through the energy landscape funnel to their native state. Physics-based ab initio modeling to PSP is based on molecular physics and involves simulating the protein's thermodynamic or kinetic behavior at the atomic level. The most commonly used technique is **all-atom molecular dynamics** (MD) simulations. By modeling the kinetics of the atoms and molecules in a protein, MD provides us with a detailed, time-resolved picture of the folding process.[25] A mathematical model called a **force field** regulates the interactions of particles representing the protein and particles representing the molecules (eg, water) surrounding it. Based on empirical data and molecular calculations, the force field describes bonded and non-bonded interactions between amino acids. The simulation tracks how the particles move in time by solving Newton's equations of motion in minute time steps (typically 1 to 2 femtoseconds). Since the total time for protein folding can be in the range of microseconds to milliseconds, this requires approximately 10^{12} femtosecond time steps to simulate larger proteins.[26]

As a consequence, early MD approaches were stunted by the computing power required to model even a single folding event; and the result's accuracy depends on the forces of computational algorithms, with new sampling methods and improved biasing techniques speeding up computations. Currently, physics-based models using deep learning algorithms like trRosetta with MD simulations provide increasingly accurate simulations of proteins' energy landscapes and more reliable predictions of structure and folding pathways. For instance, Audagnotto *et al.* (2022) demonstrated that their computational pipeline could observe entire folding events from denatured conformations to native states in proteins like adenylate kinase and T4 lysozyme, successfully capturing the conformational changes and intermediate states observed experimentally.[27] Since then, these approaches have been refined regarding the force fields used in MD simulations and computational algorithms, with new sampling methods and improved biasing techniques speeding up computations. Physics-based models provide increasingly accurate simulations of the protein's energy landscape and more reliable predictions of structure and folding pathways. In some cases, it's possible to observe entire folding events, from denatured conformations to native states.[28] While force-field accuracy has also been greatly improved, electrodynamic and hydrogen bonding inaccuracies, for example, still lead to significant errors in predicting the native state of proteins.[29,30] As simulations of larger proteins progress, there is a growing need for either more efficient models or more powerful computing techniques, with a potential method being quantum computing.

Quantum Computing

Quantum computing is a type of computing that uses the principles of quantum mechanics to make calculations. The next two sections provide an overview of quantum computing and quantum neural networks (QNNs), respectively. Readers are encouraged to read[31] and[32] for a more in-depth introduc

DOI: 10.36838/v7i3.87

tion to quantum computing. As well as[33] for more information on QNNs.

A Brief Introduction to Quantum Computing:

A quantum bit (or qubit) in quantum computing is the quantum analog of the classical bit. A qubit, unlike the familiar *0* or *1* of a classical bit, can be in a superposition of two states where it has a probability of being in either a *0* or *1* state. Dirac notation is used to describe quantum states, where $|0\rangle$ represents the qubit being in the 0 state and $|1\rangle$ represents the qubit being in the *1* state. In quantum mechanics, the wave function ψ(psi) is used to describe the overall quantum state of a system, $|\psi\rangle$ in ket notation. Any qubit can be represented as the superposition $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where α and β are probability amplitudes such that $\alpha|^2 + \beta^2 = 1$. This normalization condition means that when the qubit is measured, it has a probability $|\alpha|^2$ of being measured in a $|0\rangle$ state and a probability $|\beta|^2$ of being measured in a $|1\rangle$ state.

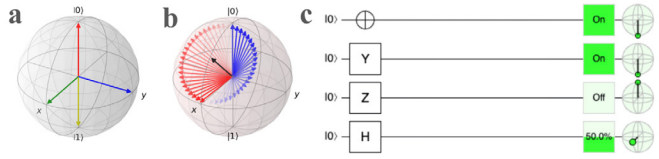

Figure 3: Illustrations of quantum gates and the Bloch Sphere. (a) Bloch Sphere Representation: This panel shows the Bloch sphere, representing qubit states with the poles as $|0\rangle$ and $|1\rangle$ and X and Y axes indicating possible rotations (created using QuTiP). (b) Hadamard Gate Rotation: This panel depicts the rotation of a qubit state from $|0\rangle$to a superposition state on the X-Z plane after applying a Hadamard gate (created using QuTiP). (c) Quantum Circuit Outcomes: This panel shows the effects of various quantum gates (CNOT, Y, Z, Hadamard) on a qubit, visualized with measurement probabilities and Bloch sphere states (created using Quirk).

A qubit can also be represented graphically as a point on the surface of a unit sphere known as the Bloch sphere (Figure 4a). The Bloch sphere shows the information stored in a single qubit, where every possible superposition of the qubit is represented by a point on the sphere's surface. The sphere's poles correspond to the basis states $|0\rangle$ and $|1\rangle$, while any other point on the sphere represents a superposition of these states. In Cartesian coordinates, a qubit state can be expressed as the following superposition:

where θ and Φ are angles that determine the position of the

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle$$

qubit on the Bloch sphere, the angle θ represents the superposition amount between $|0\rangle$ and $|1\rangle$ (the likelihood of each state when measured), and Φ represents the relative phase between the two states, represented in the Bloch sphere as a rotation in the X-Y plane. Adjusting these two angles with quantum gates changes the qubit's state, forming the basis for data encoding on qubits. The most basic gates are the Pauli Gates: X, Y, and Z. They correspond to rotations around the three axes in the Bloch Sphere. Figure 4a summarizes them, and Figure 3c illustrates their effects on a quantum circuit. Another important gate to quantum computing is the Hadamard gate, which is one of the main ways superposition is created. The Hadamard gate takes in a $|0\rangle$ or $|1\rangle$ state and transforms the qubit into an equal superposition of $|0\rangle$ and $|1\rangle$. Mathematically, the Hadamard gate H is represented as the matrix:

When applied to the basis states, the Hadamard gate operates as follows:

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}$$

This means that after applying the Hadamard gate to a qubit initially in the state $|0\rangle$ or $|1\rangle$, the qubit will be in a superposition of $|0\rangle$ and $|1\rangle$, with equal probability amplitudes for both states. The power of quantum computing becomes apparent when we consider multiple qubits systems. Looking at a system with two qubits. If both qubits are in the state $|0\rangle$

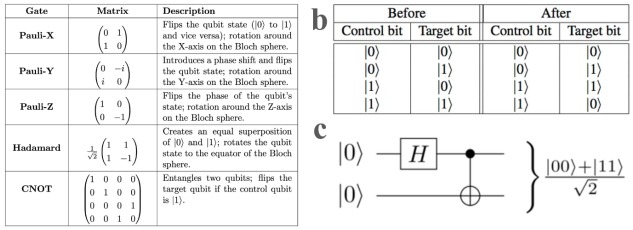

Figure 4: Illustrations of quantum gates and Bell States. (a) Matrixes and descriptions for the Pauli, Hadamard, and C-NOT gates. (b) The before and after snapshot of the CNOT gate entangling 2 qubits into the 4 Bell States· The target bit flips the control bit (via an X gate) if the control is in a $|1\rangle$ state and does nothing if the control is in a $|0\rangle$ state. (c) Quantum circuit diagram for the first Bell State. The control qubit first goes$H$ through a H gate, resulting in an equal superposition. After going through the C-NOT gate, the target qubit becomes entangled with the control qubit, such that if the control qubit is in a $|1\rangle$ state, the target qubit will flip to a one $|1\rangle$ state, and otherwise, they will both be in a $|0\rangle$ state. This results in the entangled Bell state shown in the diagram. (Created with IBM Qiskit)

and we apply the Hadamard gate to each qubit, the system will turn into a superposition of four possible states:

Expanding this expression, we get:

$$|\psi\rangle = H|0\rangle \otimes H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

$|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$ are the four possible states for the

$$|\psi\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$$

two-qubit system. Each of them is equally likely, and scaling to a system with n qubits, each will have a probability of $1/2^n$, and they can exist in a superposition of 2^n states. This scaling is exponential, which is one of the key reasons why quantum computers are considered to be powerful for certain computations, such as searching large databases or simulating NP systems.

The other main difference between quantum and classical computing is **entanglement**, where the states of two or more qubits become correlated such that the state of a single qubit cannot be determined without also determining the state

of other qubit(s). In many quantum algorithms, the CNOT gate is used to produce entangled states. The CNOT gate acts on one control qubit and one target qubit. When the control qubit is in the state $|1\rangle$, the CNOT gate inverts the state of the target qubit; otherwise, if the control qubit is in the state $|0\rangle$, the target qubit does not change (Figure 4b). The matrix representing the CNOT gate is seen in Figure 4a. To create an entangled state, you can apply a Hadamard gate to the first qubit (putting it into a superposition of $|0\rangle$ and $|1\rangle$) and then apply the CNOT gate, using the first qubit as the control qubit and the second qubit as the target. The result is an entangled state (Figure 4c), specifically it is one of the four Bell states:

$$|\Phi^-\rangle = \tfrac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \qquad |\Phi^+\rangle = \tfrac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

$$|\Psi^+\rangle = \tfrac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \qquad |\Psi^-\rangle = \tfrac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$$

Each Bell state is a superposition of two-qubit states in which a measurement of any one qubit instantly determines the precise state of the other, no matter how far away it might be. Although non-local correlation cannot be used to send information faster than the speed of light, it is a valuable resource for quantum information processing. It has been used to create the quantum communication protocols of quantum teleportation, superdense coding, and quantum cryptography.

These quantum-mechanical phenomena–superposition and entanglement–can be used to design quantum algorithms that efficiently search cost landscapes. Quantum speedup, the ability of quantum computers to solve problems faster than classical computers, means that specific problems like factorization,[34] that are intractable using classical computers, are significantly more efficient when using quantum computers.

Quantum Neural Networks:

QNNs are the quantum version of ANNs, and they work based on superposition, entanglement, and quantum parallelism. Figure 5 illustrates the three parts of a QNN: the embedding layer, the quantum layers, and measurement. The embedding layer is the first part of a QNN, where classical data is embedded into quantum states through a quantum gate, the *U*(*x*) operator in Figure 5a. One of the most common methods this is done through is angle encoding. In angle encoding, each classical input data point *n* is mapped onto a qubit by rotating the qubit's state on the Bloch sphere. The generic rotation operation *R*(*θ*) applied to a qubit can be defined as

$$R(\theta, \hat{n}) = \exp\left(-i\frac{\theta}{2}\left(\hat{n}_x\sigma_x + \hat{n}_y\sigma_y + \hat{n}_z\sigma_z\right)\right)$$

where *θ*= *xi* represents the angle determined by the data point, $\hat{n}$ is the axis of rotation along the Bloch sphere, and *σ*

is the vector of a Pauli matrices (*σ*x or *σ*y or *σ*z). This operator allows the qubit to be rotated to a specific point on the Bloch sphere, encoding the classical information into some quantum state $|\psi(x_i)\rangle = R(\theta_i)|0\rangle$. The embedding layer takes in a vector of input data and applies these rotations across multiple qubits, resulting in a quantum state that encodes the entire dataset. The result is a combined quantum superposition $|\psi(x_i)\rangle$, where all the information from the input data

is encoded.

Following the embedding, the quantum layers, represented by the W(θ) operator in Figure 4a, perform the main computations in a QNN. These layers consist of a sequence of quantum gates that tweak the quantum state to explore the solution space efficiently. Generic rotation gates *R*(*θ*) are used within the quantum and can rotate the qubit around any axis on $\hat{n}$

the Bloch sphere, providing the model greater control over the quantum state. These rotations adjust the quantum state like weight adjustments in classical neural networks. The quantum layers also include entanglement operations that create connections between qubits that cannot be replicated in classical systems. In Figure 5b, the entanglement layer is shown as a cyclical entanglement pattern, where each qubit is entangled with its neighbor in a circular fashion (e.g., qubit 1 entangles with qubit 2, qubit 2 with qubit 3, and so on, with the last qubit entangled back to the first). Mathematically, this is represented by a sequence of Controlled-NOT

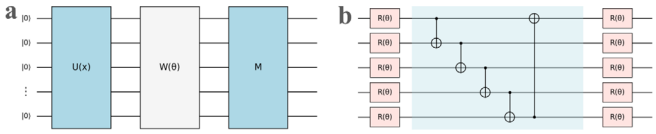

Figure 5: QNN Backbone. (a) Depicts the basic structure of a QNN circuit. The operator U(θ) is the embedding block. The main techniques for said embedding are angle embedding and amplitude embedding. The W(θ) operator illustrates the hidden quantum layers that consist of rotation gates and entanglement layers. The M operator illustrates the end of the QNN when the qubits are measured, and their state is saved or sent to classical layers. (b) Quantum circuit of one hidden layer in the W(θ) operator. The hidden layer consists of a series of rotation gates, R(θ), and an entanglement layer, in this case, cyclical entanglement. The entanglement layer is a series of C-NOT gates placed on each qubit to entangle them. These two operations are repeated for every quantum layer in the QNN.

(CNOT) gates, which, when applied to a superposition state, result in an entangled quantum state:
$|\psi_{\text{entangled}}\rangle = \text{CNOT}_{(i,j)} \cdot R(\theta) \cdot |\psi(x)\rangle$.

This layer's function is to increase the complexity of the quantum state, enabling the QNN to represent more intricate patterns and find more correlations in the data.

In a QNN, each qubit stores the superposition of both states, and operation on the qubits affects the $|0\rangle$ and $|1\rangle$ components simultaneously. In a classical counterpart, the only way to process the result is by evaluating each of the 2^n possible states one by one. However, using quantum parallelism a QNN can process multiple states at once. This property allows QNNs to explore certain solution spaces faster than classical networks. For instance, when a quantum gate is applied to a qubit in superposition, it simultaneously affects all possible states rep

DOI: 10.36838/v7i3.87

resented by the superposition, allowing the QNN to perform multiple computations in parallel, significantly proving the processing power of QNNs.

The final stage in a QNN is measurement, represented by the M operator in Figure 4a. Measurement collapses the quantum state back into classical information, where each qubit is observed in the $|0\rangle$ or $|1\rangle$ state, with the probability of each state being determined by its probability amplitude. For each qubit, the probability p of measuring a specific state, for example, $|1\rangle$, is given by $p(1) = |\langle 1|\psi_{final}\rangle|^2$, with $|\psi final\rangle$ being the state of the qubit before measurement. In real quantum computers, the measurement process is inherently probabilistic. Unlike numerical simulations, where exact expectation values can be calculated, real quantum hardware only allows for approximate values based on a finite number of measurements. The outcome is a set of discrete measurement results, each with a certain probability. The process is repeated many times to obtain a reliable estimate of the probability $p(1)$. Specifically, the error is quantified by the Chernoff bound, where $O(1/\varepsilon^2)$ repeat measurements are needed to obtain an expectation value with a combined error up to ε.[35]

Quantum Application in PSP:

The use of quantum computers in PSP could result in stronger, more accurate models. These advantages stem from the unique properties of quantum computing systems, namely superposition and entanglement, which allow quantum computers to explore certain solution spaces more efficiently than classical computers.[36] The most urgent challenge for database and physics-based methods alike is that PSP's computational complexity increases exponentially with the size of the protein. A quantum computer's ability to control exponentially large state spaces using a linear number of qubits might be able to solve the scaling problem, allowing prediction of the structure of larger proteins or protein complexes. In addition, as depicted in Figure 1b, the energy landscape funnel is characterized by myriad local minima and, thus, a rugged shape. The ruggedness of the landscape is an obstacle for classical algorithms as they might be trapped in local minima, which prevents them from finding the native state of the protein. In such cases, quantum algorithms that can exist in superposition may use parallelism to explore more than one conformation at the same time, and the superposition property may help quantum systems avoid the barriers between local minima, which restrict classical algorithms in navigating through the rugged energy landscape. The ruggedness of the energy landscape at the bottom of the folding funnel can be correlated with the conformational heterogeneity of the protein. Rugged, multi-minima landscapes can lead to more flexible proteins that bind non-specifically to ligands. They do this by selecting a conformation from a possible distribution of low-energy states that is complementary to each ligand.[38] It is possible that the use of entangled states would allow quantum computing models to better encode these many-body interactions, as they would, in principle, allow us to correlate different quantum interactions with each other. The concept of correlation has already emerged in PSP in models such as Alphafold2 through the use of attention-based (transformer) architecture.[9] Though research into quantum approaches to PSP is still largely unstudied, as seen in pilot studies 12 and 37, some specific quantum models have had success in localized prediction cases. Having an archetype of QNNs with entanglement layers in our models, we could capture and model the relationship between amino acid interactions when looking at secondary structure.

Methods

Dataset:

The protein sequences were taken from the CATH (Class, Architecture, Topology, Homology) database, which classifies protein domains into their evolutionary lineages based on specified structure details. Two different groups were selected: 'Mainly Alpha' and 'Mainly Beta' for a balanced dataset that our secondary structure binary classifier would use. One thousand sequences were extracted from the FASTA files for each group, denoted by the prefixes cath-superfamily-seqs-1 and cath-superfamily-seqs-2 for the α and β sequences, respectively. Amino acids within these sequences were encoded using a standard 25-character alphabet formalized by the UPAC-IUB.

General Approach:

The general approach involved transforming the protein sequences into a set of features that could be used as inputs to both classical and quantum models. Specifically, we selected 30 features based on key physicochemical properties linked to protein folding, like hydrophobicity and charge. Each feature was calculated using specific groups of amino acids that have been found to align with the property being measured.[39]

For example, the "hydrophobic" feature was determined by counting the occurrences of known hydrophobic residues (A, I, L, M, F, W, V) within the sequence. Given an example input sequence, **M**PENP**AA**EK**M**Q**V**LQ**V**LD, the hydrophobic residues (M, A, A, M, V, L, V, L) would be tallied, resulting in a "hydrophobic" value of 8 for the sequence. All other features were handled similarly to ensure uniform input preparation across the dataset for classical and quantum models.

Since our quantum model was simulated, the classical system needed to simulate the exponential scaling of a qubit system, causing the training time to increase exponentially with the number of input variables (qubits). In our initial trials, we lowered the total inputs to 9 distinct features, then in our final trials back up to 13 features (where the number of qubits is equal to the number of features and input variables). To determine these 13 inputs, we calculated all 30 features' average values over the α and β sequences. Then to find which features gave the most useful data, we calculated and plotted the difference in the averages (Figure 6).

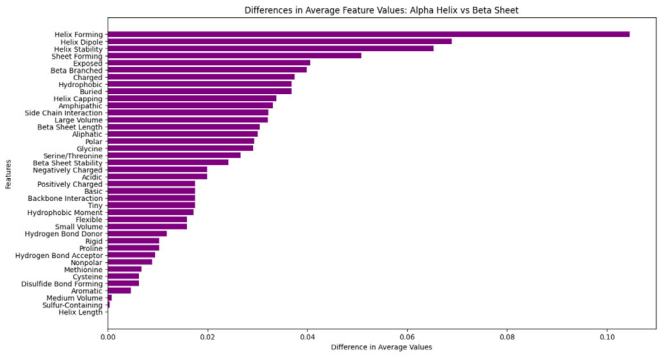

Figure 6: A plot of all 30 chosen features, ranked in order of descending magnitude of the difference in average values of the feature in the α and β datasets.

By plotting the differences, we could identify features that, on average, provided distinct values for the two secondary structures. We selected the features with the largest differences, as these are likely to help the models differentiate between α and β sequences. The 13 most distinct features, after accounting for overlap, are shown in Figure 7, with their descriptions listed in Table 1.

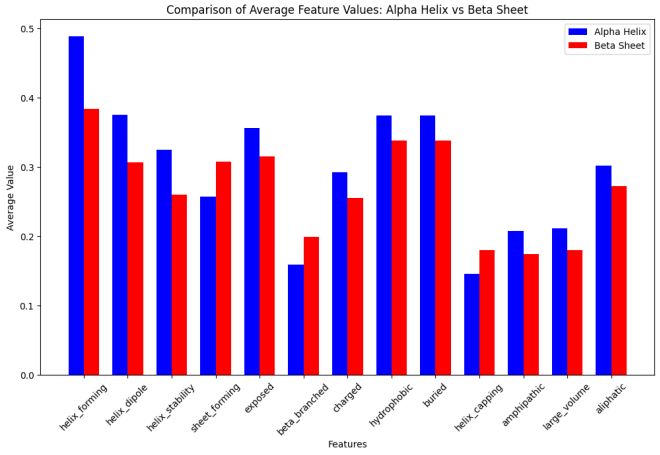

Figure 7: This graph shows the average values for the two secondary structures on the 13 features we chose as inputs. For each feature, we plotted its average value for the α and β sequences and ordered them in descending order according to the differences in the two averages.

Table 1: This table displays the 13 selected features and their descriptions, explaining how each feature helps to differentiate between structures α and β structures.

Feature	Description	Feature	Description
Helix-Forming Propensity	High tendency of amino acids to form in alpha-helix structures	Hydrophobic	The tendency of amino acids to avoid water, driving protein folding and core formation
Helix Dipole	The distribution of partial charges along an alpha-helix; affects stability and interactions	Buried	Propensity of amino acids to be located in the interior of a protein structure, away from the aqueous environment
Helix Stability	The ability of amino acids to maintain the structural integrity of alpha-helices	Helix Capping	The ability of certain amino acids to stabilize the ends of alpha-helices
Sheet-Forming Propensity	High tendency of amino acids to form in beta-sheet structures	Amphipathic	Property of having both hydrophobic and hydrophilic regions, important in membrane-associated proteins
Exposed	The likelihood of amino acids being located on the protein surface	Large Volume	Amino acids with bulky side chains can influence protein packing and structure
Beta-Branched	Amino acids containing two substituents attached to their C-beta carbon	Aliphatic	Non-polar, non-aromatic side chain amino acids
Charged	Amino acids with electrically charged side chains influencing protein-protein interactions and solubility		

Classical Model:

For the classical model, we used a 4-layer fully connected feedforward neural network (FFNN) implemented through TensorFlow's Keras API. The architecture included input layers mapping the processed features of the amino acid sequences to four dense layers, ranging from 10 to 50 nodes with ReLU (rectified linear unit) activations, which were found to be the most accurate when using cross-validation with random search. To account for overfitting, each dense layer incorporated a dropout rate of 0.2. The output layer used a binary softmax function to classify each sequence as belonging to either the α or β secondary structure.

Quantum Model:

The quantum neural network (QNN) was implemented using PennyLane, which interfaces with TensorFlow to build hybrid simulated quantum models. In the quantum model, amino acid sequences were first encoded into quantum states using angle embedding, where the features calculated from the sequences were used to determine the rotation angles of the qubits on the Bloch sphere. The QNN circuit consisted of multiple quantum layers, including rotation gates (parameterized by trainable weights) and entanglement layers. The entanglement pattern used in this study was cyclical, ensuring each qubit interacted with its neighbors, creating a highly interconnected quantum state. Following these quantum layers, a final measurement step converted the quantum state back into classical data[34], then processed by a classical neural network. The quantum circuit used in the QNN is depicted in Figure 8 and the final classical layer of the model is illustrated in Figure 9.

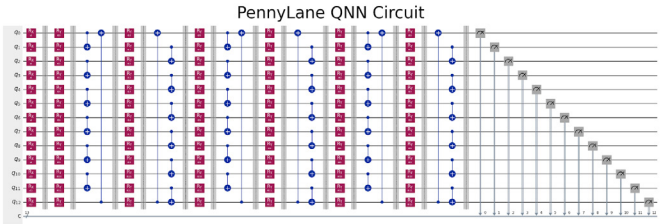

Figure 8: Our final Pennylane QNN Circuit. The diagram above illustrates the QNN portion of our hybrid model. Only 6 quantum layers are shown for clarity; our final models had 15 total quantum layers.

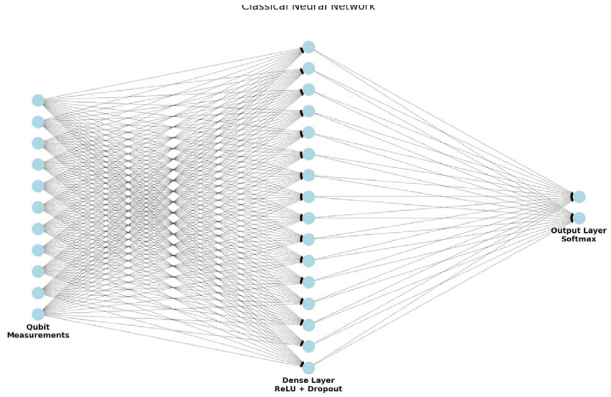

Figure 9: Illustrates the final classical layer for our hybrid quantum-classical model.

DOI: 10.36838/v7i3.87

Results and Discussion

We first ran our models with qubit counts of 9 input features, from which we plotted the averages of 5 trials of both the classical and QNN hybrid models. Figure 10 gives the "Loss vs. Epoch" and "Accuracy vs. Epoch" graphs for our 9-qubit model comparison. We also ran a larger experiment with a 13-input feature model to see how the QNN hybrid model compares to the classical model as we increase the qubit count. The results of the 5-trial average can be seen in Figure 11.

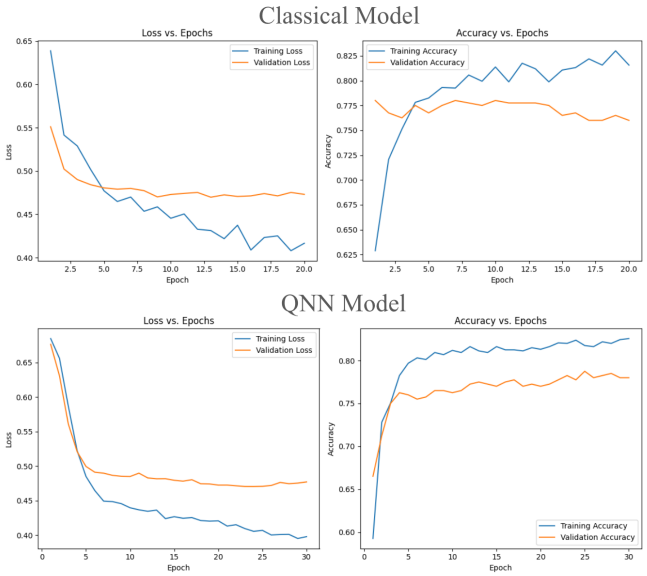

Figure 10: Graphs of the classical and QNN hybrid models with 9 feature inputs. Our classical models used 20 epochs, and our QNN models used 30 epochs. In 5 trials, our classical models achieved an average highest accuracy of 78.0%, and our QNN models achieved 78.5%.

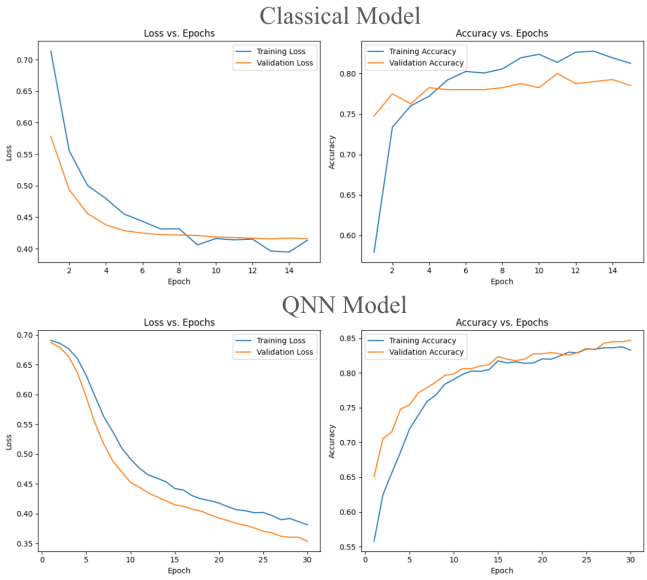

Figure 10: Graphs of the classical and QNN hybrid models with 13 feature inputs. Our classical models used 14 epochs, and our QNN models used 30 epochs. In five trials, our classical models achieved an average highest accuracy of 79.750%, and our QNN models achieved 84.5%.

From our trials, we found that QNN hybrid models outperformed their classical counterparts. In the 9-qubit trials, we see a modest 0.5% accuracy increase over classical models, whereas, in the 13-qubit trials, this gap increases with the average accuracy of QNN models outperforming classical models by 4.75%. As this average was calculated through a small number of trials (5 for each model type), a greater number of trials, with a larger data size, would need to be run to statistically validate these results. Despite our increased dataset size (1,000 sequences per structure), we still observed signs of overfitting. This was particularly evident in the classical model and, to a lesser extent, in the QNN hybrid model. This suggests that 1,000 sequences is not a large enough data set to provide sufficient diversity in the problem space for optimal model training. The apparent performance gap between QNN and classical hybrid models suggests that quantum approaches might have some advantages in capturing complex relationships in the protein structure data, which is in line with our predictions that quantum systems can navigate the rugged energy landscapes characteristic of protein folding more efficiently, and that a quantum advantage might scale with the level of complexity. However, this also highlights the need for large-scale studies. Moving on from this fairly simplistic binary classifier to more complex tertiary structure predictions is likely to require a significant leap forward on both fronts: in terms of the size and robustness of the quantum hardware (measured by the number of qubits) and in terms of the algorithm design. While these results are promising, they should be seen as an early indicator rather than proof of quantum advantage in protein structure-prediction problems.

Higher accuracy in the QNN hybrid model could be due to quantum-specific advantages in data-relationship processing, and it may be that the quantum model's ability to represent and process high-dimensional spaces enabled by superposition is helping it to capture the subtlety of amino acid sequence to secondary structure relationships. The non-linear transformations performed by the quantum circuit, through its series of rotations and entanglements, may give it a more expressive feature space, especially good at distinguishing alpha from beta structures. Further, quantum parallelism in the circuit may allow it to explore conformational protein space more efficiently, analogous to the way that proteins sample conformations in parallel during folding. The larger accuracy gap between the 9 and 13 qubit trials signifies the possibility that our quantum models excel when facing a more diverse dataset, where its ability to correlate features via entanglement could be a factor in explaining the higher relative accuracy scores.

As these findings are based on a simplified binary classification task (alpha vs. beta structures) and a limited set of input features, more research needs to be done in scaling these models to more expansive secondary structure datasets and eventually tertiary folding. While the full quantum advantage is yet to be realized, combining classical and quantum methods offers a practical approach today. Classical models can process simpler tasks effectively, while quantum circuits are integrated into parts of the pipeline that require handling complex, high-dimensional patterns. In terms of quantum application in PSP, more studies in implementing quantum computing into specific tasks within classical models will bring greater insight into the ability of higher qubit systems to excel. To

test the hypothesized quantum speedup and potential computational advantages of quantum computing in PSP, more extensive experimentation with higher qubit systems and larger proteins is needed. Future work could involve training models on available real qubit systems, such as IBM's 127-qubit quantum computer. The ability to use real quantum computers instead of computationally slow classical simulators would greatly increase the amount of qubits that could be used in a similar computation time. Additionally, research into error correction methods, including the development of logical qubits, will be crucial for improving the reliability and scalability of quantum models for PSP.[40] Finally, with the release of Deepmind's AlphaProteo in September 2024, there are more ideas for integrating quantum models in the design of novel protein binders and in finding amino acid sequences from the desired structure.[41]

Conclusion

This study used the CATH database to compare classical and quantum-inspired approaches to protein secondary structure prediction. Our QNN hybrid models outperformed classical feedforward neural networks, achieving 84.5% accuracy versus 79.75% with 13 input features. These results suggest that quantum approaches might be better at capturing complex relationships in protein structure data and searching rugged energy landscapes more efficiently. While promising, more extensive and diverse research is needed to see more conclusive data; Progressing towards tertiary structure predictions requires significant advancement in quantum hardware and algorithms. In future research, Quantum and classical methods could work synergistically, with quantum approaches complementing template and physic-based methods in advancing progress in PSP.

Acknowledgments

I was the sole contributor to all images and writing (unless otherwise specified). I would also like to acknowledge and thank Dr. Sakk for his mentorship and guidance throughout this project and Colin Rose for introducing me to the topic of protein structure prediction.

References

1. Dobson, C. M. Protein Folding and Misfolding. *Nature* 2003, *426* (6968), 884–890. https://doi.org/10.1038/nature02261.
2. Dill, K. A.; Ozkan, S. B.; Shell, M. S.; Weikl, T. R. The Protein Fro lding Problem. *Annu. Rev. Biophys.* 2008, *37*, 289–316. https:/doi. org/10.1146/annurev.biophys.37.092707.153558.
3. Anfinsen, C. B. Principles That Govern the Folding of Protein Ch ains. *Science* 1973, *181* (4096), 223–230. https://doi. org/10.1126/science.181.4096.223.
4. Cooper, A. Thermodynamics of Protein Folding and Stability. *Pro tein: A Comprehensive Treatise* 1999, *2*, 217–270.
5. Kuhlman, B.; Bradley, P. Advances in Protein Structure Prediction and Design. *Nat. Rev. Mol. Cell Biol.* 2019, *20* (11), 681–697. https ://doi.org/10.1038/s41580-019-0163-x.
6. Marion, D. An Introduction to Biological NMR Spectroscopy. *M ol. Cell. Proteomics* 2013, *12* (11), 3006–3025. https://doi.org/10.10 74/mcp.O113.030239.
7. Maveyraud, L.; Mourey, L. Protein X-ray Crystallography and Dr ug Discovery. *Molecules* 2020, *25* (5), 1030. https://doi.org/10.3390 /molecules25051030.
8. Yip, K. M.; Fischer, N.; Paknia, E.; Chari, A.; Stark, H. Breaking t he Next Cryo-EM Resolution Barrier – Atomic Resolution Deter mination of Proteins! *bioRxiv* 2020, https://doi.org/10.1101/2020. 05.21.106740.
9. (1) Jumper, J.; Evans, R.; Pritzel, A.; Green, T.; Figurnov, M.; Ron neberger, O.; Tunyasuvunakool, K.; Bates, R.; Žídek, A.; Potapen ko, A.; Bridgland, A.; Meyer, C.; Kohl, S. A. A.; Ballard, A. J.; Co wie, A.; Romera-Paredes, B.; Nikolov, S.; Jain, R.; Adler, J.; Back, T.; Petersen, S.; Reiman, D.; Clancy, E.; Zielinski, M.; Steinegger, M.; Pacholska, M.; Berghammer, T.; Bodenstein, S.; Silver, D.; Vi nyals, O.; Senior, A. W.; Kavukcuoglu, K.; Kohli, P.; Hassabis, D. Highly Accurate Protein Structure Prediction with AlphaFold. *Nature* 2021, *596* (7873), 583–589. https://doi.org/10.1038/s415 86-021-03819-2.
10. Lane, T. J.; Shukla, D.; Beauchamp, K. A.; Pande, V. S. To Millise conds and Beyond: Challenges in the Simulation of Protein Foldi ng. *Curr. Opin. Struct. Biol.* 2013, *23* (1), 58–65. https://doi.org/10. 1016/j.sbi.2012.11.002.
11. Pal, S.; Bhattacharya, M.; Lee, S. S. Quantum Computing in the Next-Generation Computational Biology Landscape: From Prote in Folding to Molecular Dynamics. *Mol. Biotechnol.* 2024, *66*, 163 –178. https://doi.org/10.1007/s12033-023-00765-4.
12. (1) Robert, A.; Barkoutsos, P. Kl.; Woerner, S.; Tavernelli, I. Reso urce-Efficient Quantum Algorithm for Protein Folding. *npj Qua ntum Information* 2021, *7* (1), 38. https://doi.org/10.1038/s41534 -021-00368-4.
13. Meiler, J.; Baker, D. Coupled Prediction of Protein Secondary an d Tertiary Structure. *Proc. Natl. Acad. Sci. U.S.A.* 2003, *100* (21), 1 2105–12110. https://doi.org/10.1073/pnas.1831973100.
14. Dirac, P. A. M. *The Principles of Quantum Mechanics*; Oxford: Cl arendon Press, 1981.
15. Pierce, N. A.; Winfree, E. *Protein Design is NP-Hard. Protein En g. Des. Sel.* 2002, *15* (10), 779–782. https://doi.org/10.1093/protei n/15.10.779.
16. Fraenkel, A. S. Complexity of Protein Folding. *Bull. Math. Biol.* 1 993, *55* (6), 1199–1210. https://doi.org/10.1016/S0092-8240(05) 80170-3.
17. Berger, B.; Leighton, T. Protein Folding in the Hydrophobic-Hy drophilic (HP) Model is NP-Complete. *J. Comput. Biol.* 1998, *5* (1), 27–40. https://doi.org/10.1089/cmb.1998.5.27.
18. Anfinsen, C. B.; Haber, E.; Sela, M.; White, F. H. The Kinetics o f Formation of Native Ribonuclease during Oxidation of the Red uced Polypeptide Chain. *Proc. Natl. Acad. Sci. U.S.A.* 1961, *47* (9), 1309–1314. https://doi.org/10.1073/pnas.47.9.1309.
19. Karplus, M. The Levinthal Paradox: Yesterday and Today. *Foldin g Des.* 1997, *2* (4), S69–S75. https://doi.org/10.1016/s1359-0278 (97)00067-9.
20. Levinthal, C. Are There Pathways for Protein Folding? *J. Chim. Phys.* 1968, *65*, 44–45. https://doi.org/10.1051/jcp/1968650044.
21. Dill, K. A.; MacCallum, J. L. The Protein-Folding Problem, 50 Years On. *Science* 2012, *338* (6110), 1042–1046. https://doi.org/10 .1126/science.1219021.
22. Schafer, N. P.; Kim, B. L.; Zheng, W.; Wolynes, P. G. Learning to Fold Proteins Using Energy Landscape Theory. *Isr. J. Chem.* 20 14, *54* (8-9), 1311–1337. https://doi.org/10.1002/ijch.201300145.
23. Zhang, Y. Progress and Challenges in Protein Structure Predicti on. *Curr. Opin. Struct. Biol.* 2008, *18* (3), 342–348. https://doi.org/ 10.1016/j.sbi.2008.02.005.
24. Berman, H. M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T. N .; Weissig, H.; Shindyalov, I. N.; Bourne, P. E. The Protein Data B ank. *Nucleic Acids Res.* 2000, 28 (1), 235–242. https://doi.org/10.1 093/nar/28.1.235.
25. Mani, H.; Chang, C. C.; Hsu, H. J.; Yang, C. H.; Yen, J.; Liou, J. Comparison, Analysis, and Molecular Dynamics Simulations of S tructures of a Viral Protein Modeled Using Various Computation

DOI: 10.36838/v7i3.87

al Tools. *Bioengineering* 2023, 10 (9), 1004. https://dx. doi.org/10.3390/bioengineering10091004.
26. Chen, J. N.; Jiang, F.; Wu, Y. Accurate Prediction for Protein-Peptide Binding Based on High-Temperature Molecular Dynamics Simulations. *J. Chem. Theory Comput*. 2022, *18* (9), 4307–4320. https://dx.doi.org/10.1021/acs.jctc.2c00743.
27. Heo, L.; Janson, G.; Feig, M. Physics-Based Protein Structure Refinement in the Era of Artificial Intelligence. Proteins Struct. Funct. *Bioinf.* 2021, *89* (8), 1181–1190. https://dx.doi.org/10.1002/prot.26161.
28. Audagnotto, M.; Czechtizky, W.; De Maria, L.; Käck, H.; Papoian, G.; Tornberg, L.; Tyrchan, C.; Ulander, J. Machine Learning/Molecular Dynamic Protein Structure Prediction Approach to Investigate the Protein Conformational Ensemble. *Sci. Rep*. 2022, *12*, 10123. https://dx.doi.org/10.1038/s41598-022-13714-z.
29. Audagnotto, M.; Czechtizky, W.; De Maria, L.; Käck, H.; Papoian, G.; Tornberg, L.; Tyrchan, C.; Ulander, J. Protein Structure Dynamic Prediction: A Machine Learning/Molecular Dynamic Approach to Investigate the Protein Conformational Sampling. *bioRxiv 2021*, 470731. https://dx.doi.org/10.1101/2021.12.01.470731.
30. Zhang, Y. Progress and Challenges in Protein Structure Prediction. *Curr. Opin. Struct. Biol*. 2008, *18* (3), 342–348. https://doi.org/10.1016/j.sbi.2008.02.005.
31. Dirac, P. A. M. *The Principles of Quantum Mechanics*; Clarendon Press: Oxford, 1981.
32. Aaronson, S. Quantum Computing Since Democritus; Cambridge University Press: United Kingdom, 2013.
33. Zhao, R.; Wang, S. A Review of Quantum Neural Networks: Methods, Models, Dilemma (arXiv:2109.01840). *arXiv 2021*. https://doi.org/10.48550/arXiv.2109.01840.
34. Shor, P. W. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. *Proc. Annu. Symp. Found. Comput. Sci.* 1994, 124–134. https://doi.org/10.1109/SFCS.1994.365700.
35. Audenaert, K. M. R.; Calsamiglia, J.; Muñoz-Tapia, R.; Bagan, E.; Masanes, L.; Acin, A.; Verstraete, F. Discriminating States: The Quantum Chernoff Bound. *Phys. Rev. Lett.* 2007, *98* (16), 160501. https://doi.org/10.1103/PhysRevLett.98.160501.
36. Jeswal, S. K.; Chakraverty, S. Recent Developments and Applications in Quantum Neural Network: A Review. Arch. *Comput. Methods Eng*. 2019, *26* (4), 793–807. https://doi.org/10.1007/s11831-018-9269-0.
37. Doga, H.; Raubenolt, B.; Cumbo, F.; Joshi, J.; DiFilippo, F. P.; Qin, J.; Blankenberg, D.; Shehab, O. A Perspective on Protein Structure Prediction Using Quantum Computers. *J. Chem. Theory Comput*. 2024, *20* (9), 3359–3378. https://doi.org/10.1021/acs.jctc.4c00067.
38. Onuchic, J. N.; Wolynes, P. G. Theory of Protein Folding. *Curr. Opin. Struct. Biol.* 2004, *14* (1), 70–75. https://doi.org/10.1016/j.sbi.2004.01.009.
39. Creighton, T. E. Proteins: *Structures and Molecular Principles*; W. H. Freeman And Company: New York, 1996.
40. Bravyi, S.; Cross, A. W.; Gambetta, J. M.; Maslov, D.; Rall, P.; Yoder, T. J. High-Threshold and Low-Overhead Fault-Tolerant Quantum Memory. *Nature* 2024, *627* (8005), 778–782. https://doi.org/10.1038/s41586-024-07107-7.
41. (1) Zambaldi, V.; La, D.; Chu, A. E.; Patani, H.; Danson, A. E.; Kwan, T. O. C.; Frerix, T.; Schneider, R. G.; Saxton, D.; Thillaisundaram, A.; Wu, Z.; Moraes, I.; Lange, O.; Papa, E.; Stanton, G.; Martin, V.; Singh, S.; Wong, L. H.; Bates, R.; Kohl, S. A.; Abramson, J.; Senior, A. W.; Alguel, Y.; Wu, M. Y.; Aspalter, I. M.; Bentley, K.; Bauer, D. L. V.; Cherepanov, P.; Hassabis, D.; Kohli, P.; Fergus, R.; Wang, J. De Novo Design of High-Affinity Protein Binders with AlphaProteo. arXiv September 12, 2024. https://doi.org/10.48550/arXiv.2409.08022.

Author

Quaid Shubin is a high school senior enrolled at Windward School. He plans to pursue his interests in Artificial intelligence and Computational Biology at university and is currently conducting further research in the field of computational biology.