BABEŞ BOLYAI UNIVERSITY, CLUJ NAPOCA, ROMÂNIA
FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

A Quantum Deep Learning Approach to the Protein Folding Problem

– Laboratory 4 –

Muscalagiu Anca Ioana

2022-2023

Abstract

The protein folding problem is one of the most challenging problems in current biochemistry and is an important problem in Bioinformatics. All current mathematical models of the problem are affected by intrinsic computational limits. The previous research offers very few approaches that make use of quantum computing to resolve this problem. In this paper we present a way of modeling of the protein structure prediction problem in order to reap the benefits of quantum programming in NP complete search problems. This paper aims to demonstrate the gain of using Quantum Deep Learning techniques in Bioinformatics, more specifically for solving one of the most difficult problems in this field. In order to validate the superior approach of Quantum Computing over the Classical one, the article presents a series of experiments both on a Quantum Simulator and a real Quantum Device, provided by IBM for open usage in quantum experiments. The data for testing our model is provided by DeepMind and EMBL's European Bionformatics Institute, the creators of AlphaFold [1], the most successfull Neural Network model designed for protein structure prediction. Experiments showed that,in comparison with the classical reinforcement learning approaches which showed better results on very particular smaller proteins like dipeptides, the new quantum deep learning based reinforcement learning method is much more suited for the protein folding problem in 2D and 3D dimensional lattice model when the number of amino-acids is larger.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Problem Domain

The determination of the three-dimensional structure of a protein, known as the Protein Folding Problem [2], from linear sequences of amino acids is one of the greatest challenges encountered by Bioinformatics. The prediction of protein structures having minimum energies represents an NP-hard problem, which requires a massive number of computations in order to reach an approximate solution. Unlike the structure of other molecules, proteins have complex structures that are very difficult to predict, therefore needing highly optimized algorithms for generating the three-dimensional structure of the protein. Its configuration after folding determines the protein's functionality within the organism and an incorrect fold could lead to potential diseases such as cancer, Alzheimer's, Huntington's or Parkinsons's disease. The possibility to predict the 3D structures after folding from its primary genetic structure could aid in numerous problems from medicine (including drug design, disease prediction, etc.) and genetic engineering (cell modeling, modification and improvement of the functions of certain proteins).

The paper compares different known approaches like full [4] or partial state reinforcement learning with quantum deep learning, hoping to achieve a polynomially faster time compared to the former solutions to the Protein Folding Problem. Other classical approaches using the Constraint Programming [5], evolutionary algorithms [6], Multi-agent systems [7] have been proven to be inferior to the reinforcement learning technique presented by AlphaFold [1], therefore this model serves as a basis for the improved quantum model we will present in this paper.

The main objective of this document is to further extend the applications of quantum deep learning

in Bioinformatics and to create an algorithm which offers a balanced trade-off between time and accuracy for prediction of the protein structure. The paper presents the outcomes of probably the first experiments with quantum reinforcement learning, setting up a basis in this type of approach for further research.

1.2 Paper structure and original contribution(s)

The research presented in this paper advances the theory, design, and implementation of the AlphaFold model with the benefits of quantum programmming: qubit superstates and entanglement using quantum circuits, obtaining an overall polynomial speedup over the classical approach in large sequences of amino-acids.

The second contribution of this report consists of building the first model of A Quantum Deep Learning Neural Network to solve this well-known problem, consisting a basis for further research in this direction. Our aim is to build an algorithm that works with the ab-initio model in 3D [8] and that can be easily extended to use different energy functions and to include further neglected aspects of the folding process, such as the side-chain fixation or the interaction of the protein with its environment.

The third contribution of this thesis consists of providing a comparison of the behaviour of quantum neural networks built in Qiskit in an ideal environment, on a quantum simulator and in a noise-prone one, on a real quantum device. In order to measure more accurately the average performance of a quantum device, several quantum computers were used for running our experiments.

The present work contains 19 bibliographical references and is structured in three chapters as follows.

The first chapter is an introduction in the folding protein problem, arguing its importance and difficulty. This section presents the current state of art in this domain, illustrating a short comparison between the existing classical methods and our approach.

The second chapter/section describes the investigated approach and the proposed experiments, along with our detailed methodology, the type of input data and the model used to describe the problem, the results of our experiments and a short retrospection on the output of our algorithm.

The chapter/section 5 details a summary of the outcomes of our model, its limitations and the further possible improvements.

Chapter 2

Related Work

2.1 Classical Approach

2.1.1 AlphaFold

One of the most important advances in solving the Protein Folding Problem is Alphafold, a model developed for the CASP14 assessment [9], significantly outperforming its competitors. In the following years, DeepMind, a department from Google dedicated for Deep learning, has greately improved the initial Alphafold model, reaching a 0.96 accuracy prediction for backbone chains and with an up to 0.95 accuracy for side chains. Using the predictions of this model, the AlphaFold Protein Structure Database was created and enriched with a large number of proteins, along with their amino-acid sequences and the corresponding folding, covering more than 190000 types of proteins.

As a result of its neural network architectures and training procedures, AlphaFold greatly improves the accuracy of structure prediction. The new architecture they present enables accurate end-to-end structure prediction by integrating multiple sequence alignment (MSA) features and pairwise features, along with a new output representation and associated loss. Iterative refinement of predictions is achieved using intermediate losses, masked MSA losses are used to train in conjunction with structural information, and self-distillation and self-estimates of accuracy are used to learn from unlabeled protein sequences. With the primary amino acid sequence and aligned sequences of homologues as inputs, AlphaFold directly predicts the 3D coordinates of all heavy atoms for a given protein.

The abstract model for the 3D backbone structure is represented as independent rotations and translations, each with respect to the global frame. These rotations and translations prioritize the

orientation of the protein backbone so that the location of the side chain of each residue is highly constrained within that frame.

In terms of accuracy and complexity, we expect similar results to Alphafold in medium-sized proteins for predicting their 3D backbone structure. Since the computational power we currently have in quantum computing is very limited (in terms of qubits) compared to the devices used by DeepMind and Google for large sequences prediction, at the moment a comparison study for proteins with more than 150 amino-acids is not feasible.

2.1.2 Reinforcement Learning

Apart from DeepMind's Alphafold, there are several approaches in the current literature that involve reinforcement learning, both model-based and model-free learning [4], [10]. Generally, in this type of solutions, the environment is a 2D or 3D lattice, with the agent moving one step at a time. Each step leads the agent into a new state (of certain coordinates in the space) and corresponds to the position of the amino-acid after the folding. After the agent reaches a final state (the whole protein was folded) the path generated by the actions chosen by the agent represents the structure of the protein after folding. In the solutions based on this type of approach, in the literature were applied multiple types of optimizations, such as using long short-term memory networks for the approximation phase in the reinforcement learning algorithm [10].

Based on this type of approach, our paper aims to extend the state of art regarding reinforcement learning usage in solving the protein folding problem. In comparison to the existent classical reinforcement learning solutions in the current literature, our quantum solution is expected to have a polynomial speed-up when tested on medium-sized to large amino-acid sequences. Our approach uses a similar modelling of the domain, with a slight difference in the approximation phase, which uses a quantum neural network to predict the best action to be taken at each step.

2.1.3 Constraint Programming

An important class of solutions presented in the literature for solving the Protein Folding problem is Constraint Programming, a programming approach used to describe and solve large classes of problems such as searching, combinatorial and planning problems. Because the search space is extensively large, the CSPs are generally modelled in a multi-agent environment, such that each variable is assigned to an agent who has control of its value, and agents must coordinate their choice of values so that

a global objective function is optimized, which is a set of constraints. The algorithms are generally asynchronous and distributed [11] or rely on local heuristics [5] in order to optimize the search strategy. One of the shortcomings observed in the literature regarding this approach is that for larger proteins, because of the greedy optimizations applied in the search strategy, the possible best energy is sometimes underestimated, which is not the case with our solution.

2.2 Quantum Approach

This section presents some of the advances in solving the protein folding using quantum approaches. The literature in this domain provides very few solutions using quantum programming, because of the fairly recent evolution of quantum computers and quantum resources.

2.2.1 Quantum Random Walks. QFold.

One of the main directions of quantum optimizations in this field were Quantum Random Walks as presented in [12]. The paper proposes using the prediction of AlphaFold as a starting point for a quantum MetropolisâHastings algorithm. The Metropolis algorithm is a Markov-chain Monte Carlo algorithm, that is, an algorithm that performs a random walk over a given graph. The algorithm is initialized with the prediction of Alphafold for a certain amino-acid sequence and enhances the result by slowly modifying the inverse temperature parameter such that the states with smaller energy become increasingly favoured by the random walk. Therefore we should end in the ground state of the system with high probability. The quantization of the algorithm is achieved through the use of a Szegedy quantum walk and quantum annealing. The abstract model used to represent the proteins takes into account are the dihedral angles of the side-chains as well as the environment of the protein. The article presents a comparison between the Classical Random Walks solution and their quantum-based approach, reaching a polynomial speedup for medium-sized amino-acid sequences. In comparison with our approach which uses the lattice-based model, they present a more-refined representation, but at the cost of increased complexity and resource consumption in terms of time and qubits, therefore making our approach more sustainable in the present state of quantum computers, which have a fairly limited number of qubits available. Another difference in the two approaches is the fact that the paper uses another existent model, Alphafold, while or model is completely developed by us, being inspired form the current state of art. One of the aim of this research is to compare and present the superiority of quantum algorithms, particularly in the protein folding problem, demonstrating a polynomial speedup in random walks, similar to our discovery regarding Quantum Neural Networks.

2.2.2 Quantum/classical variational algorithms

Another class of algorithms used for protein folding predicition are variational algorithms. One of the most important results in this direction is achieved by [13], which presents the problem as a quantum alternating operator ansatz, a member of this class. These hybrid algorithms consist of a fixed-length sequence of parametrizable quantum gates that are executed on a quantum computer whilst the gate parameters are variationally optimized by a classical computer. The principal similiarity of the approach presented in this paper is the abstract model used, the cubic lattice-based model. Apart from this, the article uses the same energy function presented by us and encodes the folded protein in an identical manner, through a path constructed by taking one of the 6 actions (left, right, up, down, front, back) repeatedly. These actions are encoded using Hadamard gates, in a similar fashion with our possible actions in the Quantum Reinforcement learning model. Unlike in our research, they provide an approach that does not have a classical equivalent, but solely relies on the specific properties of quantum gates. From the performance point of view, our approach surpasses the QAOA circuits used in the variational algorithm, providing a similar approximation of the energy function and accuracy of the 3D protein configuration generated.

Chapter 3

Investigated approach

3.1 Formal Definition of our Model

3.1.1 The Hydrophobic-Polar Model in 3D

The paper approaches the protein folding problem using a well-known class of abstract models for proteins : the lattice-based models in 3D. It describes the possible positions of protein's amino acids and their free energy value. Proteins have minimum free energy in their stable state, and our aim is to find a conformation with minimum free energy value. Amino acids are classified into two categories, hydrophobic, denoted by H, and hydrophilic (polar), denoted by P. The input sequence can be defined formally in the following way:

$$A = a_1a_2a_3..a_n,\ where\ a_i \in \{H, P\}, 1 \leq i \leq n$$

The goal of the protein folding problem in the HP model is to find the conformation C, which maps each amino-acid to a point in space, such that the amino-acids form a path (2 neighbours in the sequence will also be neighbours in the 3D spaced) and the protein energy function value is minimum. In order to define this mapping, we denote:

$$B = \{A = a_1a_2a_3..a_n|\ a_i \in \{H, P\}, 1 \leq i \leq\ n,\ n \in N\}$$

$$S = \{G = (x_i, y_i, z_i)|x_i, y_i, z_i \in R, 1 \leq i \leq n\}$$

$$C : B \rightarrow S$$

$$A = a_1a_2a_3..a_n,\ \rightarrow \{G = (x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3), .., (x_n, y_1, z_1)\},$$

$$,1 \leq i,j \leq n,\ with\ |i-j| = 1 \Rightarrow |x_i - x_j| + |y_i - y_j| + |z_i - z_j| = 1$$

In order for the protein to have a valid conformation, the path must not have any overlapping amino-acids in the 3D space:

$$1 \leq i,j \leq\ n,\ with\ i \neq j \Rightarrow (x_i, y_i, z_i) \neq (x_j, y_j, z_j)$$

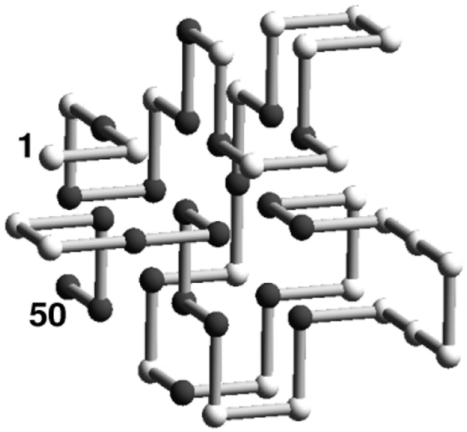

Figure 3.1: 3D Lattice Model of a protein with 50 amino-acids as described in [2]

The energy function in the HP model shows that amino acids have a tendency to form a hydrophobic core; therefore, a value equal to -1 added to the energy function value for each two hydrophobic amino acids that are mapped as neighbors in the lattice by the E function and are not consecutive in the initial amino acids sequence. The energy function can be calculated with the following formula:

$$e : \{1,2,3,....,n\} \times \{1,2,3,....,n\} \rightarrow \{-1,0\}$$

$$e(i,j) = \begin{cases} -1 & \mathrm{p}_i = p_j = H\ and\ |x_i - x_j| + |y_i - y_j| + |z_i - z_j| = 1 \\ 0 & otherwise \end{cases}$$

$$where, 1 \leq i,j \leq\ n,\ |i-j| \geq 2$$

$$E(C) = \sum_{,1 \leq i,j-2 \leq n} e(i,j)$$

Our aim is to find a conformation C whose energy E(C) is minimum using the methods presented in the section below.

3.1.2 Quantum Deep Reinforcement Learning

The DQN (Deep QNetwork) is a model-free, off-policy deep reinforcement learning algorithm that uses a neural network to approximate the Q-function from the basic Q-learning algorithm. In reinforcement learning scenarios an agent is trained to maximize a scalar reward by acting within an environment. The agent learns solely from its interactions with the environment without any direct instructions. At each time step t the agent observes a state s from the set of possible states S. For the DQN algorithm, the agent then picks an action at from its discrete action set A following an k-greedy policy. That is, with probability k, belonging to [0, 1], the agent performs a random action at and with probability 1-k the best action according to the current estimation of the Q-function:

$$a_t = argmax_a Q_\theta(s_t, a)$$

The Q-value is a function assigning a value to each state and action combination and is approximated by a neural network with parameters θ. After performing a_t in the environment, the agent observes a new state s_{t+1} and receives a scalar reward $r_{t+1} = r(s_t, a_t, s_{t+1})$. The interaction $(s_t, a_t, r_{t+1}, s_{t+1})$ is stored in a replay buffer from which at each time stamp, a mini-batch of interactions is sampled to update Q_θ with stochastic gradient descent, minimizing the loss.

In order to introduce the quantum algorithm behind QNNs we will make a short introduction into quantum states notations. The fundamental objects in quantum computing are qubits, which are the analog to bits in classical computing. Unlike classical bits, which can be in one of the two states 0 and 1, a qubit can be in a state, which is a linear combination of those states. Using the bra-ket notation, a qubit state $|s\rangle$ can be written as:

$$|s\rangle = \alpha|0\rangle + \beta|1\rangle, \ with \ \alpha, \beta \in C, \ |\alpha^2| + |\beta^2| = 1$$

The $|0\rangle$ and $|1\rangle$ notations are basis states of the underlying singlequbit Hilbert Space. During the probabilistic measurement process, the qubit will collapse to one of the two basis states, and $|\alpha|^2 \ and \ |\beta|^2$ can be interpreted as the probabilities for the respective basis states. Before the measurement, the state can be modified by applying quantum gates U. , which are formally described by unitary operators:

$$|s'\rangle = U|s\rangle$$

This formulation can be extended to multi-qubit systems by preparing an n-qubit quantum register. For quantum computers, this is commonly initialized in its computational basis state $|0\rangle^{\oplus n}$.

Variational quantum algorithms are a promising method to implement algorithms on current and near-term quantum computers as they are well suited for systems with a relative small number of qubits, noisy operation and limited coherence times [14]. Their basic principle of operation is the combination of a parameterized quantum circuit whose parameters are adjusted by a classical optimizer towards the desired outcome while evaluating the quantum circuit with adjusted parameters at each optimization step [15]. Using this method, we can create the equivalent of a Artificial Neural Network in Quantum Computing.

A parameterized quantum circuit (PQC) is a series of quantum gates U(θ, x) which is applied to the computational basis of n qubits $|0\rangle^{\oplus n}$. These gates are parameterized by variational parameters θ and classical input data x. The measured expectation value of the quantum computation can be interpreted as the computation of a parameterized function $f_O\theta, x$ depending on the observable, circuit parameters and input. The parameters θ are tuned with an appropriate method to fit a target function. Several analytic and numeric methods allow for calculating gradients of quantum circuits with respect to their parameters.

Another important aspect of a PQC is the encoding of the data in order to provide input to the circuit. Various encoding methods for quantum machine learning tasks have been suggested but recent results on the expressiveness of quantum circuits emphasize the advantages of repeated encodings, also referred to as data reupload. PQCs with data re-upload have layers which consist of data encoding unitaries $U_{in}(x_l)$ in each layer l. The circuit starts with parameterized unitaries $U(\theta_l)$ applied on the n-qubit register $|0\rangle^{\oplus n}$ followed by a sequential implementation of the layers.

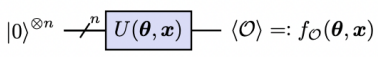

Figure 3.2: Basic principle of a parameterized quantum circuit (PQC) as defined in [3]

3.2 Methodology

The research plan adopted for this project requires a methodology that adopts both theoretical analysis and practical design, achieved through implementation and experimentation on both simulators and real-life quantum computers. In order to have the possibility to experiment on both environments we used Qiskit, an open-source SDK for working with quantum programming.

From an algorithmic point of view the article approaches the problem using reinforcement learning. The environment of the agent is the lattice itself, while each state s is identified through the type of the current amino acid, its position in the lattice, and the current amino acidâs order in the initial amino acids sequence (its number). The action space includes four actions, and each action calls each amino acidâs position from the previous position in the lattice. The possible actions are as up, down, right, and left, which are encoded with U, D, R, and L, respectively. Our reward function was defined based on the fact that hydrophobic amino acids have a tendency to form a hydrophobic core in their stable state. Therefore, we defined the reward function as â1, if each two hydrophobic amino acids are neighbors in the lattice which indicates that the current amino acid is up, down, right, or left of the previous hydrophobic amino acid, or a small positive number otherwise . Also, they should not be in consecutive order in the initial amino acids sequence. In our algorithm, the lattice is encoded as a 3D multidimensional array, using -1 for slots in which a hydrophobic amino-acid appears, 1 for hydrophilic ones, respectively 0 for an empty slot. The DRL algorithm is proposed to learn optimal policy functions that represent the maximum cumulative reward. In optimal policy, the agent tries to take the best action in any state as specified by it, the quality of the output of that action taken in state s being measured through the Q value, which represents a type of cumulative rewards. In order to approximate this values, we will be using a quantum deep neural network as an estimator. The Q-value is given as the trainable parameter to the QNN, which in our case will be a quantum observable value. We calculate the error of the aproximation given by the neural network using Bellman's equation error. The training loss function has the aim to minimize this error after each epoch of training has passed. Forward passing the loss function minimizes the errorâs rate, and then backward passing the updated gradient values corrects the weights in the QNN and completes the training process. The interface for QNNs in Qiskit provides us with both functions in order to train our model. For the learning process we start with a larger learning rate of 0.001, favouring exploration over exploatation in the beginning, but switching to a smaller learning late after a few epochs have passed.

3.3 Protein Datasets. Benchmark amino-acid sequences for 3D lattice-based models.

The data we used in our experiments contains a set of standard 3D HP benchmark instances with different sizes that have been widely used in the literature for solving the PFP in the 3D cubic lattice model. The dataset was chosen with the purpose of easily comparing our algorithm on the same data

as the most important research papers using the 3D lattice-based model. We provide in this section a short list of the most important proteins that we used in our experiments:

Sequence	Length
HPHPPHHPHPPHPHHPPHPH	20
HHPPHPPHPPHPPHPPHPPHPPHH	24
PPHPPHHPPPPHHPPPPHHPPPPHH	25
PHPHPHHHPPHPHPPPPPPPPPPHHP	27
PHHPPPPPPPPPPHHPPHHPPHPPHPH	27
HHHHPPPPPHPPPPHHHPPPPPPPPH	27
HPPPPPPPPPPPHPHPPPPPPPPHPHH	27
PPPPPPPHHHPPPHPHHPPPHPPHPPP	27
PPPPPHHPHPHPHPHPPHHPHHPHPPP	27
PPPHHPPHHPPPPPHHHHHHHPPHHPPPPHHPPHPP	36
PPHHHPHHHPPPHPHHPHHPPHPHHHHPHPPHHHHHPHPHHPPHHP	46
PPHPPHHPPHHPPPPPHHHHHHHHHHPPPPPPHHPPHHPPHPPHHHHH	48
HHPHPHPHPHHHHPHPPPHPPPHPPPPHPPPHPPPHPHHHHPHPHPHH	50
PHPHHHPHHHPPHHPHPHHPHHHPHPHPHHPPHHHPPHPHPPPPHPPHPPHHPPHPPH	58
PPHHHPHHHHHHHHPPPHHHHHHHHHHPHPPPHHHHHHHHHHHHPPPPHHHHHHPHHPHP	60

Table 3.1: Protein Benchmark Data

3.4 Results

The same experiments were run on both environments mentioned above, the Aer Simulator provided by Qiskit and with several quantum devices that were available at that time. Because the devices were differing in terms of noise and quantum coherence, we considered an average of outputs of the experiments on all quantum devices as our thesis final result.

3.4.1 Validation

In order to compare our model with other existing solutions, we have to compare the energy value of the protein after folding obtained within our experiments with results of other studies. The better performing model is the one closest to the labelled energy for the respective protein that is recorded in the AlphaFold Protein Structure Database, which is verified by experts in the biological field. Besides the energy value, in cases of correctly predicted values by both compared models, the generated structures are also analysed by measuring the distance of each amino-acid mapping in 3D to its actual place in the configuration presented in the Alphafold Database.

3.4.2 Discussion

As a statistical output of our experiments, the results showed that for dipeptides and tetrapeptides the accuracy of the prediction of structure and energy is very similar to the previous models, but our quantum model performed slower than the classical version of our model, while for average size proteins, with up to 60 amino-acids in the primary structure, the accuracy was slightly greater than comparable models such as Alphafold [1] and the time was significantly improved. Our experiments fall short when trying to run our QNN on greater sequences because of the lack of resources in terms of qubits on the real quantum devices and in terms of processing power when running on a simulator. Although our resources are limited at the moment, along with the development of quantum devices in the future, our model could be able to solve increasingly complex protein structures.

Chapter 4

Case Study

4.1 Small Data

In this section we will discuss one of our first experiments in an initial model of the networks having 3 layers.

4.1.1 Environment setup

The environment for the reinforcement learning problem was created using the gym library along with KerasRl for integrating the artificial neural network with our custom RL agent. For creating the QNN we have used the Qiskit library for Machine Learning and for creating its Classical equivalent we have used an ANN from Keras, having 3 Dense layers. The characteristics of the custom environment created are:

- *the action space* consists of a set of 6 discrete values, representing the movement in one of the 6 directions: left, right, back, front, down, up, which are represented using tuples of (dx, dy, dz), deplicting the differences on each axis in the 3d space
- *the observation space* is represented by a 3D space, having

$$x, y, z \in [-n, n],\ where\ n\ =\ length\ of\ the\ protein\ sequence$$

- *the rewards* given for the agent are the following:

$$r(\pi_k|s_1, \pi_1, ..., \pi_k - 1) = \begin{cases} 0.01 & \text{i}f\ a_\pi\ is\ not\ valid \\ -E_\pi & ,if\ k = n - 1 \\ 0.1 & \text{o}therwise \end{cases}$$

We train our agent over 50000 steps in the created environment and we test it over 100 episodes and compute a mean of the rewards obtained by our agent.

4.1.2 Data

The data used for this experiment was a small dataset of short amino-acid sequences, as deplicted in the table below :

Protein	Sequence	Length
s1	HPHPPHHPHPPHPHHPPHPH	20
s2	HHPPHPPHPPHPPHPPHPPHPPHH	24
s3	PPHPPHHPPPPHHPPPPHHPPPPHH	25
s4	PHPHPHHHPPHPHPPPPPPPPPPHHP	27
s5	PHHPPPPPPPPPHHPPHHPPHPPHPH	27

Table 4.1: Small dataset used for First Experiments

4.1.3 Result Analysis

As we can observe our model can accurately predict the values for small and medium-sized proteins with a high accuracy when compared to the output of other models based on the 3D ab-initio lattice model, such as [6]. In the following table we present the proteins we tested within our experiments, the energy predicted by our reinforcement learning model in comparison with the energy predicted by the model illustrated in [6].

Protein	Length	Expected Energy	Obtained Energy
s1	20	-11	-12
s2	24	-13	-13
s3	25	-9	-9
s4	27	-9	-9
s5	27	-11	-11

Table 4.2: Resulted Energy Values Comparison for First Experiments

While running multiple executions of our experiments our agent generated multiple configurations with the same energy value, all predicted structures being considered a correct folding, as factors such as particularities of the environment of the protein are not being taken into account in our model, but this can provide an open improvement direction of our model in further research. Some of the predictions made by our algorithm for s1 and s2 proteins represented in the 3D cubic lattice are:

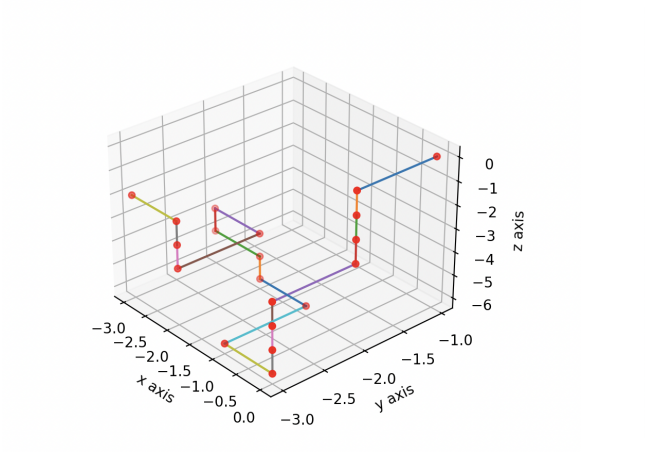

Figure 4.1: s1 Protein after Folding

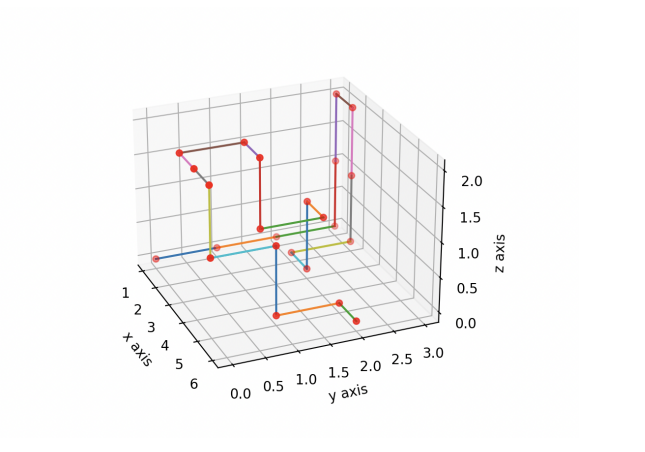

Figure 4.2: s2 Protein after Folding

These results were obtained after training of our agent across 50000 steps in 2500 episodes. The training process was splitted in 5 intervals, on which we measured the average reward obtained by our agent, the minimum and the maximum reward per interval.

Interval	Average Reward	Maximum Reward	Minimum Reward
1	-3.092	-10.750	1.000
2	-2.918	-10.500	1.500
3	-2.951	-9.000	1.000
4	-3.035	-8.875	1.000
5	-2.735	-8.575	1.000

Table 4.3: Rewards obtained by our agent in training Phase

Chapter 5

Conclusion and future work

Firstly, our results provide evidence for the fact that quantum programming can provide polynomial optimisations in NP-complete search problems. This claim is also confirmed in [12]. The paper presents similar results with our thesis, demonstrating that Quantum Random Walk is a consistent improvement over the Random Walk, with a O(sqrt(n)) complexity speed-up. These findings provide a clear answer to a popular question in the Bionformatics field: how can we improve the structure prediction time without losing accuracy? While the majority of papers incline towards distributed approaches or better heuristics, our thesis shows that the answer to this question might be an entirely different shift in our perspectives: Quantum Mechanics.

Secondly, our paper brings value to the research in this area, being one of the first few quantum approaches in solving the Protein Folding Problem. More importantly, our model provides the starting basis for introducing QNNs to predict protein structures. The aim of our article is to provide an open source quantum model that can be further extended and improved by other researches. Since we currently use a fairly simple abstract model for the protein structures, a possible improvement would consist in using a more detailed version, introducing aspects such as the angles between 2 amino-acids, secondary chains and so on.

Although quantum algorithms provide a very efficient approach for solving NP-complete problems, they have numerous limitations, being exceedingly difficult to engineer, build and program. As a result, they are crippled by errors in the form of noise, faults and loss of quantum coherence, occasionally leading to loss and corruption of data. Therefore, one of the shortcomings of our research was the lack of resources in terms of qubits at the certain moment, deficiency which could be overcomed with the future improvements on quantum devices.

Bibliography

[1] Richard Evans, Michael OâNeill, Alexander Pritzel, Natasha Antropova, Andrew Senior, Tim Green, Augustin Žídek, Russ Bates, Sam Blackwell, Jason Yim, et al. Protein complex prediction with alphafold-multimer. *BioRxiv*, pages 2021–10, 2022.

[2] Ken A Dill, S Banu Ozkan, M Scott Shell, and Thomas R Weikl. The protein folding problem. *Annual review of biophysics*, 37:289, 2008.

[3] Dirk Heimann, Hans Hohenfeld, Felix Wiebe, and Frank Kirchner. Quantum deep reinforcement learning for robot navigation tasks. *arXiv preprint arXiv:2202.12180*, 2022.

[4] Gabriela Czibula, Maria-Iuliana Bocicor, and Istvan-Gergely Czibula. A reinforcement learning model for solving the folding problem. *International Journal of Computer Technology and Applications*, 2(01), 2011.

[5] Abu Dayem Ullah and Kathleen Steinhöfel. A hybrid approach to protein folding problem integrating constraint programming with local search. *BMC bioinformatics*, 11(1):1–9, 2010.

[6] Cheng-Jian Lin and Shih-Chieh Su. Protein 3d hp model folding simulation using a hybrid of genetic algorithm and particle swarm optimization. *International Journal of Fuzzy Systems*, 13(2), 2011.

[7] Luca Bortolussi, Agostino Dovier, and Federico Fogolari. Multi-agent simulation of protein folding. In *Proceedings of the first international workshop on multi-agent systems for medicine, computational biology, and bioinformatics*, 2005.

[8] Ken A Dill, Sarina Bromberg, Kaizhi Yue, Hue Sun Chan, Klaus M Ftebig, David P Yee, and Paul D Thomas. Principles of protein foldingâa perspective from simple exact models. *Protein science*, 4(4):561–602, 1995.

[9] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[10] Reza Jafari and Mohammad Masoud Javidi. Solving the protein folding problem in hydrophobic-polar model using deep reinforcement learning. *SN Applied Sciences*, 2(2):1–13, 2020.

[11] Rolf Backofen and Sebastian Will. A constraint-based approach to structure prediction for simplified protein models that outperforms other existing methods. In *International Conference on Logic Programming*, pages 49–71. Springer, 2003.

[12] Pablo Antonio Moreno Casares, Roberto Campos, and Miguel Angel Martin-Delgado. Qfold: quantum walks and deep learning to solve protein folding. *Quantum Science and Technology*, 7(2):025013, 2022.

[13] Mark Fingerhuth, Tomáš Babej, et al. A quantum alternating operator ansatz with hard and soft constraints for lattice protein folding. *arXiv preprint arXiv:1810.13411*, 2018.

[14] Marco Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, et al. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, 2021.

[15] Jarrod R McClean, Jonathan Romero, Ryan Babbush, and Alán Aspuru-Guzik. The theory of variational hybrid quantum-classical algorithms. *New Journal of Physics*, 18(2):023023, 2016.

[16] Martin Mann and Rolf Backofen. Exact methods for lattice protein models. *Bio-Algorithms and Med-Systems*, 10(4):213–225, 2014.

[17] Andrea Skolik, Sofiene Jerbi, and Vedran Dunjko. Quantum agents in the gym: a variational quantum algorithm for deep q-learning. *Quantum*, 6:720, 2022.

[18] Rajat Kumar Vishwakarma. Protein folding using quantum computers. *Authorea Preprints*, 2022.

[19] Mikita Misiura, Raghav Shroff, Ross Thyer, and Anatoly B Kolomeisky. Dlpacker: deep learning for prediction of amino acid side chain conformations in proteins. *Proteins: Structure, Function, and Bioinformatics*, 90(6):1278–1290, 2022.

[20] Berat Doğan and Tamer Ölmez. A novel state space representation for the solution of 2d-hp protein folding problem using reinforcement learning methods. *Applied Soft Computing*, 26:213–223, 2015.

[21] Nathan Wiebe, Ashish Kapoor, and Krysta M Svore. Quantum deep learning. *arXiv preprint arXiv:1412.3489*, 2014.

[22] Leonardo Alchieri, Davide Badalotti, Pietro Bonardi, and Simone Bianco. An introduction to quantum machine learning: from quantum logic to quantum deep learning. *Quantum Machine Intelligence*, 3(2):1–30, 2021.

[23] Abu Kamruzzaman, Yousef Alhwaiti, Avery Leider, and Charles C Tappert. Quantum deep learning neural networks. In *Future of Information and Communication Conference*, pages 299–311. Springer, 2019.

[24] Yoav Levine, Or Sharir, Nadav Cohen, and Amnon Shashua. Quantum entanglement in deep learning architectures. *Physical review letters*, 122(6):065301, 2019.

[25] Zhenwei Yang and Xiangdong Zhang. Entanglement-based quantum deep learning. *New Journal of Physics*, 22(3):033041, 2020.