Georgia State University

ScholarWorks @ Georgia State University

Computer Science Dissertations | Department of Computer Science

12-16-2020

A Large Scale Deep Learning Application for Protein Structure Prediction and Analysis

Chinua Umoja

Follow this and additional works at: https://scholarworks.gsu.edu/cs_diss

Recommended Citation

Umoja, Chinua, "A Large Scale Deep Learning Application for Protein Structure Prediction and Analysis." Dissertation, Georgia State University, 2020.
doi: https://doi.org/10.57709/20472584

This Dissertation is brought to you for free and open access by the Department of Computer Science at ScholarWorks @ Georgia State University. It has been accepted for inclusion in Computer Science Dissertations by an authorized administrator of ScholarWorks @ Georgia State University. For more information, please contact scholarworks@gsu.edu.

A LARGE SCALE DEEP LEARNING APPLICATION FOR PROTEIN STRUCTURE PREDICTION AND ANALYSIS

by

CHINUA T. UMOJA

Under the Direction of Robert Harrison, PhD

ABSTRACT

Proteins are such a vital piece of the scientific community's understanding of molecular interactions in organisms, which is why the accurate analysis of proteins are such a pivotal piece of drug and toxin design. In this research we introduce the foundation of a software suite of interconnected novel tools that use machine learning and deep learning tools. These tools will provide not only insights and connected patterns about the proteins that they study, but also a new data set of information that can be used in future research. This tool would enable analysis from several different inputs, such as: the primary protein structure, FASTA, DSSP, or PDB files. Also, in this research we introduce a database of results that would use similar analysis techniques as social media analysis tools to attempt to analyze the interconnected nature of proteins in a meaningful way. This research is not intended to be considered a final product but a foundation and proof of concept. The favorable results found by the initial tools being used as inspiration along with a clear path of the future to build a more complete tool with the goals of one day being made publicly so the information can grow to its full potential.

INDEX WORDS: Protein Structure Prediction, Restricted Boltzmann Machines, Genetic Algorithm, Shape Prediction, Protein Footprint, Weisfeiler-Lehman Graph Kernels, Protein-Protein Interaction Network, Fuzzy Determination

A LARGE SCALE DEEP LEARNING APPLICATION FOR PROTEIN STRUCTURE PREDICTION AND ANALYSIS

by

CHINUA UMOJA

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

in the College of Arts and Sciences

Georgia State University

2020

Copyright by
Chinua Umoja
2020

A LARGE SCALE DEEP LEARNING APPLICATION FOR PROTEIN STRUCTURE PREDICTION AND ANALYSIS

by

CHINUA UMOJA

Committee Chair: Robert Harrison

Committee: Rajshekhar Sunderraman
Irene Weber
Yanqing Zhang

Electronic Version Approved:

Office of Graduate Studies
College of Arts and Sciences
Georgia State University
December 2020

DEDICATION

I dedicate this to my family, my community, and everyone or anyone who needs this as inspiration to take the steps they need to get their goals accomplished.

ACKNOWLEDGEMENTS

There are so many people that are responsible for allowing me to be in this position, and if any one was left out of this acknowledgement, know that you are still appreciated. First and foremost, I would like to thank my family, I would have not been able to do anything if not for your love, motivation and pressure you have provided. You all challenge and inspire me to try to be the best version of myself. Dr. Robert Harrison, thank you for your patience and guidance. Not only have I appreciated your mind and knowledge base, but your humor and understanding. You are truly a one-of-a-kind individual and I can't imagine having anyone else as an advisor (or any advisor being nearly as patient with me). Dr. Irene Weber and her lab, thank you for opening your doors to me and allowing me the opportunity to learn so much outside of what had been my comfort zone. I really enjoyed being in a wet lab and crystallizing proteins. To my lab mates and co-writers James Torrance, Dr. Erin Elizabeth-Durham, Dr. Andrew Rosen, Dr. Brendan Benshoof, Michael J. McDermott, Dhara Shah, Chris Freas and their families, thanks for the laughs, the journey, the reading recommendations and the community you provided me. In closing, I also want to acknowledge Morehouse College Math and Computer Science departments (especially Dr. Peng and Dr. Ng), Molecular Basis of Disease fellowship, Dr. Raj, Tammie Dudley, Venette Rice, Celena Pittman, Adrienne Martin, Dr. Alex and the many friends I made with his lab.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . . **v**

LIST OF TABLES . . . . **ix**

LIST OF FIGURES . . . . **x**

LIST OF ABBREVIATIONS . . . . **xii**

PART 1 INTRODUCTION **1**

1.1 An Overview of Computational Protein Analysis 1

1.2 Genetic Algorithms 3

1.2.1 Sequence Alignment 3

1.2.2 Protein Structure Prediction 4

1.2.3 Molecular Interactions of Proteins 5

1.3 Tool Proposal 6

1.3.1 Binding Site Prediction 7

1.3.2 Tertiary Structure Prediction 9

1.3.3 Secondary Structure Prediction and Determination 10

1.3.4 Protein-Protein Interaction Analysis 12

1.4 Research Outline 13

PART 2 SHAPE DETERMINATION BASED OFF OF PREDICTED STRUCTURE **15**

2.1 Introduction 15

2.2 Methodology 17

2.2.1 Data Import 18

2.2.2 Molecule Surface Creation 18

2.2.3 Shape Determination 20
2.2.4 Charge Definition 23
2.2.5 Fuzzy Ranking and Comparison 25
2.3 Experiment 26
2.4 Future Work 27
2.5 Conclusion 29

PART 3 HISTORY OF MACHINE LEARNING IN PROTEIN STRUCTURE PREDICTION **31**
3.1 Genetic Algorithms 32
3.1.1 GAs Model Selection in Protein Structure Prediction 34
3.1.2 Common Methodology 38
3.2 Supervised Machine Learning Algorithm 42
3.2.1 Artificial Neural Networks 42
3.2.2 ANNs in Protein Structure Prediction 44
3.2.3 Support Vector Machines 45
3.3 Fuzzy Application 47
3.3.1 Fuzzy Logic 48
3.3.2 Fuzzy SVMs 48
3.3.3 Adaptive-Network-Based Fuzzy Inference Systems 49
3.3.4 Fuzzy Decision Trees 50
3.4 Conclusion 51

PART 4 HYBRID EVOLUTIONARY ALGORITHM FOR TERTIARY STRUCTURE PREDICTION **53**
4.1 Population Generation and Baseline Secondary Structure Prediction 54
4.2 Generating New Populations Through Crossover and Mutation 56
4.3 Fitness Test Application 57

4.4 Secondary Structure Classification 58

PART 5 RESTRICTED BOLTZMANN MACHINE APPLICATIONS TO SECONDARY STRUCTURE PREDICTION . . . 60

5.1 Boltzmann Machine 60

5.2 Restricted Boltzmann Definitions 61

5.3 Clique Restricted Boltzmann Machine 64

5.4 Restricted Boltzmann Protein Analysis 65

5.5 Results and Conclusions 67

PART 6 WEISFEILER-LEHMAN RELATIONSHIP GRAPHS . 68

6.1 Preliminary Definitions 69

6.1.1 Graphs and Multi-Graphs 70

6.1.2 Graph Kernels 71

6.1.3 Weisfeiler-Lehman Graph Kernels 72

6.2 Weisfeiler-Lehman Relationship Graph Kernel 72

6.3 Experiments 73

PART 7 CONCLUSION 74

7.0.1 Future Works 74

7.0.2 Outside Questions and Application 75

REFERENCES 76

LIST OF TABLES

Table 2.1 Atoms and Van der Waals radius 20

Table 2.2 Top ten results based on fuzzy membership results, with combined shape and affinity determination 27

Table 2.3 Top ten results based on fuzzy shape comparison values 28

Table 2.4 Top ten results based on binding affinity values 28

LIST OF FIGURES

Figure 1.1 Primary sequence of protein 1AIX [1]. 2
Figure 1.2 Secondary structure of protein 1AIX. 2
Figure 1.3 A tertiary mesh of the surface of protein 1AIX. 3
Figure 1.4 Lock and key representation of molecular docking. Left: blue represents a small molecule or substrate; green represents a protein or enzyme. Right: small molecule bound in a protein. 5
Figure 2.1 PDB source file format 18
Figure 2.2 Relational database to hold the atom and protein data 19
Figure 2.3 This is a representation of the dense model of the surface of the 3TTP protein (multi-resistant HIV-1 protease in complex with darunavir) with a distance of 0.01 of angstrom apart. 21
Figure 2.4 This is a representation of the sparse model of the surface of the 3TTP protein (multi-resistant HIV-1 protease in complex with darunavir) with a distance of 0.1 of angstrom apart. (PDB code 3TTP) . . 21
Figure 2.5 A visual representation of the probe step of ShapeDoc, around the 3TTP protein. 23
Figure 3.1 A figure of a protein's chemical structure, *1AIX bonds*. 32
Figure 3.2 An example of the possible moves in the HH model (top), as well as a sequence with 48 monomers [HHHHPHHPHHHHHPPHPPHHPPH-PPPPPPHPPHPPPHPPHHPPHHHPH] from [2]. 35
Figure 3.3 A molecule used to depict the ϕ label phi , ψ labeled psi, and ω dihedral angles. 37
Figure 3.4 The final product of an all atom model would look similar to this. This is the actual backbone of protein 1AIX. 37
Figure 3.5 Basic structure of an ANN. 43

Figure 3.6 Basic structure of a SVM. 46

LIST OF ABBREVIATIONS

- AMMP - Another Molecular Mechanics Program
- ANFIS - Adaptive Neural Fuzzy Inference System
- ANN - Artificial Neural Network
- BM - Boltzmann Machine
- cRBM - Clique Restricted Boltzmann Machine
- DSSP - Database of Secondary Structure Assignments
- EA - Evolutionary Algorithm
- GA - Genetic Algorithm
- FDT - Fuzzy Decision Tree
- HH - Hydrophobic-Hydrophobic
- HP - Hydrophobic-Polar
- Kcal - Kilocalorie
- PDB - Protein Database
- PSP - Protein Structure Prediction
- PSSM - Position Specific Scoring Matrix
- QSAR - Quantitative Structure-Activity Relationship
- RBM - Restricted Boltzmann Machine
- SDM - Spatial Data File

- SVM - Support Vector Machine
- WL - Weisfeiler-Lehman
- WLRGK - Weisfeiler-Lehman Relationship Graph Kernel

PART 1

INTRODUCTION

Since their introduction into early education, proteins are coined as ”the building blocks of life.” Proteins are responsible for the most basic functions that occur in an organism, and they extend well beyond the layman’s understanding of” muscle development.” In the academic world, the study of proteins is a significant area of research. It is the physical sciences of biology and chemistry and the theoretical and mathematical disciplines of computer science and statistics. These research areas extend to concepts like low energy protein structure confirmations, protein functions and molecular interactions, protein folding patterns and shape, etc. Although there has been an ample number of research projects focusing on the analysis of proteins, developing a tool that can speedily and accurately give you valuable information or draw new conclusions about individual proteins remains a large area of need in the aforementioned fields.

1.1 An Overview of Computational Protein Analysis

Proteins are a class ofcompounds that consist of large molecules composed of one or more long chains of amino acids; where amino acids are a simple organic compound containing both a carboxyl (–COOH) and an amino (–NH_2) group. Proteins are responsible for the most basic functions that occur in an organism, which include but are not limited to energy production, waste and toxin removal, movement, and cellular communication. Individual proteins are defined by their structures which are represented in the following classes:

- *Primary Structure* - The sequence consisting of any number of the 20 different standard amino acids used by cells for protein construction. Figure 1.1.

- *Secondary Structure* - Distinct local structural conformations that can occur in the protein, these are dependent upon hydrogen bonding. Figure 1.2.

```
IVEGSDAEIGMSPWQVMLFRKSPQELLCGASLISDRWVLT
AAHCLLYPPWDKNFTENDLLVRIGKHSRTRYERNIEKISM
LEKIYIHPRYNWRENLDRDIALMKLKKPVAFSDYIHPVCL
PDRETAASLLQAGYKGRVTGWGNLKETWTANVGKGQPSVL
QVVNLPIVERPVCKDSTRIRITDNMFCAGYKPDEGKRGDA
CEGDSGGPFVMKSPFNNRWYQMGIVSWGEGCDRDGKYGFY
THVFRLKKWIQKVIDQFGE
```

Figure 1.1. Primary sequence of protein 1AIX [1].

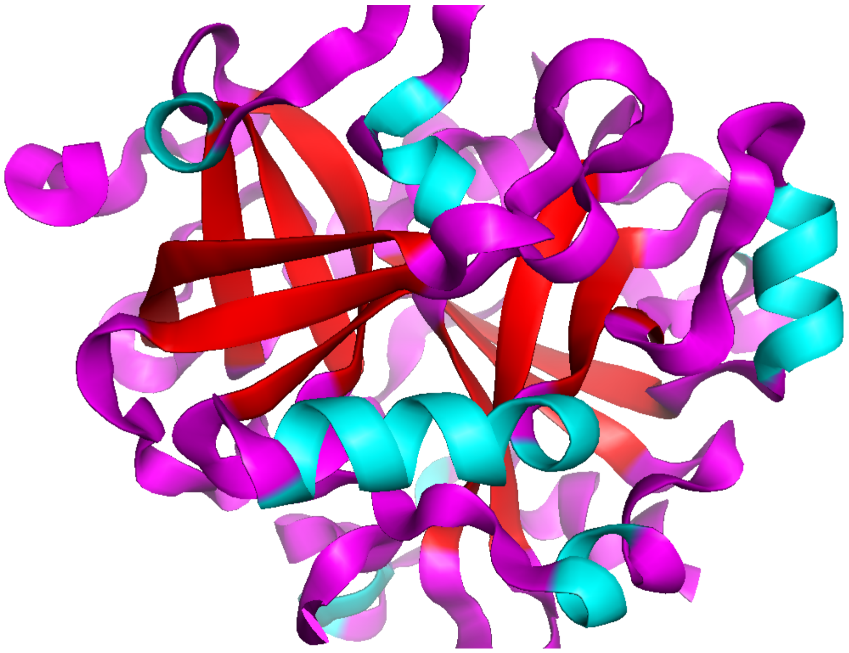


Figure 1.2. Secondary structure of protein 1AIX.

- *Tertiary Structure* - The molecules that comprise the protein in a three-dimensional space that correlates with a three-dimensional shape. The positions of molecules are said to be determined by the lowest energy state in such a way that molecules may achieve maximum stability. Figure 1.3.

- *Quaternary Structure* - The interactions and arrangement amongst separate protein subunits that create a larger protein complex.

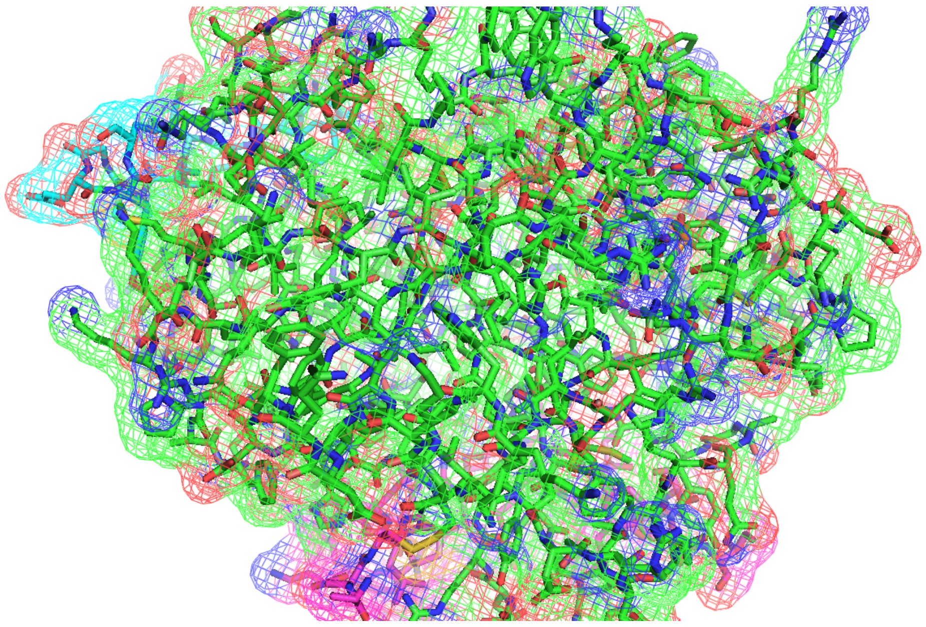


Figure 1.3. A tertiary mesh of the surface of protein 1AIX.

1.2 Genetic Algorithms

To be general, the study of proteins has presented a very robust problem set in computer science. The computational analysis of proteins has historically included:

- Comparison of protein sequences.
- The determination and prediction of functional groupings based on sequence and structure.
- The prediction of more complicated structures based on primary structure.
- The prediction (and analysis) of molecular interactions.

Examples of computational analysis of proteins will be discussed over the next few subsections.

1.2.1 Sequence Alignment

Sequence alignment is used to identify regions of similarity that may be a consequence of functional, structural, or evolutionary relationships between the sequences. The first dynamic programming algorithm to determine the two sequences' best global alignment came in 1970 [3], the Needleman–Wunsch Algorithm. This algorithm assigns a value for aligning the two sequences by using replacements, deletions, and insertions. In 1981 [4], the Smith-Waterman algorithm introduced a new scoring system that allowed for determining similarity between regions of protein sequence, otherwise called local sequence alignment. Although this happens at the global score's expense, global algorithms are generally not sensitive for highly diverged sequences with some localized similarities within them. These and other advancements in sequence alignment have provided a lot of knowledge when determining secondary protein structures.

1.2.2 Protein Structure Prediction

Protein structure prediction is one of the most popular areas of study in protein analysis. It is defined by the understanding and prediction of the protein's tertiary structure from its primary structure. When referring to the protein structure, we are talking about the functional conformation of that protein; each atom that makes up the protein has its charge; the composition of these atoms inside the protein creates the protein's electrical potential overall. This electrical potential and charge information allows for what we consider the function of the protein — i.e., the way other molecules interact with the protein as a whole.

The application for the determination of protein structure has produced many advances in the biotechnological field: the design of new proteins and folds [5]; disease study [6]; drug design projects [7]; refinement of theoretical models obtained by comparative modeling [8]; and obtaining experimental structures from incomplete nuclear magnetic resonance data [9]. To be able to predict the native structures of proteins would help the scientific community take advantage of the large amount of biological information that is being generated by genome sequencing projects; It would also allow for labs and researchers to determine when/if human error took place in the sampling or collection of protein data.

Determining the protein structure is one said by many to be viewed as the most exciting challenges of modern computational biology [10]. Not only has determining protein

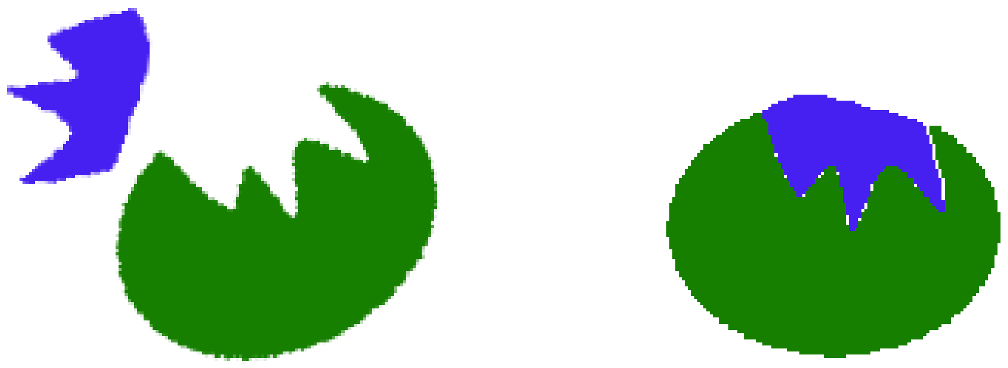


Figure 1.4. Lock and key representation of molecular docking. Left: blue represents a small molecule or substrate; green represents a protein or enzyme. Right: small molecule bound in a protein.

structures from their respective amino acid sequences remained a central problem in computational biology, but this determination has also crept its way into computer science and machine learning because of the scope and complexity of the problem. Even some of the most simplistic reductions of the structure prediction problem have been deemed an NP-Hard problem [11][12].

1.2.3 Molecular Interactions of Proteins

The study of proteins' molecular interactions is dedicated to determining how proteins interact with other macromolecules, ligands (a molecule that binds to another molecule), and proteins, represented in Figure 1.4. This study area is of the practical implication of protein function and has a great deal of impact on drug and toxin design because of it. Studies in the area have often been done under binding affinity, protein-ligand interaction, and protein-protein interaction. Proteins have such a wide array of interactions with many other molecules (some that extend further than just direct physical interactions), and tracking these interactions can form a functional web of protein-protein links, much like social media networks and the interactions between users on a platform. Proteins do not necessarily need to undergo a stable physical interaction to have a specific functional effect: proteins can catalyze subsequent reactions in a metabolic pathway; proteins can regulate each other transcriptionally or post-transcriptionally; or proteins can jointly contribute to larger, structural assemblies without ever making direct contact. Empirical scoring functions have been developed to estimate the binding affinity of a given protein-ligand or protein-protein complex with known three-dimensional structures. These scoring functions include terms accounting for Van der Waals interactions, hydrogen bonding, deformation penalty, and hydrophobic effect. Three different algorithms have been implemented to calculate the hydrophobic effect term, which can generally be run in parallel and used as scoring functions [13].

The timely determination of molecular interactions at the protein level is a significant focus of the pharmaceutical industry. The ability to have methods with fast producing and accurate results are essential to most techniques for structure-based drug design; these techniques also greatly aid in making medicines and analyzing toxins, so the speed of the results they generate has a direct impact on the lives that could be affected. This concept will be vital to one of the primary goals of this overall research later in this document and is a foundation for the research.

1.3 Tool Proposal

The overarching goal of this research is to provide a foundation for the development of a suite of machine learning applications and standalone tools that in combination allows for users to analyze proteins in multiple aspects from multiple starting points, as well as draw more profound conclusions from these protein conformations, and the information that generates them so that we can be advanced our overall understanding of how composition affects function. After development and integration, if managed properly, this software suite should be able to draw connections well past the currently available tools and apply a social media analysis tool to determine deeper meaning in protein interactions. This research would also lead to creating a database of reliable and verified information that could be used as training and testing sets for future research not tied to this project. The product of this research should be able to:

- determine the functional conformations of proteins from primary and secondary sequence

- determine likely mobility of atoms in the protein conformation
- determine likely folding patterns of the protein
- determine the potential interactions of protein-ligand and protein-protein complexes using binding affinity calculations and the shape information gathered from the topology of the tertiary structure of both molecules
- establish a database of proteins, protein shapes, and interactions that can be analyzed for deeper patterns of connection

That being said, the analysis of a protein's primary structure to determine its overall tertiary structure would be based on using a predetermined secondary structure in combination with empirical scoring functions to determine the all-atom low energy conformation of the protein. The methodology for predicting the tertiary structure uses a GA that judges the likelihood of a structure based on the conformation energies in the system and the secondary structures present in the tertiary conformation versus the structures that were predicted from its primary structure. These conformations are giving us likely positions of atoms in the protein and, in combination with a fuzzy logic, a potential range of movement that can exist for individual atoms or amino acids in the protein itself. After the possible conformations have been determined, we can also resolve the shape of the protein, particularly the individual pockets around the surface that can be used to reduce the search space in the determination of potential binding with other molecules.

1.3.1 Binding Site Prediction

This step of the tool, which is used to determine the binding affinity of regions of the protein, is done by determining the shape topology of the ab initio (all-atom) confirmation of the tertiary structure that is used as input into this stage of the software suite. This stage designs a surface around the protein structure based on the Van Der Waals forces of each atom, and then the exterior or reachable area of the protein is also mapped. In this shell mapping, small and large molecules that have been previously measured by this tool are compared to the inverse of the mapping to see if the shapes can fit given the valleys and peaks that may naturally occur in the mapping; these areas have been named functional footprints. Each functional footprint is measured for electrostatic potential, Leonard–Jones potential, and Coulomb's potential to judge its merits for docking against other functional footprints that are of similar shape as well as footprints that can fit within them. This information will be used to determine similar functional groups and potential binding locations for other proteins and molecular compounds.

Adjustments can be made to the initial algorithm developed to allow mobility amongst the atoms in the tertiary structures sent in from the recently mentioned step. This step is meant to allow for a level of flexibility amongst the protein shapes, dependent upon the electrical conformation of the space it currently occupies. The functional footprints can also be allowed to adjust for this range to allow for greater flexibility in functional footprints to be matched.

As the information that has been and will be stored in a database grows with the use of the tool, it is believed that this will hopefully lead to functional group determination based on shape and charge information; as well as become a source of information for potential connections between proteins and. Other molecules big and small alike. Amino acid grouping of these areas can be studied to determine the pattern that leads to the individual functional groupings. This tool can also lead to larger data sets to allow for deep learning applications between the protein's different structural classes.

Preliminary Results The current version of the algorithm uses a single probe per shape, comparison measures for shape similarity to consider the width, height, and depth of a potential docking location, and produces good results in terms of speed of return. Fuzzy logic is applied to both the shape and the electrical potential separately; then, the membership values are applied to a logical AND operation by multiplying the two values together using standard fuzzy logic. When we ran the Darunavir molecule on the shapes which correlated to each of the seven proteins in the database, it searched through all feasible docking locations on each protein very quickly.

With already favorable results, it is believed that advances can be made with concepts introduced in this dissertation. Those advances will allow for even greater success in predicting known locations. Moreover, the natural usage and growth of the database shall result in this system becoming a repository of similar data; in the future, it would also be desirable to use machine learning techniques, such as ANNs and potentially RBMs, to find functional groups of proteins using shared shape information, as geometry plays a large role in protein function. The determination and classification of functional groups of proteins is a large open area in the bioinformatics community.

1.3.2 Tertiary Structure Prediction

To generate highly favorable tertiary structures as a possible input to the shape determination, a hybridized genetic algorithm will be introduced for the tool's tertiary prediction portion. This algorithm attempts to integrate the accuracy of secondary structure predictors, like Support Vector Machines and Artificial Neural Networks, with the search optimization of the Genetic Algorithm to find protein conformations that are energetically favorable as well as attempting to maintain the predicted secondary structure. This algorithm requires not only the initial primary protein sequence as well as the secondary structure information that is attributed to each amino acid (which can be derived from the next section), but this algorithm will also need a fast-acting determination of the secondary structures present in the three-dimensional conformations of the predicted proteins. This portion of the algorithm must happen fast enough to allow for the program to generate a wide variety of conformations and run a shape comparison without requiring a vast amount of time between each generation.

Genetic Algorithms that find the most favorable conformations of protein have been done. The only methodologies that integrate the secondary structures of proteins are models that involve threading or adjusting the phi and psi torsion angles in between those structures. The algorithm we propose is wildly different than these models because it operates closer to an all-atom form where the torsion angle between the alpha carbons of each amino acid will the elements be adjusted. The knowledge of secondary structure is only applied to the fitness function; this will allow the individual molecules to loosen and find more positions of energetic preference. This will also give us information on the mobility and folding patterns of certain regions on the protein, allowing for the binding step of the tool to have information on how stationary the protein is at certain parts. The desire is this improves the overall accuracy of the all-atom tertiary structure predictors in comparison to what has been visualized by real-life experimentation; historically these two have not lined up very accurately.

1.3.3 Secondary Structure Prediction and Determination

This portion of the tool is dedicated to supplying secondary structure information as input to the hybridized genetic algorithms fitness function to check against generated conformations; this with the goal to help increase the accuracy of the ab initio tertiary structure prediction. Secondary structure predictors have already been well established to determine highly accurate predictions of secondary structures from primary structures; and though this area has already been researched, we attempt a novel application of a neural network that creates a quick prediction that can be attached to an in house tertiary structure predictor. This being said, the use of another secondary structure tool that can easily be integrated into our tool, as long as it is fast and can easily express its final result.

In this research, the novel application of the Restricted Boltzmann Machine (RBM), with different encoding methodologies will be applied to protein data with the goal of predicting secondary structures within the protein. These encodings range from just including the amino acid code to using the codons used to create the individual amino acids, including extra information regarding the amino acid being hydrophobic versus hydrophilic, polar vs. charged, etc. Comparisons with other popular and highly accurate techniques will also occur for large and small protein sizes.

The primary goal of this is to prove that one of the RBM encoding types will be able to produce results very rapidly compared to the more popular methods and remain as accurate as of the more popular methods. The more popular methods could also be used, but it would severely reduce the tertiary prediction model's speed. The secondary goal of this research is to leverage the RBM model to use and create deep learning techniques to find any underlying patterns in the construction of the secondary structures; this concept alone, if valuable pattern information can be pulled from the model, would be a significant advancement in the community understanding of how and why secondary structures for in proteins.

Even though the suite tool's beauty is that any step can be interchanged with either the finished product or another methodology to produce the same result, RBMs speed and size would be the preferred method to allow for machine limitations as the speed of runtime.

Preliminary Results Our lab has developed multiple methodologies RBMs to test the feasibility of the concept that the RBM would be an excellent machine-learning algorithm to analyze and predict the secondary structure; the primary being: a standard RBM (contrastive divergence and back-propagation learning strategy), a full-Ising RBM (minimizes the free energy of the system to approximate the minimum gradient), and a clique RBM. In the full Ising model, an application of Ising's Exact Solution is applied to allow for a slight speedup in selecting what is interpreted to be node values and weight adjustments.

Using a standard and full Ising RBM, we were able to train models to recognize an α-helix with a fair degree of accuracy. A sample of 160 member proteins was taken (from a collection of over 2000 proteins with α-helicies) and used as a training set. From this training set, we recognized the α-helix structures that we were training the model for and some α-helix that were not. Its worth mentioning that sequences that were significantly different from the relatively small training set used in creating the model were not well detected (even though some things are still recognized). This difference indicates that there may need to be different strategies to be applied in our training steps that may require a more extensive training set or multiple RBMs to detect a diverse group of secondary structure conformations. These results strongly suggest that RBMs are a useful way to encode higher-order descriptions of protein structure.

1.3.4 Protein-Protein Interaction Analysis

In the final significant section of this research, we will describe the software suite are used to analyze the interactions between proteins and other molecules. In the system we have described in the other sections, potential interactions of various types are collected, and no more profound analysis plan is described. In my proposed system, we would store the information about these relationships in the form graph and keep these connections using a graph database platform like neo4j. At this point, we would be able to analyze the relationships of the proteins themselves in a very similar manner to how they study the interconnected relationship between users in a social network like Facebook, Instagram, or Twitter. When applied to the information stored in our system, we would determine the similarity between proteins because of the information they hold and the topology of their graph kernels and relationship sub-graphs. We may even be able to conclude what molecules may be missing from individual interaction networks applied in that position.

This research proposes an adjustment upon the Weisfeiler-Lehman Graph Kernel algorithm that studies not only the topology of an interconnected element explicitly based upon the classification of edges that link nodes together but allows for patterns of specific classifications to be used in this analysis. This form of analysis will provide more accurate information about the type of edges that may exist between two nodes in a graph kernel. Still, it will allow those using the tool to determine if there are deeper connections in the two graph kernels they are analyzing.

Due to sample size limitations and data reliability, a proof of concept of this theory is applied to a social media data set instead of one of a biological nature. The application of the described graph labeling technique is performed on a social media site that provides information about user feelings upon connections: upvote/like vs. downvote/dislike; sentiment analysis of comments left under the original post; emoticons posted on messages and pictures; etc. This analysis was done with the standard Weisfeiler-Lehman Graph Kernel algorithm on the graph kernels of users in the system, as well as using the relationship-based Weisfeiler-Lehman Graph Kernel algorithm. This comparison is made to determine if using the relationship-based Weisfeiler-Lehman Graph Kernel, as many conclusions can be drawn from the same data if not more robust findings. This would be a controlled test where both methods are used to analyze the system, and then edges are removed to determine if both systems would be able to predict what type of edge would be missing.

1.4 Research Outline

During this document, you as the reader will be presented with a chronology of tools and information that details this research's ever-growing nature. The research will start as the project began, providing an accounting of the tool that prompted this research, ShapeDoc: a tool created to significantly reduce the search space to determine binding locations for small molecules on proteins. Next, the reader will be provided a history of the machine learning applications used to determine the tertiary structures of proteins and the initial drawbacks and results of the goals attempted by the Computer Science researchers that studied them. After this, a hybrid evolutionary algorithm for determining likely protein conformation will be discussed; as well as the limitation posed to it for the level of the desired accuracy. This portion of the research has to be noted as a future goal after other pieces are in place to make the desired algorithm perform as quickly as possible and produce multiple favorable results that would hopefully provide insight into the potential folding patterns and atom mobility within the macromolecules themselves. The penultimate area of the research will discuss the use of the RBM in the fast and accurate prediction of secondary structures from sequences of amino acid strings and the inspiration it provided for the full-Ising RBM and the creation of the clique RBM. In the final area of research, we will define the Weisfeiler-Lehman Relationship Graph Kernel and detail its potential use in the analysis of protein-protein and protein-ligand interaction networks; as well as apply this strategy, as a proof of concept, to the data pulled from the Stack Exchange data dump of the cstheory section of the site. This research will study and compare the graph kernels' isomorphic nature queried and determine specific elements of graphs that may be missing information. This document will then conclude with future desires for the overarching project and the next steps in creating a full tool to achieve the desired level of protein analysis.

PART 2

SHAPE DETERMINATION BASED OFF OF PREDICTED STRUCTURE

In this chapter, a method of searching for possible docking locations between a molecule and a protein is proposed. This novel method is done by determining and storing the unique shape conformations of a molecule and determining each shape's charge information. This area of the research portion was the inspiration for the entire software suite. Additional tools were made to improve the results this method provides and analyze, track, and verify potential connections determined by this algorithm. This research area also has the most practical implication as to its impacts on the field of drug and toxin design and analysis so much.

When it comes to analyzing drugs/toxins and their effect on organisms, the computational prediction of the binding of small molecules to proteins is a critical first step in developing novel drugs, inhibitors, and toxins. While many approaches for predicting binding have been developed and widely used in computational chemistry, there is room for significant improvement in computational performance and prediction accuracy. This work aimed to develop a very efficient algorithm for the rapid screening of many molecules (small and large) that is still reasonably accurate and suitable for data mining of chemical databases. The method described in this chapter runs a comparison of the shape and charge information, which is used to create a fuzzy measure that ranks the likelihood of docking.

2.1 Introduction

Molecular docking is a computational procedure that attempts to predict the non-covalent binding of macromolecules with other ligands; where non-covalent interactions are molecular interactions that do not involve the sharing of electrons, non-covalent interactions instead involve dispersed variations of electromagnetic interactions between molecules or within a molecule. This non-covalent binding can affect the behavior of the molecules around the protein [14] and lead to various effects in organisms, whether those effects are the production of inhibitors or the reproduction of viruses. For this reason, the determination of the binding of small molecules to proteins has a convenient application in the field of drug and toxin design [15].

There are many molecular docking methods, and the algorithms to predict binding that is being developed are steadily increasing [16]. Many of these algorithms share common methods with some of the most popular being:

- Molecular Dynamics Methods [17] [18]
- Genetic Algorithms and evolutionary programming [19]
- Fragment-based methods [20] [21]
- Point Complementary Methods [22]
- Quantitative Structure-Activity Relationship (QSAR) Models [23]

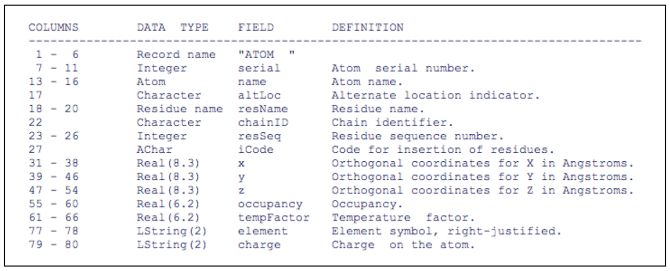
Where the terms above are defined, molecular dynamics methods are those in which simulations of protein solutions are performed with constant temperature or pressure, where what is analyzed is the conformational energies of the overall solution and the protein as well as the fluctuations of large molecules in solution or crystal environments; genetic algorithm methods are those involving general optimization algorithms that mimic the process of evolution through Darwinian natural selection, gene mutation, and reproduction; where fragment-based methods are those that attempt to use the divide and conquer strategy, that molecules are broken into fragments that are docked independently, and the molecule is rebuilt from adjacent fragment orientations [20]. The point-complementary methods use geometric recognition algorithms that use pattern recognition from a geometric approach to identify if the molecular surface is complementary. And where quantitative structure-activity relationship models are those that aim to find reliable measures of predicting the interaction of large numbers of molecules with each other [23].

Our novel method performs the classification of shapes along each protein and molecule to significantly reduce the search space of likely docking locations by eliminating pairings of unlikely or seemingly impossible shapes. Our algorithm's cornerstone was the integration of a relational database to hold the atom and protein data. In small-molecule files, using the information, it contains to calculate the electromagnetic potential or charge information. We also calculate the gradient of potential for each shape to measure binding affinity. With the shape information and the gradient of the potential, a fuzzy determination is created to determine the possible docking locations' ranking system. Using a set of 13000 atoms, we proved that our method succeeds in considerably speeding up the process of finding potential docking locations.

2.2 Methodology

The algorithm for the tool this research has named ShapeDoc is relatively simple, but the results are quite effective. In our methodology, we leverage our shape determination and our use of relational databases to reduce our problem's search space and increase the overall speed at which we can find solutions. To do this, we follow the following algorithm:

1. Import the protein into the database as individual atom types, their respective x,y,z coordinates, and the amino acid in which they belong
2. Determine the surface of the protein-based on Van-Der Waals radius
3. Classify Shapes around the protein
4. Determine the electric potential of areas around shapes
5. Add the shape and charge definitions to the database.
6. Determine the shape and charge definitions for the small molecule.
7. Cross-reference the two different lists of shapes to generate possible interactions and rank these interactions according to electrostatic favorability.



COLUMNS	DATA TYPE	FIELD	DEFINITION
1 - 6	Record name	"ATOM "	
7 - 11	Integer	serial	Atom serial number.
13 - 16	Atom	name	Atom name.
17	Character	altLoc	Alternate location indicator.
18 - 20	Residue name	resName	Residue name.
22	Character	chainID	Chain identifier.
23 - 26	Integer	resSeq	Residue sequence number.
27	AChar	iCode	Code for insertion of residues.
31 - 38	Real(8.3)	x	Orthogonal coordinates for X in Angstroms.
39 - 46	Real(8.3)	y	Orthogonal coordinates for Y in Angstroms.
47 - 54	Real(8.3)	z	Orthogonal coordinates for Z in Angstroms.
55 - 60	Real(6.2)	occupancy	Occupancy.
61 - 66	Real(6.2)	tempFactor	Temperature factor.
77 - 78	LString(2)	element	Element symbol, right-justified.
79 - 80	LString(2)	charge	Charge on the atom.

Figure 2.1. PDB source file format

2.2.1 Data Import

The initial starting point for the proof of concept requires extracting or parsing amino acid information and storing that information inside a relational database of atom and protein data. Initially, this was done by extracting the needed information from files from the protein database (PDB [24]); in the case of small-molecule files, using the source file information provided in the PDB files to precalculate and storing charge information allowed us to build a reusable data store of atomic-level information about the proteins, shape information about the proteins (including physical and chemical information), and small molecule shape information like in figure 2.1. Importing this positional information allows for us to query the database to make sure there does not already exist a copy of this file; and in case there is a comparison of the molecular position performed to determine if this version of this molecule is already stored. If the protein has a different configuration, then the shape of the protein's electromagnetic surface would also be different. A representation of the relational database structure used for this research can be found in figure 2.2.

2.2.2 Molecule Surface Creation

After importing the necessary information is loaded into the database signifying a molecule that has been accounted for, the next step is to determine a surface for the molecule that we are studying; this step applies to all molecules analyzed, whether it is a protein or a ligand. The way the surface is determined is by defining a molecular border for each atom present in our molecule. For this to be done, each atom in our molecule is accounted

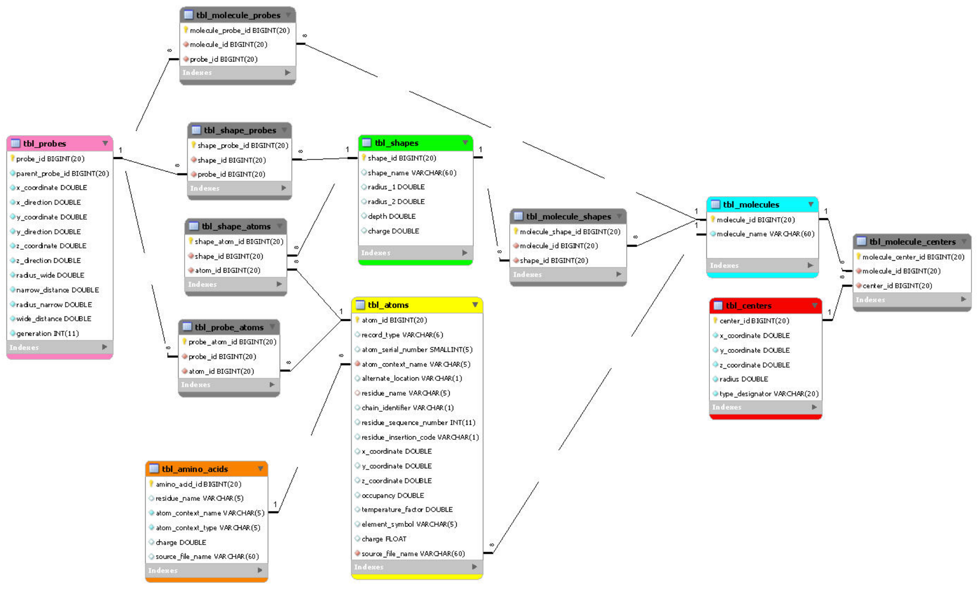


Figure 2.2. Relational database to hold the atom and protein data

for and given a surface using its Van der Waals radius. The Van der Waals force represents the distance-dependent interaction between particles, and the Van der Waals radius is the distance between two unbounded atoms when the electrostatic forces between them are balanced.

$$\theta = \frac{\arccos\left(-s^2 - 2r^2\right)}{2r^2} \tag{2.1}$$

For reference, the noteworthy atoms, seen in amino acids, and their respective Van der Waals radius have been included in Table 2.1. Using these radii of the respective atom and a shell spacing constant s, we represent each atom's surface by using a system of x, y, z coordinates, as referenced in Equation (2.1). When modeling the surface, a shell is built around each atom; the shell's radius is determined by the Van der Waals radius of the element.

Next, all points on this sphere that overlap a neighboring atom's sphere are absorbed and removed, combining (or taking the union of) the shells of the atoms together. The union of all atom surfaces represents the complete surface of the molecule, as referenced in

Table 2.1. Atoms and Van der Waals radius

Atomic Number	Element Symbol	Van Der Waals Radius
1	H	1.2
6	C	1.7
7	N	1.55
8	O	1.52
9	F	1.47
15	P	1.8
16	S	1.8
–	–	2

Equation (2.2) and figures 2.3 and 2.4. After the molecule surface has been fully defined, it will be stored inside of the database to save time for future comparisons.

$$protein_{surface} = \bigcup_{atom \in protein} atom_{surface} \tag{2.2}$$

2.2.3 Shape Determination

After surface determination, the topological shapes on the surface of the molecule can now be defined. In the proof-of-concept version of this research, the data for a single protein is retrieved from the database based on four-character PDB code and stored as an array of atoms; where each atom object contains the spatial coordinates, Van der Waals radius, charge (based on the standard charges of the atoms in amino acids), and information on the parent amino acid of the atom it represents.

At this point, a novel semi-common sense approach method is applied to the molecular surface determined in the previous steps. In this method, we map a series of points using the following system equation:

$$atom_{surface} = (x + r\cos(\theta * a)\sin(\theta * b), y + r\sin(\theta * a)\sin(\theta * b), z + r\cos(\theta * b)) \tag{2.3}$$

where in the equation r represents the radius from the center of the molecule, equation (2.5) and (x, y, z) shell surface point and C is a constant used for buffer space; a and b are

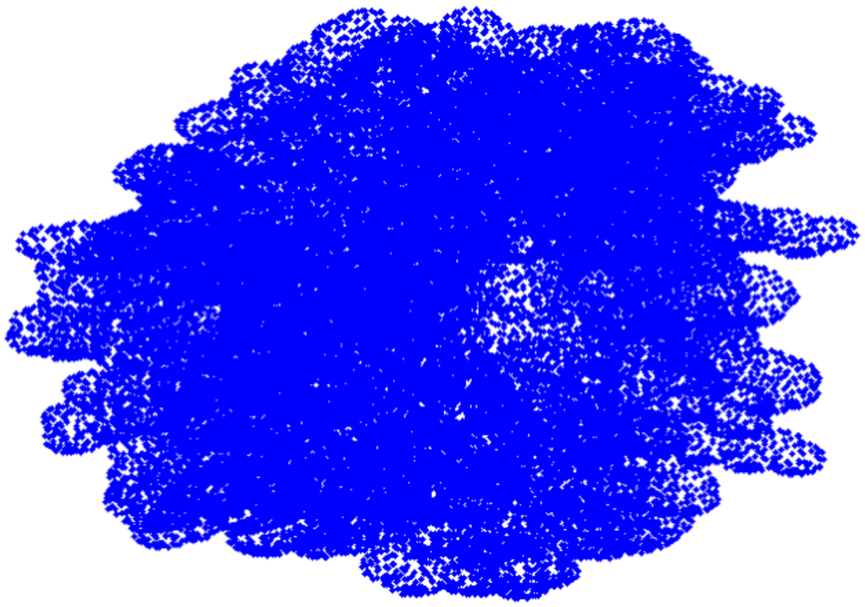


Figure 2.3. This is a representation of the dense model of the surface of the 3TTP protein (multi-resistant HIV-1 protease in complex with darunavir) with a distance of 0.01 of angstrom apart.

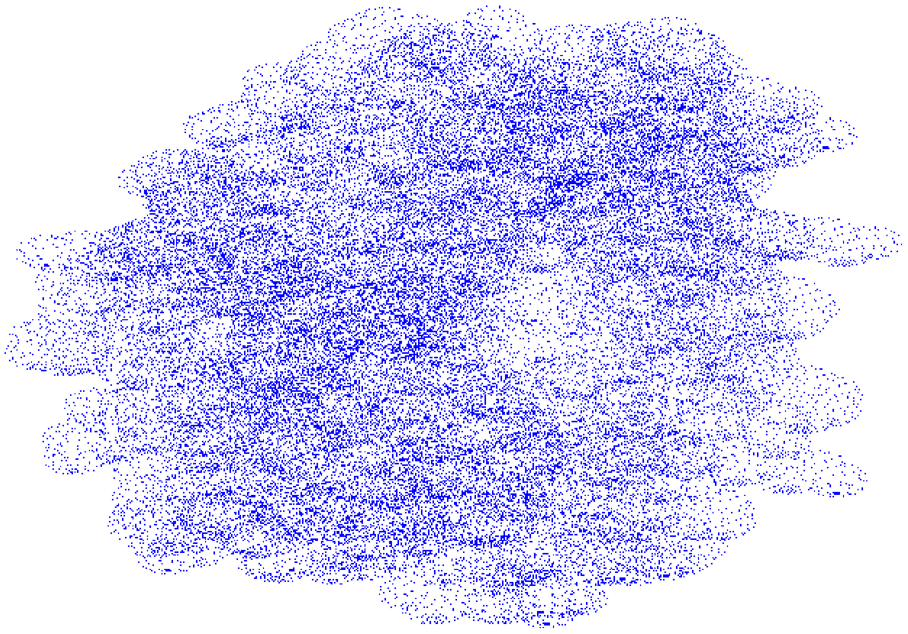


Figure 2.4. This is a representation of the sparse model of the surface of the 3TTP protein (multi-resistant HIV-1 protease in complex with darunavir) with a distance of 0.1 of angstrom apart. (PDB code 3TTP)

represented by (2.3) $\forall a \in [0, A]$, and $\forall b \in [0, B]$ where $A = \frac{2\pi}{\theta}$, and $B = \frac{\pi}{\theta} + 1$ respectively; and θ represents the angle of separation. This allow for use to map points spherically around the entire molecule based on the geometric center equation (2.4), of the molecule and the decided distance you would like the points to have from each other. The smaller the distance between the points, the more well defined the shape.

$$shell_{center} = \left(\frac{x_{min} + x_{max}}{2}, \frac{y_{min} + y_{max}}{2}, \frac{z_{min} + z_{max}}{2} \right) \tag{2.4}$$

$$shell_{radius} = max\sqrt{(x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2} + C \tag{2.5}$$

$$probe_{direction} = \left(\frac{x + x_c}{shell_{radius}}, \frac{y + y_c}{shell_{radius}}, \frac{z + z_c}{shell_{radius}} \right) \tag{2.6}$$

On the sphere, each point is considered as the starting point of a probe, and probes are extended inwards by a step size calculated by equation (2.6) where x_c, y_c, z_c are points taken from the geometric center of the molecule:

$$shell_{center} = (x_c, y_c, z_c)$$

This step is done until they touch they reach the surface of the molecule. A visualization of this is represented in 2.5

These probes are stored in the database as the definition of the shape of the molecule. When dividing the ligand surface into individual shapes suitable for matching as potential docking sites, each probe is examined and determined whether it is a local minimum or a maximum; where local minimums or probes with lengths longer than their neighbors, along the protein surface, are considered to be the low point in a valley; and local maximums or probes with lengths shorter than their neighbors along the protein surface are considered the high points in a peak. If a probe note considers a peak or a valley, it is added to a queue, and the next point down is examined in sequence until a known shape is encountered or a minimum or maximum is reached, defining a new shape. This process is repeated for all probes until every part of the surface has been explored and assigned to a shape.

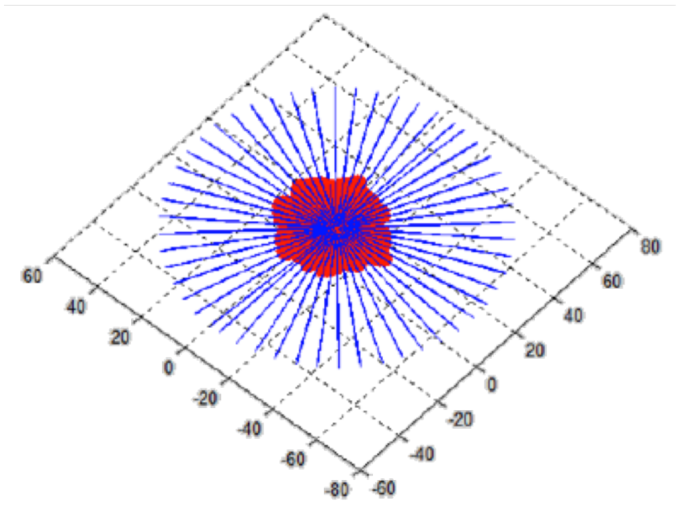


Figure 2.5. A visual representation of the probe step of ShapeDoc, around the 3TTP protein.

It should be noted that this is a one-to-one mapping, and in this iteration of the software, each probe can be owned by a single shape. The atoms that compose this shape are then determined by proximity to the probes that make the shape in question. Four physical parameters define the shape: height, number of member atoms, radius-1, and radius-2. The shape's height is determined by the difference between the height of the longest probe and that of the shortest probe in shape. When determining the two radii of the shape, two transformations are done so that the center point of the shape rests on the z-axis. The lengths of the shape in the x and y directions are then found, and the smaller size is assigned to radius-1 and the larger to radius-2. These parameters are what are used to compare shapes to fit physically.

2.2.4 Charge Definition

After the shapes have been determined and stored in the database, the next step is to define the charge information for the shape. This charge information needs to be defined to be able to determine likely molecular interactions. It is noteworthy to say that this charge definition is dependent on whether the molecule is a protein or a small molecule.

$$r = \sqrt{(x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2} + C \tag{2.7}$$

$$V_E(r) = \frac{1}{4\pi\epsilon_0} \frac{Q}{r} \tag{2.8}$$

$$E = \sum_{atoms} V_E \tag{2.9}$$

If the shape is on the surface of a protein, the charge is defined as the gradient of the electrostatic potential inside the shape. The electrostatic potential or Coulombs Potential and the derivation of its gradient are shown in the equation (2.8) and equation (2.9) respectively, where r is the positional charge of atoms given in equation (2.7); Q is the specific atom charge (sourced from a PDB file) and is equal to the electrical constant, $322.17752 Kcal/mol$ per angstrom. The gradient of potential was calculated for each shape based on the atoms that comprised it. This equation results in the potential being in the form of kcal/mol (kilocalorie per mol).

$$D_E = Qx\hat{i} + Qy\hat{j} + Qz\hat{k} \tag{2.10}$$

For the charge definition of a shape on a smaller molecule's surface, the dipole moment was used. The calculation of the dipole moment equation, (2.10), uses the small molecule's entire molecular structure and is based on the assumption of an overall neutral charge. Because there is no standard definition for an atom's charge, it had to be computed based on the known structure; Spatial Data Files (SDF) were used for this computation. SDFs are used by chemists and define what atoms are in a molecule and what bonds exist between them. This information is fed into an existing program, Another Molecular Mechanics Program (AMMP), to determine the special position of the atoms, their bonds, and the charge on each atom due to positioning. This electric potential, calculated by the equation (2.12) based on the aggregation of potentials of the atoms in the environment from the equation (2.12), is stored in the database along with the corresponding shape to be called when the fuzzy logic and comparison rankings are applied.

$$\nabla V_E = -\frac{Q}{4\pi\epsilon_0 r^3}\left(x\hat{i} + y\hat{j} + z\hat{k}\right) \tag{2.11}$$

$$\nabla E = \sum_{atoms} \nabla V_E \tag{2.12}$$

2.2.5 Fuzzy Ranking and Comparison

The first stage of the ranking system contains a pruning step that eliminates all shapes that correlate to peaks from small molecules that are too large for the protein's valleys. This pruning is done by comparing depth, width, and height to the two respective shapes. From this, we calculate a shape similarity by equation (2.13).

$$SS\left(p, m\right) = \left(1 - \frac{w_p - w_m}{w_p}\right) * \left(1 - \frac{d_p - d_m}{d_p}\right) \tag{2.13}$$

where w_p and w_m represents the width of the protein or valley shape and the width of the molecule or peak shape, respectively; and d_p and d_m represents the depth of the protein or valley shape and the depth of the molecule or peak shape respectively. In the case the size of the shape of the peak of the "docking molecule" is too large for a prospective valley shape, it is automatically given a membership value of zero, which effectively cancels the likelihood of membership. This step is a major speedup in search space when combined with our relational database because our initial query for shapes can filter out the infeasible shapes.

The second stage of the ranking system is used to determine the binding affinity of the two shapes. This step is not the end all be all from a biological standpoint because of the many different molecular interaction types. Still, as our database grows and our understanding of the information we get back from the interactions grows, our results from this binding affinity measure can be better interpreted for those separate interactions. When measuring the binding affinity of the two shapes, the dot product of the vector of the charges of the peak and the valley is calculated to determine the fuzzy membership value for binding affinity: represented by equation

(2.14).

$$BA\left(p,m\right) = \frac{C_p \cdot C_m}{max\left(\|C_m, C_p\|\right)^2} \tag{2.14}$$

where C_p and C_m represents the charge of the protein or valley shape and the charge of the molecule or peak shape, respectively. Ideally, the two charges will be equal and opposite, meaning that perpendicular charges would be unfavorable.

From the previous two steps in the ranking process, a general affinity membership may be calculated to factor in the charge information and the shapes that are in question. That being said, when combining shape similarity and binding affinity calculations, and overall membership value for binding can be calculated. This is done using the simple fuzzy equation represented in equation (2.15). The proof-of-concept version of the Fuzzy Determination calculation weights the shape more heavily in the overall membership function.

$$f_d\left(p,m\right) = \frac{SS_{p,m}\left(BA\left(p,m\right)+1\right)}{2} \tag{2.15}$$

2.3 Experiment

For this research, when analyzing this algorithm, their interest in the accuracy of the results and how swiftly those results could be attained. This research's goal was for the overall speed of the algorithm and the effectiveness of our shape similarity, binding affinity, and fuzzy measures to be determined as effective. For the ShapeDoc algorithm testing, a Darunavir molecule, a 75 atom compound, was tested again proteins stored in the MySQL database keeping a set of eight medium-sized proteins: 3TTP, 3OXC,5CHA, 3TKW, 4DQC, 1CHG, and 3UFN. Darunavir is a protein inhibitor known to bind with HIV-1 protease (PDB code 3TTP): the algorithm's flexibility and overall speed were demonstrated by running directly against Darunavir versus just an individual protein as most docking algorithms do.

The algorithm took 112.388 seconds to find the shapes correlated to Darunavir and store each shape, along with its charge information, into the database, with an average time of 4.5 seconds per shape. After all, shapes were found, the tool took 27.644 seconds to compare the shape of the Darunavir against the shapes correlated to each of the eight proteins in the database, from which 553 potential binding locations were returned.

Table 2.2. Top ten results based on fuzzy membership results, with combined shape and affinity determination

MDB ID	AA	Residue #	Ss(p,m)	Ba(p,m)	f_d (p,m)
3OXC	ASP	160	0.872433214	0.222370164	0.533218166
3TTP	MET	46	0.959990745	0.102672443	0.52927767
3TTP	GLN	92	0.864394472	0.216091439	0.525591359
3UFN	ARG	57	0.87642945	0.185890614	0.519674729
3OXC	ILE	166	0.863601159	0.202379828	0.519188307
3TKW	PRO	44	0.901837759	0.136108968	0.512292983
3TKW	ASP	60	0.793681923	0.285039546	0.509956329
2PKA	LEU	235	0.998250039	0	0.499125019
2PKA	LEU	108	0.994662513	0	0.497331256
3OXC	LEU	123	0.954290608	0.030470563	0.49168419

Table 2.2 records the top 10 results from the testing against the Darunavir molecule, a 75 atom compound. In Tables 2.3 and 2.4, the results are sorted and ranked by the top ten results for binding affinity and shape similarity (best fuzzy membership value) respectively. When ranked by best fuzzy membership value, of the top ten results, eight are proteins that Darunavir is known to interact with, while the other two results are the same protein 2PKA. Interestingly enough, the binding affinity for the 2PKA result had a membership value of zero - which implies that while the shape of the pocket is very favorable, the binding affinity would make it too challenging to dock at that location. Between the results sorted by binding affinity and shape similarity, the same proteins are highly ranked. However, because of the low shape similarity scores, the fuzzy membership values are still low.

2.4 Future Work

In future iterations, a measure of suitability could also be created by determining the probes' alignment to see if the differences in probe-length for a shape lines-up well with each

Table 2.3. Top ten results based on fuzzy shape comparison values

MDB ID	AA	Residue #	Ss(p,m)	Ba(p,m)	f_d (p,m)
3TKW	GLN	61	0.06214061	0.961263387	0.060937051
3TKW	LYS	45	0.073144085	0.922902992	0.07032449
3OXC	THR	196	0.107108736	0.808699771	0.096863773
4DQC	TRP	6	0.127841634	0.795284156	0.11475603
3OXC	ALA	128	0.04927735	0.769947463	0.04360916
3OXC	PRO	9	0.072587562	0.759986237	0.063876555
3TTP	ASN	98	0.218196564	0.759423495	0.191950081
4DQC	ARG	87	0.05298619	0.758817716	0.046596525
4DQC	ILE	62	0.19828912	0.75768409	0.174264816
3OXC	ASP	130	0.087277074	0.750739213	0.076399698

Table 2.4. Top ten results based on binding affinity values

Top 10 Shape Binding Affinities					
MDB ID	AA	Residue #	Ss(p,m)	Ba(p,m)	f_d (p,m)
2PKA	LEU	235	0.998250039	0	0.499125019
2PKA	LEU	108	0.994662513	0	0.497331256
2PKA	GLY	188	0.980740874	0	0.490370437
3TTP	MET	46	0.959990745	0.102672443	0.52927767
3OXC	LEU	123	0.954290608	0.030470563	0.49168419
2PKA	ILE	200	0.929318268	0	0.464659134
5CHA	GLY	187	0.926654171	0	0.463327086
3TKW	ARG	87	0.913662621	0.05274973	0.480929038
5CHA	PRO	4	0.906068794	0	0.453034397
3TKW	PRO	44	0.901837759	0.136108968	0.512292983

other. In terms of the binding affinity, future work will include factoring in hydrophobicity, which is said to be highly important in terms of docking [25]. Improvements in shape determination can also be made to refine more actual definitions inside the pocket (the potential docking location).

In the future, accounting for protein to protein interactions, which currently is not a more considerable step to take since the pockets of all proteins are already stored; only needing to store ”peak” information or possible binding regions that could bind to other proteins' pockets would be necessary. Natural usage and growth in our database will cause it to become a repository of similar data: in the future, it would also be desirable to use machine learning techniques to find functional groups of proteins using shared shape information, seeing that geometry plays a large role in protein function and that the determination and classification of functional groups of proteins is a large open area in the bioinformatics community.

2.5 Conclusion

The research of ShapeDoc brought forth as many questions as it provided answers. Initially, bringing in PDB files meant that the atom positions and charge were provided to us, but brings into question its reliability of this information. PDB files are of crystallized proteins meaning they are not in their natural state wherein most organisms are surrounds by hydrogen molecules; the absence of these molecules affects the electrostatic potential of the environment around the molecule. Also, it is known, the conformation of the molecule may change as the energies around the molecule changes as well as a protein may have mobility amongst the atoms that comprise it that still maintain the low energy conformation of its natural state or states. With the rigidity provided by the PDB file, accounting for this level of molecular flexibility becomes very difficult.

The statements bring into view the following questions: Can there be a level of flexibility in the proteins or the molecules that they comprise that allows flexibility in the shape we are looking to approach? Can this algorithm start from a secondary or primary structure and generate possible inputs for our database to grow information? How can existing observed knowledge about proteins be used to improve the data stored in our database? The questions require an analysis of protein structure prediction and the different techniques used to predict these other confirmations; this analysis is provided in the next section.

PART 3

HISTORY OF MACHINE LEARNING IN PROTEIN STRUCTURE PREDICTION

As previously stated, the determination of a protein's functional conformation is one said by many to be viewed as the most exciting challenges of modern computational biology. There are two classes of methodology that are used for this challenge: comparative modeling method; or ab initio methods; These methodologies present different complexity and, in many cases, solutions. In comparative modeling, databases of known structures are referred to for the development of the protein structure. Comparative modeling methods include homology modeling, threading, fold recognition. Ab initio methods use the amino acid sequence (the primary structure) and their interactions to determine the structure. In ab initio methods, the correct structure is generally calculated by determining the overall energy of the structure; the lower the energy, the more stable the structure, which increases its probability of correct conformation and structure.

When developing protein structure development strategies, the major goals are to develop more reliable ways to determine the correct and incorrect structure and create or improve methods that find functional conformations from those correct structures. To determine if a structure is correct or incorrect is the definition of energy conformations, one must account for an array of physical properties: Gibbs free energy, hydrophobicity, electrostatic potential, Van der Waal's force, etc. Physically it is understood that conformations with the lowest Gibb free energy are the most stable structures and are, by definition, the natural conformations. Protein folding may be altered by outside factors like chaperones and electrostatic conditions of the environment, and therefore isolated, and refolded proteins may be non-native. Different machine learning algorithms have been proposed for searching through possible conformations as well as efficiently evaluating the quality of pro-

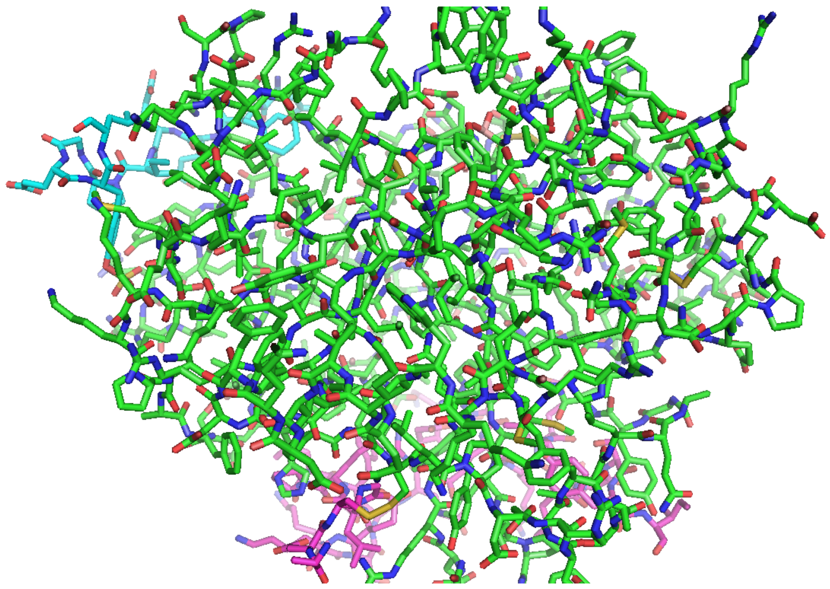


Figure 3.1. A figure of a protein's chemical structure, *1AIX bonds*.

tein models; these algorithms include but are not limited to evolutionary algorithms like *Genetic Algorithms* (GA), Monte Carlo Methods, and supervised learning methods like *Artificial Neural Networks* (ANNs) and *Support Vector Machines* (SVMs). This section will cover the machine learning methods currently being used to solve the challenge of protein structure prediction and the benefits and drawbacks of them.

The rest of this section is organized as follows: in part 1, GAs and the factors that should be taken into account when applying them are described. In part 2, the application of Supervised Machine Learning is described. In part 3, the use of Fuzzy Logic for protein structure prediction is described. In part 4, this section is summarized and concluded, giving rise to the research's next steps.

3.1 Genetic Algorithms

One of the most common machine learning applications for ab initio protein structure prediction is the evolutionary algorithms' use, specifically *Genetic Algorithms* (GAs). GAs are general optimization algorithms that mimic the process of evolution through Darwinian natural selection, gene mutation, and reproduction. This mimicry comes into play with the significant steps of this type of algorithm: fitness test, crossover and mutation. Recent studies from Custòdio *et al.* [2] show that genetic algorithms are said to be superior to the other evolutionary algorithms.

The benefits of GAs lie in the distinct characteristic of maintaining a large population of potential solutions characterized as chromosomes, genes, or individuals, depending on the author. These chromosomes are all in competition with each other in the form of a fitness function used to determine a possible solution's strength. Chromosomes are written as strings of information (in basic cases, binary string where each bit represents a specific value). The initial population is usually a random set of possible solutions. Each generation, including the initial population, every member is evaluated and ranked by their respective fitness. In most common GAs, the highest fitness individuals are very likely to be chosen to be members of the next generation. In contrast, individuals with minimal fitness values are improbable to be selected to go on to the next generation.

Mutation and crossover are additional GA operations applied to chromosomes' sets to generate new individuals from generation to generation. The mutation operation represents the random and slight genetic changes that happen in nature; while the crossover operations represent the transfer of information in reproduction. There are multiple types of each, and depending on your problem, some versions of mutation or crossover may work better than others.

The algorithm is given a percentage likelihood to select random bits of information to be changed in mutation. This step usually happens in one of the following three ways: a single point mutation (bit inversion), segment mutation, or order changing. In a single point mutation, only a random bit of information is changed to another possible value. In segment mutation, a block of data is all mutated; sometimes this presents itself as a single bit being mutated and the rest of the gene adjusting to make it an optimal solution. In the order changing method, two or more bits of information are exchanged.

In a crossover, the algorithm is given a percentage likelihood crossover will occur in the current generation of individuals; the fitness function's ranking usually influences how individuals will be paired to perform a crossover. Crossovers typically happen in one of the four following ways: single point crossover; multi-point crossover; uniform crossover; and arithmetic crossover. A single point cross over a point is selected in both individuals; this point is called *crossover point*. The new genes are created by separating individuals at the crossover point and combining those halved genes with the opposite half of the other individual. In a multi-point crossover, like a single-point crossover, you select multiple crossover points, which splits each individual into segments combined with another individual's respective segments. In a uniform crossover, the child gene is created by choosing random genes from both parents, and in arithmetic, the child gene is a product of some function of both of the parents (e.g., intersection, exclusive-or, etc.).

In each GA, the functions covered are applied in some way in hopes that after a certain number of generations and with a large enough initial population, the individuals will converge on an optimal solution or solution set. The effectiveness of a GA is all in determining how the fitness test, crossover, and mutation are applied.

3.1.1 GAs Model Selection in Protein Structure Prediction

When applying GAs to ab initio protein structure prediction, the primary goal is to find the lowest energy conformation's structural conformation. This process is not an easy task. One of the simplest and common applications used to find possible low energy conformations (the hydrophobic-hydrophilic model) was determined by Unger and Moult to be an NP-hard problem [11, ?]. But before looking at possible ways to determine energy conformation, when designing a GA for this problem, one must first decide what type of model to use to formulate the problem. Currently, the types of models one has to choose from are cubic lattice models and all-atom models. The cubic lattice model is a discretization of the protein's conformational space; predicting the exact protein structure is impossible [2]. However, this system is still very widely used because it is significantly less complex and still captures many of the aspects of protein structures [26]. Using the cubic lattice, you can predict folding patterns. It still presents the issues that arise from predicting the energy landscape

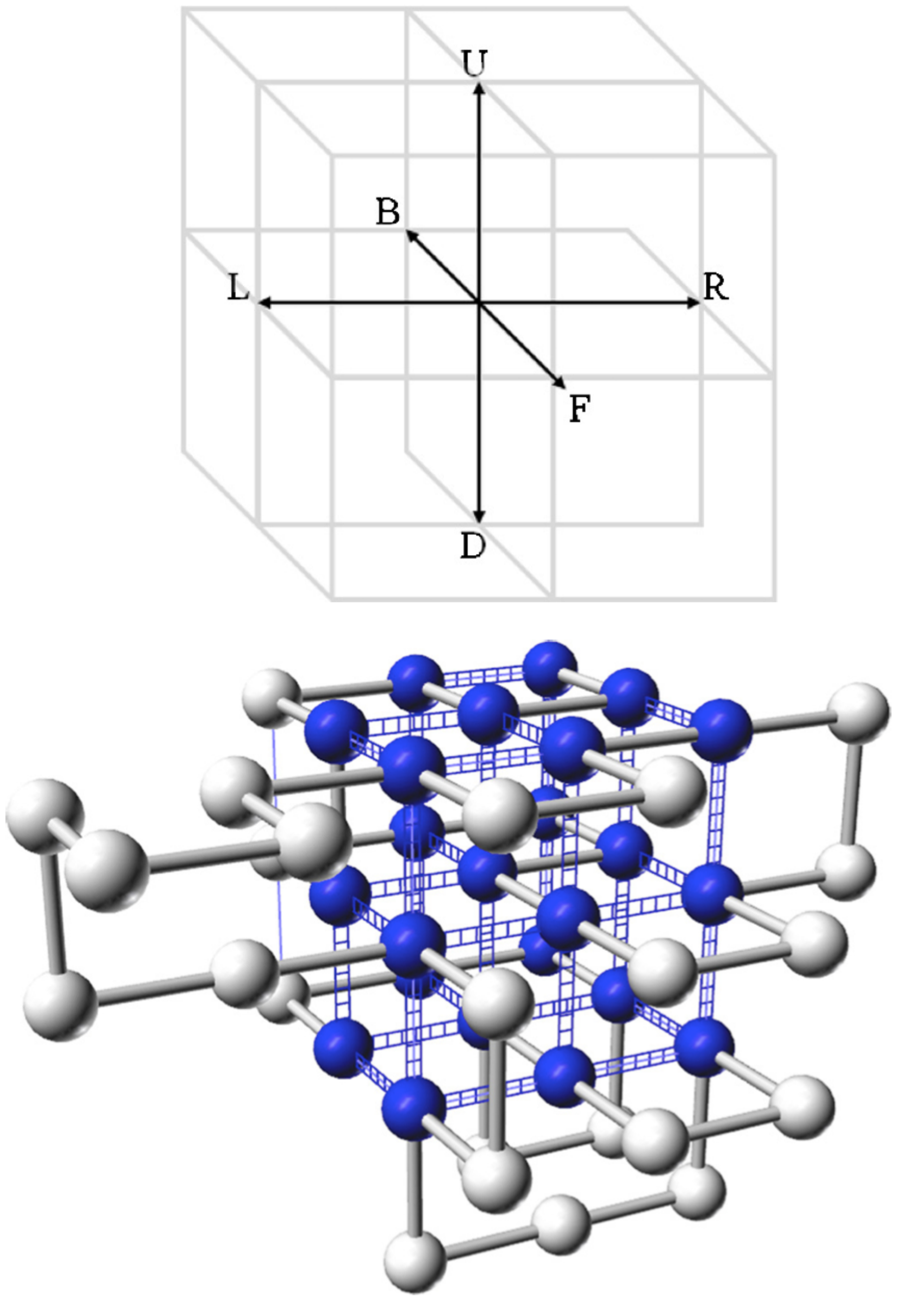


Figure 3.2. An example of the possible moves in the HH model (top), as well as a sequence with 48 monomers [HHHHPHHPHHHHHPPHPPHHPPHPPPPPPHPPHPPPHPPHHPPHHHPH] from [2].

of a protein. Some of these issues are: massive multimodality; multiple conformations with equal energy; and large regions of unfeasible conformations [2]. With the cubic lattice model, each amino acid of the protein is given a position on a three-dimensional grid and linked together in a chain with backbone dihedrals as depicted in figure 3.2.

Most authors that use the cubic lattice model [2][6][12][26] use a specific type of cubic lattice model called *Hydrophobic-Hydrophilic* or *Hydrophobic-Polar* (HP) model. This application is a simplified model in which each amino acid is recognized as hydrophobic or hydrophilic. This step is because hydrophobic amino acids tend to migrate toward each other to avoid polar solvents; this migration causes the formation of a hydrophobic core. The only requirement for this type of system is that there can be no collisions of amino acids in the conformation. In this system the energy conformation is calculated by counting the number of *Hydrophobic-Hydrophobic* (HH) contacts in adjacent non-bonded sites in the lattice, so the equation for a valid conformation would be the following:

$$E(c) = -n$$

In this function for energy conformation c represents the conformation and n represents the total number of HH contacts in the conformation. There are other equations that may calculate energy conformations for the cubic lattice model, some of which will be further discussed in the next section.

The all-atom model is designed to be much more involved in behavior than the cubic lattice model. In this model, elements are no longer the amino acids in the protein, but all the atoms comprise the protein. The chain is no longer bound by a lattice or backbone dihedrals in the all-atom model (figure 3.3 and figure 3.4). The all-atom model is considered more problematic because an experimentally observed native structure may have a higher potential energy conformation than an unnatural but optimized structure. This assertion is because, in all-atom models, the energy conformation is not the only thing that matters. It is also necessary to account for the effects on the system's entropy as a whole [2].

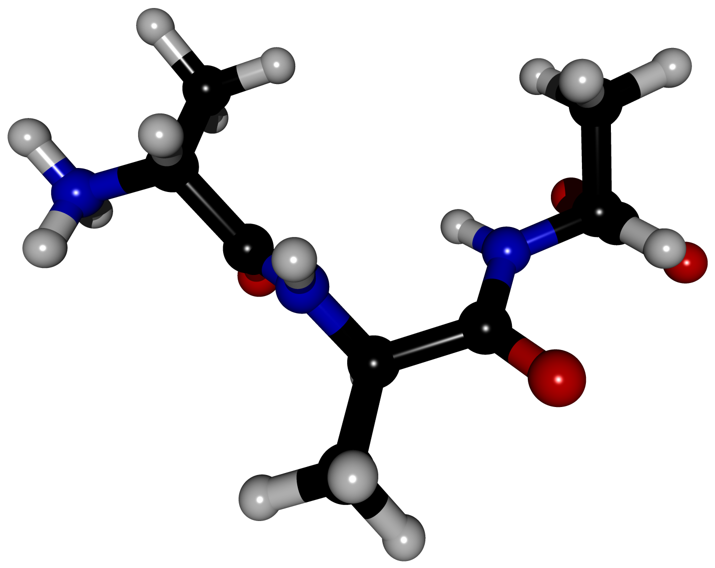


Figure 3.3. A molecule used to depict the ϕ label phi , ψ labeled psi, and ω dihedral angles.

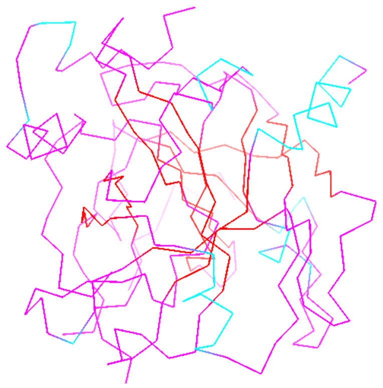


Figure 3.4. The final product of an all atom model would look similar to this. This is the actual backbone of protein 1AIX.

In all-atom models, energy conformations are calculated by using the physics of the atoms themselves. The potential energy is evaluated by calculating force fields, which are empirical and classical physics approximations for the properties of atoms and chemical bonds. The Leonard-Jones potential and Coulomb's Law are used to describe the electrostatic interaction between particles, which take into account the polarity of every atom in the system. The uses of both the Leonard-Jones potential and Coulomb's Law are discussed in more detail in the next section and the equations used to represent them. While the force field can be directly optimized, a better approach is to use molecular dynamics simulations to model the atoms' motions and search conformational space for minimums.

3.1.2 Common Methodology

In a GA, the methodology comes down to a few key points: individual representation, fitness test; mutation rates and attributes; and crossover rates and attributes. Though population size and the number of generations are critical for a GA's performance, there is little general guidance from theory. They are constants that are empirically adjusted to maximize the effectiveness of a GA on a specific problem. It is noteworthy to mention that even though each individual in the initial population is randomly generated, it is better to have a diverse set of individuals to cover a broader range of potential solutions.

Individual Representation In the models previously discussed, current work only has the representation of the individuals in a very particular way. For the cubic lattice model, whether it is an HP model or something slightly more complicated, an individual's representation is usually very similar. In [12] Unger and Moult stated the concept of giving amino acids in the individual an absolute direction on the square lattice, i.e., an individual is coded by a sequence $(U, D, L, R, F, B)^{n-1}$ where U, D, L, R, F and B code respectively up, down, left, right, forward and backward movement on the lattice, and n is the length of the protein. In [27] the backward move was removed, and the remaining moves where given in terms of relative movement from the current position; this step was done to avoid invalid conformations further. This representation is still in use for these types of models [2].

In the representation for all-atom models each atom is given three dihedral angles (figure 3.3) usually characterized by (ϕ, ψ, ω). In some cases, ω is excluded because even though the torsion angle ω are explicit, they do not change during execution and are generally kept in the $trans(180°)$ configuration models [2].

Fitness Test The fitness functions of most GAs for protein structure prediction are directly related to the energy functions or force fields of proteins. Molecular mechanics can measure energy functions and force field components. Some of the functions are bond length potential, bond angle potential, torsion angle potential, improper torsion angle potential, dihedral angles, van der Waals pair interactions, electrostatic potential, hydrogen bonds, etc. [628628]. As mentioned in the previous section, a simplistic method for the cubic lattice model consists of counting the number of HH contacts. A model with more contacts is in lower energy conformation than one with fewer contacts.

Because of the natural complexity of the system in an all-atom model, many more functions can be used but this does increase the overall complexity of the GA. Some fitness functions measure the summation of the different potentials [28] but newer ones use equations that account for instances. An example fitness function, based on [2], is:

$$E_{total}(r_i conformations) = E_{torc} + E_{LJ} + E_{coul} \tag{3.1}$$

Where E_{torc} is the torsion energy (equation (3.2)), E_{LJ} is the Lennard-Jones potential (equation (3.3)) and E_{coul} is the Coulomb or electrostatic potential (equation (3.4)).

$$E_{torc} = \sum_{n}^{(N_\phi)} K_{\phi_n}[1 + cos\,(n_n\phi_n - \delta_n)] \tag{3.2}$$

Where, in the above calculation of the torsion energy, n_n is the period, ϕ_n is the torsion angle, δ_n is the phase angle, and K_{ϕ_n} is an energy constant based on the torsion of a bond.

$$E_{LJ} = \sum_{i \leq j}^{N_{atoms}} - \left(\frac{A_{ij}}{r_{ij}}\right)^6 + \left(\frac{B_{ij}}{r_{ij}}\right)^{12} \tag{3.3}$$

Where, in the above calculation of the Leonard-Jones potential, values A_{ij} and B_{ij} are Lennard-Jones parameters that depend on the atomic type and the distance, r_{ij}, between atoms i and j at which the inter-particle potential is 0.

$$E_{coul} = \sum_{i<j}^{N_{atoms}} \frac{q_i q_j}{4\pi\varepsilon_0\varepsilon_r(r_{ij})r_{ij}} \tag{3.4}$$

Where, in the above calculation of the electrostatic or Coulomb's potential, q_i and q_j are the charges of atoms i and j, and $\varepsilon_r(x)$ is a dieletric function that depends on that distance represented by x [2].

Crossover and Mutation Applying the proper type and amount of crossover and mutation is very important to the optimal solutions' speed. Though fitness functions are also essential, mutation and crossover are how the individuals change, and new solutions arise. In an early paper of Krasnogor, they determined in the HP that single point crossover could not transfer blocks of information well. Still, segment mutations acted as a powerful local search, which leads them to stat the best combination of parameters had a small crossover probability and a high mutation and segment mutation probability [26].

In [2] Custòdio *et al.* used combinations of six different genetic operators to get solutions in both their HP model and their all-atom model. These six comprised a two-point crossover, a multi-point crossover, a segment mutation, an exhaustive search mutation, a local move, and a loop move. With exhaustive search mutation Custòdio *et al.* test all possible directions for a randomly selected base and keep only the best solution; This demand required four fitness functions in their paper and was developed to hybridize the GA and an exhaustive local search. Local moves and loop moves are ordered, changing mutations. Where local changes swap the movements to random locations, loop movements swap locations from two units that are five bases apart. Custòdio *et al.* also mention a repair function when collisions happen that fix invalid conformations when crossovers and mutations create them.

Tracking Results The quality of the model found with a GA depends on the speed of the calculations and the quality of the energy function that is being searched for an optimum. The calculation speed can be estimated from the number of energy evaluations the GA has to perform as the system evolves from generation to generation. Also, from [7] we know when testing against a known structure, a prediction is considered to be similar if (i) secondary structure elements agree with the known structure; (ii) the orientation of those secondary structures are within 20° of the ones in the known structure; and (iii) the length of the secondary structure elements different by no more than two residues of the known structure.

We know from foundational papers like Unger and Moult [11], [12] that applying GAs to the HP model outperforms the Monte Carlo method in terms of finding a global minimum of two-dimensional lattices [12]. This concept was improved upon later by [27] which achieves a high number of HH contacts with fewer energy evaluations. We must mention that many early prediction methods are said to show experimental bias [29]. Pederson and Moult described in their paper that potentials in [7] were parameterized to favor experimental structures; earlier papers including [30] relied on experimental secondary structures for success. They even mention their previous works were only effective on small peptides.

More recent papers have shown less bias and continued success in furthering the belief that GAs are a great way to tackle the protein structure prediction issue. In the most recent of the studies, Cusòdio *et al.* [2], prove their method is not only an efficient way to investigate the complex energy landscape of proteins but also show an increase in the probability in finding native structures. For the cubic lattice method, they developed tests that were made against three HP benchmark sequences. It showed comparable solutions to models specifically designed for the HP model and superior to other evolutionary algorithms like the Hydrophobic Zipper strategy [31], the Contact Interactions method [32], Constraint-

Based Hydrophobic Core Construction [33], and others. Even the all-atom implementation they developed found native structures with a success rate of 100% when they tested it against the poly-alanine peptides.

3.2 Supervised Machine Learning Algorithm

Referring back to the introduction, in terms of the protein structure prediction methods like homology modeling and threading, the most common machine learning algorithms applied are those under the umbrella of supervised learning. Supervised learning algorithms are those in which inferences are made from labeled training sets. By labeled training set, we mean a provided set of examples where we know what group or solution each sample is provided; the set itself in generally called a *training set* and the labels that correspond to the training set are called *training labels.* In supervised learning algorithms, examples are usually provided in the form of attribute vectors. After your algorithm is trained to predict a label hypothesis for each example provided, you run it on your sample data to generate your answer. Traditional statistics methods and methods created to mimic the way neurological networks work are generally the functions one uses to create these label hypothesis. This section will discuss the application of the two most popular applications of supervised learning algorithms to the protein structure prediction problem, specifically the homology modeling and threading techniques. These applications of supervised machine learning algorithms are including *Artificial Neural Networks* (ANN) and *Support Vector Machines* (SVM).

3.2.1 Artificial Neural Networks

Artificial Neural Networks (ANNs) are computational models that were first created by Rosenblatt in 1959 [34]; inspired by the networks of biological neurons that contain multiple layers of simple computing nodes. In an ANN, each node operates as nonlinear summing devices and is represented as the neuron in the brain, accepting input values from feeding information and sending the processed outputs through the whole network. ANNs are very robust and have been successfully applied to solving many complex problems, including

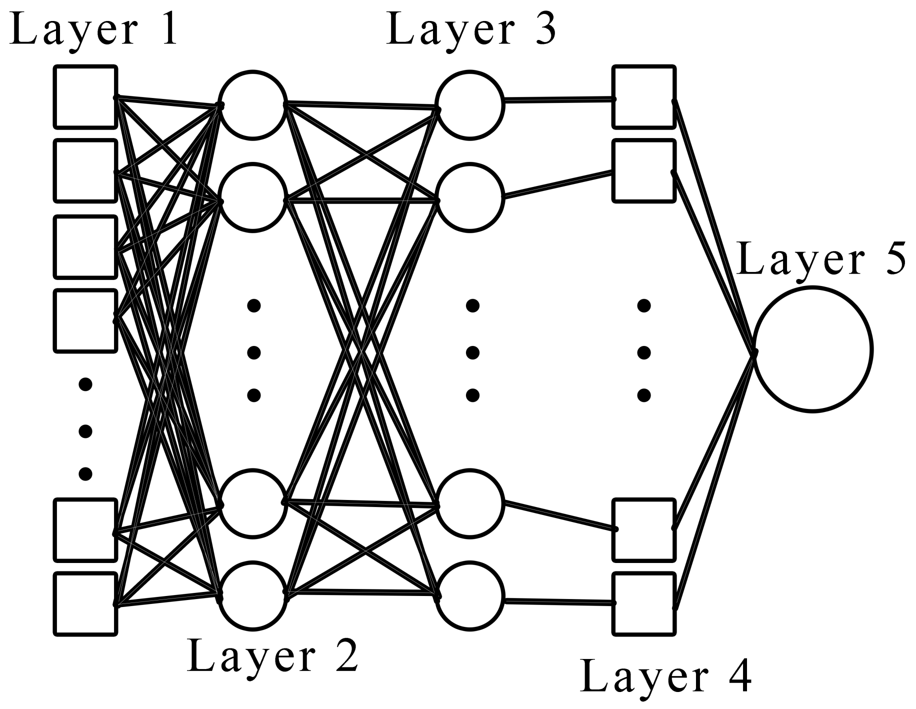


Figure 3.5. Basic structure of an ANN.

prediction, classification, and pattern recognition, and so on. Several layers give the structure of an ANN. These layers are divided into three groups: the input layer, the hidden layer, and the output layer. The input layer is the level at which the input data is fed into the ANN. The hidden layer is the level at which calculations and most of the processing is performed; the hidden layer is also the level at which the input values are subjected to the weighted values used to represent learned behavior. The output layer is the level at which results are provided. The most commonly used ANN in the protein structure prediction problem, and one of the most commonly used in general, is the feed-forward back-propagated ANN. Feed-forward ANNs are those where the nodes do not form a directed cycle; information moves only in one direction. Back-propagation ANNs adjust the weights in the network to minimize the error in the output. In a feed-forward back-propagated ANN, each node's activation state X_i is a real value between 0 and 1. The strength of the connection between a node i and another node j is represented by a real number W_{ij}. The activation of a given node in the ANN can be calculated by summing the products of every input node's output Y_j and weight W_{ij} and then adding a bias term, b_j

$$X_i = \sum_j W_{ij} Y_j + b_j$$

Having calculated the activation X_i for a node, the output of that node Y_i can be computed using the logistic sigmoid function and then propagated to the ANN's next layer.

$$Y_i = \frac{1}{1 + e^{-X_i}}$$

During each cycle, the input has been given to the ANN. The nodes' weight has been adjusted at the end of the cycle, and this procedure is repeated. This form of supervised training, in which the desired output is presented to the ANN and the inputs, has been used to train the ANN [35]. In protein structure prediction, ANNs are used to determine the protein backbone's conformation using ϕ and ψ dihedral angles.

3.2.2 ANNs in Protein Structure Prediction

In 1988, Qian and Sejnowski [36] were the first to use ANNs to predict protein secondary structure. The model they used is a fully connected *Multi-Layer Perceptron* (MLP) with one hidden layer [37] (Haykin). The input is a small window of 13 amino acids, and the three output nodes are the probabilities of an amino acid belongs to a helix, sheet, or loop region. They randomly weighted each input and used back-propagation [37] to train the system on a PDB file chosen from the PDB. The whole system achieved up to 64% accuracy, and the authors suggested that the significant improvement could not be made. However, despite this, a wave of papers using ANNs to predict protein secondary structure was published later using similar techniques [38][39][40]. Scientists and researchers are working hard to improve the performance of ANNs. One method is to include more information in a combination of the inputs, such as the evolutionary information and multiple sequence alignments [41][42]. Moreover, due to the PDB database's rapid growth, the training sets for ANNs also increased. This therefore enhanced ANNs' performances [43][44][45]. Furthermore, other algorithms were combined with ANNs in order to improve their performances [35][46].

Servers were built, based on these algorithms, to provide a service to the structural biology community. McGuffin built PSIPRED protein structure prediction server based on three methods, PSIPRED, GenTHREADER, and MEMSAT 2, and his colleagues in 2000 [47]. In 2008, Jpred 3 secondary structure prediction server [48] was settled based on the Jnet algorithm proposed by Cuff and Barton in 2000 [49]. Since these artificial intelligence methods are based on an empirical risk minimization principle, they have some disadvantages, for example: finding the local minimal instead of global minimal; has a low convergence rate; more likely to over-fitting; and when the size of fault samples is limited, it might cause poor generalization. Even though scientists are trying to overcome these disadvantages [50], most of them remain.

3.2.3 Support Vector Machines

The theory of *Support Vector Machines* (SVMs) was developed by Vapnik and Cortes in 1995 [51]. Though they can expound to multiple classes, at a very basic level, SVMs are designed for two class classification problems. What is provided to the machine is a set of $n - D$ dimensional vectors x_i and their associated classes y_i:

$$\{x_1, y_1\}, \ldots, \{x_n, y_n\} \in \mathrm{I\!R}^D \times \{\pm 1\}$$

The main idea is to construct a hyperplane to separate the two classes (labeled $y \in \{-1, 1\}$ so that margin (the distance between the hyperplane and the nearest point) is maximal. It is equivalent to solving the following optimization problem:

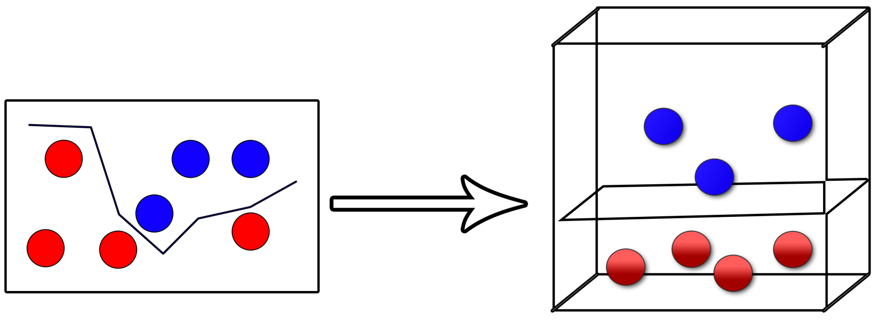


Figure 3.6. Basic structure of a SVM.

$$\min \frac{1}{2}(w \cdot w) + C\sum_{i=1}^{n} \xi_i$$

with constraints

$$y_i\left((w \cdot x_i) + b\right) \geq 1 - \xi_i]$$

$$\xi_i \geq i = 1, \ldots, n$$

where ξ_i is a slack variable giving a soft classification boundary and C is a constant corresponding to the value $\parallel w \parallel^2$ [52]. The boundary is also called the *optimal separating hyperplane*, which is chosen by maximizing the distance from the closest training data [53].

SVMs are considered amoung the most efficient supervised machine learning methods not only because of their well-developed learning theory [54][55], but also because of their superior performances in the practical applications in pattern recognition problems

[51][56][57][58][59]. SVMs have many useful features, including: The avoiding of the over-fitting with the use of structural risk minimization; the capability of handling large feature spaces; the insurance that the training converges to a globally optimal solution; the condensation of information in a given data set is made without a loss of useful information [60].

SVMs have been applied to solve the protein structure prediction problem since 2001 [60]. In Hua *et. al* [60], the authors used frequency profiles with evolutionary information as input. The goal was to improve the performances of the SVM. Their research issues were to define the type of kernels used and what data to represent, or encode, for the training.

One method is to study different SVM kernels, for example using a quadratic kernel function to increase the accuracy of helix prediction [61], dual layered SVMs with a profile [62], two layers of SVMs with weighted cost functions for balanced training [63] and mixed-modal SVMs [64].

Another way to increase the accuracy of SVMs is to include different encoding methods as input for SVMs: SVMpsi incorporated PSI-BLAST *Position Specific Scoring Matrix* (PSSM) profiles as an input vector [65], mixed encoding schemes of an orthogonal matrix, hydrophobicity matrix and BLOSUM62 substitution matrix together with SVMs [66]. Chen *et al.* [67] used PseAAC [68] as the protein sequence representation, the optimal length of local protein structure was used as input [69]. Furthermore, combinations of the above two methods have been tried to improve the performances of SVMs [70]. Normally, SVMs are binary classifier. In some studies, scientists also extend this classification method to predict multi-class structures [71][61].

3.3 Fuzzy Application

A not entirely novel concept, but less explored is applying fuzzy logic to protein structure prediction. Fuzzy logic systems incorporate a system of many-valued logic. An element in a set is given a value to describe its membership to different groupings; an example of this can be understood by visualizing an item that is partially true and partially false. Biological systems are not always rigid. In some protein structures, torsion angles change, and amino acids are not always in a constant position relative to the main structure. It has already been shown that computationally efficient implementations of *Fuzzy Decision Trees* (FDTs) [72] are competitive with other efficient learning algorithms to analyze and predict protein structure; and have been effective in recognizing and extracting sections in RNA structures. Another example is in the homology modeling and threading methods that are so commonly used in the supervised learning techniques; the application of fuzzy logic would assign a confidence value to the structures that the algorithms would be comparing as done in [73].

3.3.1 Fuzzy Logic

Fuzzy logic and fuzzy set operate under the notion of uncertainty: the idea that an element may not completely belong to one set and its operations impact that belief. As a reminder, in common logic systems (referred to as crisp) elements of a set are either inside the set or outside the set, true or false, 1 or 0. A fuzzy set A is defined as, a set $A = \{x_1, x_2, \ldots, x_n\}$ for $n = \| A \|$ and there is a membership function $\mu_A : X \longrightarrow [0, 1]$ where $0 \leq \mu_A (x_i) \leq 1$ for $x_i \in A$ for $i = 1, \ldots, n$. This logic extends to the same sets of operations given to crisp sets (e.g., intersection, union, etc.)[74].

3.3.2 Fuzzy SVMs

The fuzzification of SVMs replaces data as a function of parameters with data as a function of belief in class memberships. This fuzzification happens by applying a membership value to each input point of SVM, which requires reformulating the machine so that two inputs can make different contributions to the learning. The basic definition of an SVM is changed in the following manner: Let S be a set of training points of labeled data, containing $n - D$ dimensional vectors x_i and their associated classes' y_i, as well as their associated fuzzy membership value s_i:

$$x_1, y_1, s_1, \ldots, x_n, y_n, s_n \in \mathrm{I\!R}^D \times \{\pm 1\}$$

Where $x_i \in \mathbb{R}^D, y_i \in \{\pm 1\}$, and $0 \leq \sigma \leq s_i \leq 1$ with $i = 1, \ldots, n$. σ is treated as a minimum confidence value used to remove small membership function values. Let $z = \varphi(x)$ denote the corresponding feature space vector with a mapping φ from $\mathbb{R}^D$ to a feature space Z. With the incorporation of the fuzzy membership value s_i should be defined over x_i, the measure of error of that point ξ_i is adjusted by the membership. This is to say that the fuzzy membership must be included in the SVM to maximize the margin, and it is done with this function [75]:

$$\min \frac{1}{2}(w \cdot w) + C \sum_{i=1}^{n} s_i \xi_i$$

with constraints

$$y_i ((w \cdot x_i) + b) \geq 1 - \xi_i]$$

$$\xi_i \geq i = 1, \ldots, n$$

3.3.3 Adaptive-Network-Based Fuzzy Inference Systems

An *Adaptive-Network-based Fuzzy Inference System*, or *Adaptive Neural Fuzzy Inference System* (ANFIS) is an ANN developed by [76] based on the Takagi-Sugeno fuzzy inference system [77]. This method consists of 5 layers (Fig. 10):

- Layer 1: Each node is this layer is a membership function

$$O_i^1 = \mu_{A_i}(x)$$

  where x is the input to node i, A_i is the group or label of membership associated to this node.

- Layer 2: Consists of nodes, which multiply the incoming signals and send the product out to the next layer. Each nodes output represents the firing strength of a specific fuzzy rule.

- Layer 3: Each node normalizes the firing strength based on node i over the summation of all of the nodes.
- Layer 4: Each node i in this layer is a given the following node function

$$O_i^4 = \bar{w}_i f_i = \bar{w}_i \left(p_i x + q_i y + r_i\right)$$

  where $\bar{w}_i$ is the output of layer 3, and p_i, q_i, r_i is the consequent parameter set.

- Layer 5: Consist of a single node that computers the overall output as the summation of all of the incoming signals from the previous layer

$$O_i^5 = overalloutput = \sum_i \bar{w}_i f_i$$

3.3.4 Fuzzy Decision Trees

The *Fuzzy Decision Tree* (FDT) is another machine learning method that we have yet to discuss in this chapter. Considered one of the most popular methods, FDTs represent knowledge classification with a fuzzy rule set to make its classifications more robust and immune to data uncertainties [78]. In [78], there is an extension of the methodology described in [79] to develop an FDT that gives better accuracy than the original. The algorithm developed in [78] is as follows:

1. Generate the root node with a fuzzy set of all training data with membership value set to 1.
2. A node is a leaf node if the fuzzy set of data at the node satisfies any of the following conditions:
   (a) The number of objects in the node's data set is less than a given threshold.
   (b) All the objects in the node's fuzzy data set belongs to one class.
   (c) No more attributes are available for classification

3. The class assigned to a leaf node is the name of the class with the greatest membership value in the node fuzzy data set.

4. If the node does not satisfy any of the above conditions, then do what follows:

   (a) If the current node is the root node of the tree. Calculate the classification ambiguity of each attribute, and select the attribute with the smallest classification ambiguity as to the test attribute

   (b) Else the current node is not a root node. For each attribute not selected so far as a test attribute up in the path from the current node to the tree's root node, calculate the information gain, and select the attribute with the maximum information gain as a test attribute.

   (c) Divide the fuzzy data set at the node into fuzzy subset using the selected test attribute, with the membership value of an object in a subset set to the product of the membership value in parent node data set and the value selected attribute fuzzy term in the data at the node.

   (d) For each of the subsets, generates a new node with the branch labeled with the selected attribute's fuzzy term.

5. For each new generated node repeat recursively from step 2.

3.4 Conclusion

Research suggests GAs, ANNs, and SVMs are the most successful metrics in determining protein structure. Given that the result of ShapeDoc left questions that required some level of protein structure prediction, a solution with one of these machine learning models is an apparent need. Although the application of fuzzy methodologies to these machine learning techniques has obtained favorable results, the software suite this research is geared toward tool doesn't require a fuzzy method. Still, a fuzzy methodology can be applied to improve results potentially. In Zhong *et al.* [73] discovered that cluster membership functions in SVMs may not reveal the nonlinear sequence-to-structure relationship; because of this limitation, the use of SVMs for this research will not be used. The comprehensive tool that is the goal of this research still makes it possible to use an SVM to determine and classify primary protein structures as tertiary (or even secondary) structures; the lack of a more profound connection between the data and the prediction leads us to other alternatives.

Given that SVMs will not be used for this tool, it is believed that a hybridized method of GAs and ANN will lead to the most favorable answers to the questions that were left by the ShapeDoc algorithm. It is believed that the possibility of multiple favorable conformations that would be generated by a hybridized GA method could provide enough information to successfully introduce a level of flexibility in the proteins at a molecular level. This level of flexibility would not only provide information about the proteins themselves and the possible folding patterns that can arise but, more importantly, allow users to incorporate this molecular mobility in the ShapeDoc algorithm.

PART 4

HYBRID EVOLUTIONARY ALGORITHM FOR TERTIARY STRUCTURE PREDICTION

As discussed in the previous chapter, Genetic Algorithms are a popular method in determining protein conformation. By the model's very nature, a GA will successfully generate several different low energy protein confirmations. The information gained from these different conformations can provide insight into the potential mobility of the molecule. It is also worth noting the side effect of this methodology because these multiple low energy conformations provide us information that could help determine the natural folding pattern of the individual protein.

This section discusses the foundation of a custom genetic algorithm that would be used to predict protein tertiary structure from the primary structure. The customization of the genetic algorithm coming from integrating secondary structure predictors using Artificial Neural Networks is used to increase the model's overall accuracy. Incorporating neural networks used to determine secondary structures with the Genetic Algorithm search optimization would provide a way to find protein conformations that are energetically favorable and attempt to maintain the predicted secondary structure. This method has interchanging parts and can be performed with different machine learning techniques but what is necessary is having two classification models and a search optimization method. The application of this hybrid algorithm requires certain constraints on the methods used to find optimal solutions promptly.

This algorithm is divided into three sections. The last two are repeated until a stop condition occurs; this stop condition could be a certain number of desired outcomes with a high enough fitness score or a timing condition. The first section of the algorithm is population generation and initial secondary structure prediction. The second section is a reproduction. This reproductive section revolves around crossover and mutation, which are standard in evolutionary algorithms, notably the Genetic algorithm. An error correction and detection phase is also included in this step to eliminate false occurrences as in collisions that wouldn't be able to occur in any offspring in a real-world scenario. The third and final section is the fitness function, which is an evaluation of each solution. In this research, all that is required is a simple energy calculation equation being applied and a classifier that quickly determines the secondary structures formed by the atoms' positions in this 3-dimensional space along the backbone of the protein. The fitness function will have to occur at each generation for each new feasible confirmation; if factoring shape information is time demanding, the entire algorithm becomes in-feasible.

4.1 Population Generation and Baseline Secondary Structure Prediction

Population generation occurs by first extracting a protein's primary sequence information from a file. This file will usually be in FASTA format. In some test cases, it may also be removed from a PDB formatted file or a database of secondary structure assignments file (DSSP files also come in a PDB format). The purpose of this is to allow for multiple entry points were the only thing specifically required is the sequence of amino acids. DSSP files would provide the predetermined secondary structures, so a determination of those structures would not be necessary, as required with the input of a FASTA or a PDB file. If a PDB file is used, some sequences in the population will use the natural coordinates provided in the document since a low energy conformation is assumed in that standard. We generate an initial population of conformations from the protein sequence to create other conformations and determine a baseline secondary structure using a neural network that will be detailed in the following chapter. This baseline secondary structure predictor can be replaced with one of many accurate secondary structure predictors as long as they perform relatively quickly not to hang the algorithm.

Each protein confirmation is represented in the gene as an array of tuples, where each element is a tuple of the dihedral angles $(\phi, \psi,)$(figure 3.3) at that amino acids bond with the next element along the backbone chain (the series of covalently bonded atoms that together create the continuous chain). Every amino acid would be stored in the system based on the molecules, and their relative positions are given a starting point of its Carbon elements along the backbone. Ideally, these genes of tuple arrays would be analyzed for its feasibility, so before its fitness function is applied, the protein conformations are analyzed to verify there is no overlap; But since these overlaps would lead to extremely unstable energy confirmations, it is not necessary to eliminate the possibility from the pool of potential confirmation. The belief behind keeping these elements is that a minimal adjustment to an element's backbone could fix the overlap at the same time, creating a very energetically favorable conformation. Also, the fixing or correction of the overlap would require creating multiple corrected variants that could require dynamic programming to determine what the would all be, which would severely slow down the overall algorithm.

The initial population can be generated in one of two ways, both representing very commonly used evolutionary algorithms: the random generation of N different conformations is ranked and measured for fitness to determine how the following population will be generated (which would happen in a Genetic Algorithm). The generation of N conformations all of the same variety to be changed in small but different ways to find the optimal solution over many populations (which would happen in simulated annealing). This research will take the GA route and generate many different conformations to find many optimal solutions.

In previous sections, we mention the high accuracy of SVMs and ANNs that have been used to secondary structures amongst the protein. In this research, we shall use a modified Restricted Boltzmann Machines (RBM) that is specifically developed to find secondary structure information based on training from the DSSP. This method is chosen for its prediction timeliness but could be replaced by many other algorithms. Each amino acid in the initialization stage will be labeled with the secondary structure is supposed to be associated with; This labeling will only be used during the fitness determination to see if the geometric confirmation aligns with the predicted secondary structure.

4.2 Generating New Populations Through Crossover and Mutation

Mutation and crossover are how the primary factor in individuals change when generating a new population and why new conformations arise. Applying the proper type and amount of both crossover and mutation is very important to the optimal solutions' speed. In an early paper of Krasnogor, they determined in the HP that single point crossover could not transfer blocks of information well. Still, segment mutations acted as a powerful local search, which leads them to state the best combination of parameters had a small crossover probability and a high mutation and segment mutation probability [26].

In [2] Custòdio *et al.* used combinations of six different genetic operators to get solutions in both their HP model and their all-atom model. These six comprised a two-point crossover, a multi-point crossover, a segment mutation, an exhaustive search mutation, a local move, and a loop move. With exhaustive search mutation Custòdio *et al.* test all possible directions for a randomly selected base and keep only the best solution: in their paper, this demands four fitness functions and is developed to hybridize the GA and an exhaustive local search. Local moves and loop moves are ordered, changing mutations. Where local changes swap the positions of to random locations, loop movements switch places from two units that are five bases apart. Custòdio *et al.* also mention a repair function that can be applied when collisions occur. The purpose of the function is to fix invalid conformations when crossovers and mutations create them.

When performing crossover, different opinions have been expressed when trying to predict protein structure. In most methods, you would have a random selection with a bias towards the individuals with higher fitness; this simulates keeping the stronger individuals throughout the generation. There is some belief in the tournament method, where you take a constant number of random individuals and use the crossover to create new individuals. Neither one of these methods preserves the individuals' diversity in a given population from generation to generation. This point can be an issue because if you do not have a diverse population of highly fit individuals, you reduce the chance of converging on the optimum solution; also, answers converging to local minimums become a lot more likely. A solution to this problem is given in [2] based on the works of [80] and [81] called *crowding.* In this method, the new individual created a search over the parent generation for the most similar parent. Either the parent or child is kept for the next generation, depending on which individual has the lower energy. If the energies are the same, then either individual is given a 50% chance of being represented. This research will be applying this method, as well.

For the algorithm, we use a 90% chance of mutation where a random floating-point number between -0.1 and 0.1 is generated and added to ϕ and ψ angles separately. Each angle in each conformation has a 90% chance of being adjusted. We also use a 20% chance of crossover; at random, one of the above methods is selected, and the crossover is applied to the previous top 10 genes, and ten genes were chosen at random. At the end of this process, each new conformation is added to a pool to have that individual gene's fitness tested and ranked.

4.3 Fitness Test Application

The quality of the model found with a GA depends on the calculations' speed and the quality of the energy function that is being searched for an optimum. This dependency issue is the crux of the efficacy of the research. The fitness function, in terms of this application, is two-fold: the first is the energy conformation calculation used to determine how favorable a chemical confirmation is. The second compares the observed or determined, apparent secondary structures that appear in the generated conformations and the initial predicted structure. The comparison of the two structure is weighted to show favor to the structure similarity; This comparison is because current research shows that secondary structure predictors' accuracy is more significant than exhaustive search methods like evolutionary algorithms.

In terms of calculating the energy conformation, the calculation speed can be estimated from the number of energy evaluations the GA has to perform as the system evolves from generation to generation. Also, from [7] we know when testing against a known structure, a prediction is considered to be similar if:

- secondary structure elements agree with the known structure
- the orientation of those secondary structures are within $20°$ of the ones in the known structure
- the length of the secondary structure elements different by no more than two residues of the known structure

To successfully determine the three-dimensional geometric structure correlates to the proper secondary structure, we must first have a model to recognize the shape created by the confirmation. This will need to be able to take in the dihedral angles along the protein backbone as well as the size of the amino acid so the backbone can be created in three-dimensional space; the entire amino acid isn't required to be recreated, just the backbone of conformation as the whole.

4.4 Secondary Structure Classification

The GA described in this section is a relatively standard outside of the one portion that checks for the present secondary structure. Without this one feature, this GA isn't noteworthy, but the prediction model's inclusion makes this algorithm relatively novel. The introduction of the classification requirement also brings into issue feasibility requirements. Suppose this algorithm is to generate thousands of genes over several generations. In that case, a classification tool must determine this classification rapidly not to hold up the algorithm and make it unbearably slow. GA's can be easily parallelized and distribute work amongst several separate workers by providing specific metrics. The goal is to have an algorithm efficient enough to run in single-core non-networked machines and still get results in a reasonable amount of time, despite that fact.

Another goal of these classifiers is to determine if a certain level of information can be drawn out of the several thousand configurations the models would or could be used for training. By their very nature, deep learning models determine patterns amongst data points to make decisions about the problems presented to them. Suppose a typical deep learning tool like the Restricted Boltzmann Machine is used to make these determinations of secondary structures. In that case, the model's analysis could one day further reveal information about the nature of the amino acid configurations that lead to the secondary structures we observe. In the next chapter, George Hinton's Restricted Boltzmann Machine is introduced. The original and custom RBMs application is applied to the secondary structure prediction problem presented by this hybridized GA.

PART 5

RESTRICTED BOLTZMANN MACHINE APPLICATIONS TO SECONDARY STRUCTURE PREDICTION

In this section, we introduced the Restricted Boltzmann Machine, most often referred to as the RBM. A history of the RBM and why it's deemed to be useful is provided. Customizations of the standard RBM algorithm are also introduced and are used to analyze the primary structures of proteins to determine if secondary structures could be accurately predicted. This section of the research aims to develop a set of classifiers that could accurately and quickly predict the secondary structures that should appear given an amino acid sequence and predetermined information about each amino acid. A data set was created using the DSSP program's entirety, designed by Wolfgang Kabsch and Chris Sander. DSSP is a database of secondary structure assignments for every protein that is in the Protein Data Bank.

Throughout this section, the Boltzmann Machine's history and evaluations and the Restricted Boltzmann Machine will be provided. Next, there will be an introduction of customizations of the standard Restricted Boltzmann Machine algorithm; as well as an evaluation of these customizations and why they are viewed as potential improvements. Finally, a discussion of applying the standard RBM and the customized methods of the RBM on protein data will be discussed.

5.1 Boltzmann Machine

The original learning algorithm for the Boltzmann Machine (BM), also called stochastic Hopfield network with hidden units, required randomly initialized Markov chains to approach equilibrium distributions to estimate both data-dependent and independent expectations connected pair of binary variables would both be on. A BM, much like a Hopfield Network, is a network of nodes where each edge between them represents an adjustable weight wherein a combination of the Gibbs free energy of the model can be calculated. The purpose is to learn essential and possibly hidden aspects of an unknown probability distribution based on samples from this distribution. The embedding of BMs into the framework of probabilistic graphical models is said to provide immediate access to a wealth of theoretical results and well-developed algorithms [82].

We calculate the energy of the BM with the following equation(5.1):

$$E = -\left(\sum_{i=1}^{n}\sum_{j=1}^{n} w_{i,j} s_i s_j + \sum_{i=1}^{n} \theta_i s_i\right) \tag{5.1}$$

Where E is the Gibb's free energy of the model; n represents the number of nodes in the model; $w_{i,j}$ represents the weighted edge connecting nodes i and j; s_i and s_j represent the binary state $0, 1$ of node i and j respectively; and θ_i represents the bias of node i in the global energy of the model.

We use this same energy metric in the training of the BM in which, in our nodes, we classify them as hidden and visible. In training, our visible nodes are set to values from our training set. We use the weighted model to reproduce the original set, continually adjusting the edges' weights until this task can be performed and recreate the initial input. This step is usually done with contrastive divergence. This original learning process was considered difficult and time-consuming, often resulting in exponential time complexity. The learning process is simplified by imposing restrictions on the network topology; a machine of this type is coined the Restricted Boltzmann Machine (RBM).

5.2 Restricted Boltzmann Definitions

RBMs are powerful generative models for labeled and unlabeled data based on latent variables, which are used to model an input distribution. Though RBMs are considered by many to be the first step of another learning algorithm because they are widely used to either provide a preprocessing of the data or initialization for neural networks' parameters. In

[83] it is shown that RBMs can provide a self-contained framework for deriving competitive non-linear classifiers. RBMs are often applied to problems involving high dimensional data such as images and text and have been successful in classification problems either as feature extractors [84]. Their conditional form can be used to model high-dimensional temporal sequences such as video or motion capture data. Their most important use is as learning modules that are composed to form deep belief nets.

Mathematically, RBM is usually represented by two collections of stochastic, typically binary, connected by symmetrically weighted edges. The two collections of stochastic nodes are called the visible layer, and the hidden layer; where the visible layer is treated as a binary input and the hidden layer are treated as feature detectors.

$$E = -\sum_{i \in visible} a_i v_i - \sum_{j \in hidden} b_j h_j - \sum_{i,j} v_i h_j w_{i,j} \tag{5.2}$$

In equation (5.3) we see the energy function of the Restricted Boltzmann Machine. In this function v_i and h_j are the (often binary) representations of the visible and hidden nodes i and j respectively; a_i and b_j are the biases for the visible and hidden nodes i and j respectively; and $w_{i,j}$ is the weight between the visible and hidden nodes i and j. In the created graph the is a probability associated for every possible pair of visible and hidden nodes being one of its binary options is determined by the following equation (**??**):

$$p\left(v,h\right) = \frac{e^{-E(v,h)}}{Z} \tag{5.3}$$

Where Z is defined as a partion function calculated by summing every possible pair of visible and hidden node (equation (5.4)):

$$Z = \sum_{v,h} e^{-E(v,h)} \tag{5.4}$$

And the probability of the graph assigning a value to a visible node, v, is given by summing over all possible hidden vectors (equation (5.5))

$$p(v) = \frac{1}{Z} \sum_{h} e^{-E(v,h)} \tag{5.5}$$

The Restricted Boltzmann Machine uses this system of equations the train or models around accurate representations of the training data. This system is done by adjusting the weights and biases to lower the entire model's energy. The derivative of the log probability of a training vector with respect to a weight is given with the following equation (5.6)

$$\frac{\delta \log p(v)}{\delta w_{i,j}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model} \tag{5.6}$$

where the angle brackets represents the expected values of the input vector versus what the model actually produces, which leads to the learning rule that impacts the change in the weighted edges in the model (equation (5.7))

$$\Delta w_{i,j} = \epsilon \left(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model} \right) \tag{5.7}$$

where ϵ represents the learning rate which generally represent a constant value to control the amounts of change amongst the weights.

When training the RBM, the RBM's visible nodes are set to the values provided by a training vector. The binary nodes of the hidden units are all computed in parallel using the following equation (5.8):

$$p(h_j = 1|v) = \sigma \left(b_j + \sum_i v_i w_{i,j} \right) \tag{5.8}$$

where $\sigma(x)$ is the sigmoid function $1/(1+\exp(x))$. After all hidden nodes are computed, the visible layer is then reconstructed using equation (5.9):

$$p(v_i = 1|h) = \sigma \left(a_i + \sum_j h_j w_{i,j} \right) \tag{5.9}$$

Using the values provided by the reconstructions versus the expected values provided by our data is what allows for equation (5.7) to adjust the energy of the entire model. Following these steps until you can reconstruct the data is called contrastive divergence.

In this research, updated models of this contrastive divergence step were performed in addition to the standard RBM construction. The first update was instead of performing the contrastive divergence step; the RBM leverages the full Ising model to achieve a 14 fold speed up in execution. The second update was a novel recreation of the RBM model called the clique RBM (cRBM). Since information is lost from the conversion from the full BM to its restricted counterpart, the RBM with the clique RBM's restrictions on the RBM hidden assets are loosened by allowing a set of connections between nodes of the same level. The time-consuming nature of the full BM doesn't have to be adapted with this model because though this methodology does increase the complexity, we consider the information gain is worth it in small cliques.

5.3 Clique Restricted Boltzmann Machine

Restricted Boltzmann machines avoid connections between nodes of the same type, i.e., between hidden nodes or visible nodes. The belief with the clique Restricted Boltzmann Machine (cRBM) is that limitations can be placed upon the connections between units of the same variety and still achieve the speed up and efficiency necessary that the RBM provides over the Boltzmann Machine that allows it to be useful.

In [84] we find that we can derive the following energy equation by separating the stochastic binary units into their respective type and the types of connections between them:

$$E\left(v,h;\theta\right) = -\frac{1}{2}v^{T}Lv - \frac{1}{2}h^{T}Jh - v^{T}Wh$$

where $\theta = \{W, L, J\}$ are the weights or model parameters for visible-to-hidden, visible-visible , and hidden-hidden symmetric interaction terms respectively. Diagonals along L and J are set to 0 because no node has a weighted connection to itself. This equation leads to weight determination being as follows:

$$\Delta w_{ij} = \epsilon \left(\langle v_i h_j \rangle_v - \langle v_i h_j \rangle_{model} \right)$$

$$\Delta L_{ij} = \epsilon \left(\langle v_i V_j \rangle_v - \langle v_i V_j \rangle_{model} \right)$$

$$\Delta J_{ij} = \epsilon \left(\langle h_i h_j \rangle_v - \langle h_i h_j \rangle_{model} \right)$$

In the cRBM, connections between elements of the same unit type are restricted to have a set number of connections, N_v for the number of connected nodes for the visible units and N_h for the hidden. These connections are further restricted to form cliques of size N_v for visible units and N_h for the hidden. In the respective matrix for these connections all weights to nodes outside of the clique are set to 0. With these restrictions a full BM would be a lRBM with $N_v = \#ofVisibles - 1$ and $N_h = \#ofHiddens - 1$ and a standard RBM $N_v = 0$ and $N_h = 0$.

Mathematically, the cRBM is more feasible because the time it takes to add one connection in the worst-case scenario with no additional speed up increases the time constraints by a factor of 2. Even though this demand is not insignificant, it is still very feasible. One of the purposes of this research is to see what number of connections the time demand becomes in-feasible for particular problems and examine the tradeoff of information gained and time complexity added.

5.4 Restricted Boltzmann Protein Analysis

For this research, we apply the RBM application to amino acid sequences where $\alpha-$helix structures are present. This application is made by training a standard RBM as well as a full Ising model RBM to recognize an $alpha-$helix on the corresponding sequence. The application of the full Ising model allows for fast training to recognize new structures. To do this, firstly, we scrape the PDB for all DSSP files to create a training set. Using these files, we parse out sections of the protein sequence with an $\alpha-$helix structure where there are six types. We must also convert our data to be something that can be input into an RBM, meaning that these protein sequences must be converted to a binary string.

To make this conversion, we choose to look at the twenty sequential amino acids' windows in the primary structure. In those twenty, we only train against character string that contains at least ten amino acids the correlate to one of the $\alpha-$helix structures. Each amino acid is given its binary string equivalent containing digits representing ancillary information about that specific protein, i.e., hydrophobic/hydrophilic. This conversion allows the RBM to have a specific number of visible nodes to reflect the 20 character string.

Implementation of the algorithm also starts with a standard RBM built to the specification of [84]. This was done using python v3 without using machine learning libraries such as Theano, SciKit Learn, TensorFlow, etc. This was done to compare speed and accuracy between a standard binary RBM that uses a contrastive divergence and a more streamlined version that leverages the Ising model. Leveraging the Ising model allows for more direct minimization of the system's free energy to approximate the minimum gradient. Also, as is standard for RBMs, the weight matrix can be initialized to any uniform random distribution.

It must be said that the RBM is assumed to have a convex weight matrix, where there is one global solution for a class of secondary structures. This concept leads to the theory that any RBM trained on a single helix is equally valid as a solution to the entire training set like any other for the same training set. This solution is an approximation. Due to the convexity, the final weight matrices are averaged together to arrive at one generalized solution that works well for the entire training set. The generalized solution is a strong assumption, but as all the underlying training data describes the same type of physical structure, it follows that training continues iteratively in the standard model that any training sample would eventually approach a similar solution.

5.5 Results and Conclusions

Using a standard and Ising RBM, we were able to train models to recognize an $\alpha-$helix with a fair degree of accuracy. In the testing, both RBMs recognized both the $\alpha-$helix structures that were trained for (160 members) and some that were not. While testing, we noticed that using such a small sample in training that sequences that are significantly different from those in the training set werent well detected. These results provide us the belief that RBMs are a useful way to encode higher-order descriptions of protein structure. Clustering independently supports the belief that there are underlying building blocks and rules, i.e., a primer for protein structure folding, that would allow for very fast folding based on our protein binary string creation and identify interchangeable structures for protein assemblage quickly.

PART 6

WEISFEILER-LEHMAN RELATIONSHIP GRAPHS

This final section of this research will describe the portion of the software suite used to analyze proteins and other molecules' interactions. In two of the previous sections (part 2 and 4), when the shape doc tool and the hybrid evolutionary algorithm is discussed, several potential interactions of various types are collected with no more in-depth level of analysis. A plan of analysis for this information generated is required to ensure that information generated is adequately leveraged. The analysis plan being proposed in this section is a modified application of the Weisfeiler-Lehman Graph Kernel algorithm that was proposed by Shervashidze et al., [85].

Recall from the introductory chapter and subsection 1.2.3, where an introduction to molecular interactions is provided; there are a wide array of molecular interactions between proteins and other molecules, not all of which are considered physical interactions. Each distinct interaction a protein exhibits forms the protein's overall function; the classification and reference of those individual types of interactions develop an accurate understanding and profile of that protein that can be compared against other proteins for similarity. This type of data can easily be stored in a graph of proteins and other macromolecules that are connected by directional edges labeled with the relationship classification it is describing.

Using the knowledge of the practical relationships between the data points in our graph of protein connections, we can pull graphs of different proteins' networks and compare them to determine the similarity between proteins and draw more information. Very similar techniques are used in social media networks to classify and profile the users in the system. In this research, we plan to compare us accounting for patterns of relationships to determine a more accurate comparison between sub-graphs that have multiple different types of edges; we call this methodology the Weisfeiler-Lehman Relationship Graph Kernel (WLRGK) algorithm. The system described in this research information about these relationships is stored in a multi-graph where each relationship is a labeled directed edge between two entities.

We would test this algorithm again in an ideal experiment, a very detailed and verified database of protein interaction information. Although our overarching research is trying to develop this, a robust version of this database does not currently exist. Due to this drawback, a proof of concept for this algorithm is to be performed on publicly available social media data where the relationship can quickly be drawn from the information stored. Because the relationships of proteins themselves occur in a very similar manner to how they study the interconnected relationship between users in a social network like Facebook, Instagram, or Twitter, we feel this is an appropriate substitute to prove our concept WLRGK algorithm.

Throughout this section, we will detail all of the preliminary definitions required to fully understand what this research is attempting to accomplish and see what is novel about the WLRGK. After, we will define the WLGRK Algorithm and detail its use cases, and whats separates it from a standard WL or WL Graph Kernel algorithm. Finally, we will describe what a future experiment would look like that test the validity of the WLGRK versus the WL Graph Kernel algorithm; this description will highlight the outline of the investigation that has not been performed previous to writing this section.

6.1 Preliminary Definitions

In this section, we are introducing terms that have not been previously defined. These terms are not expected to be understood by someone with general knowledge of computer science, biological or bioinformatics research (much less the layman trying to learn more about the topic). We will provide the definition and context to several of them required to understand this section. These terms include graph, multi-graph, graph kernel, and Weisfeiler-Lehman algorithm.

6.1.1 Graphs and Multi-Graphs

A graph is a general term in computer science as well as mathematics. For this research, we will be using the computer science definition of a graph; in which, a graph is an abstraction used to represent a set of objects where links or specific relationships connect some pairs of items; in the definition, the connected objects are named nodes or vertices (where the singular is vertex), and the relationships, or links, are called edges. Formally, a graph is a pair of sets (V, E), where V is the set of vertices and E is the set of edges connecting the pairs of vertices.

There are several classifications of graph dependant upon restrictions you place on the edges that connect vertices. The three most commonly taught classification of graphs are directed, undirected, and weighted graphs. A directed graph is a graph in which all the edges point in a single direction, i.e., a directed edge from V_1 to V_2 suggest you can only travel from vertex V_1 to vertex V_2; these edges are also called unidirectional. An undirected graph is a graph in which all the edges do not point in any specific direction. In an example, an undirected edge from V_1 to V_2 suggest you can both travel from vertex V_1 to vertex V_2 and vice versa; these edges are also called bi-directional. Finally, we have weight graphs. Each edge is assigned a weight or cost in a weighted graph, suggesting a penalty for choosing to travel one edge versus another.

In this research, we use a very particular type of graph that's not as commonly taught; this graph is called the multigraph. In graph theory, a multigraph is a graph that allows for multiple edges to exist between two vertices; These are also called parallel edges. The same classifications of graphs can also be applied to multigraphs, i.e., directed multigraphs, undirected multi-graphs, and weighted multigraphs. For our research, we will be using directed multigraphs, seeing that each edge in our graph will represent a relationship between two vertices, and there may be several types of one-way relationships.

6.1.2 Graph Kernels

Graph kernels are a relatively new area of study. The idea graph kernels were introduced in 1999 [86]. Their paper introduced this method of constructing kernels on sets whose elements are discrete structures like graphs. The term graph kernel was introduced in 2002 [87], where the purpose of the graph kernel is primarily to allow for the characterization and comparison of graphs and subgraphs (a graph contained within another graph). When the term graph kernel was introduced, the study's suggested application was on the internet and the hyperlinks that reference individual web pages.

In [85], graph kernels are defined as instances of a method of determining kernels on graphs by comparing all pairs of decompositions of the graph; A decomposition relation R decomposes that graph into any of the subgraphs and the remaining portion of the graph. It is noted that graph kernels are convolutional kernels on pairs of graphs and not the vertices that comprise them. The kernelization of graphs allows for applying the kernel method on graphs, i.e., classification, feature extraction, clustering, etc.

Algorithm 1 One iteration of the 1-dim. Weisfeiler-Lehman test of graph isomorphism

```
1. Multiset-label determination
      • For $i = 0$, set $M_i(v) := l_0(v) = \lambda(v)^2$
      • For $i > 0$, assign a multiset-label $M_i(v)$ to each node $v$ in $G$ and $G'$ which consists
   of the multiset $l_{i1}(u)|u \in N(v)$.
2. Sorting each multiset
      • Sort elements in $M_i(v)$ in ascending order and concatenate them into a string $s_i(v)$
      • Add $l_{i1}(v)$ as a prefix to $s_i(v)$ and call the resulting string $s_i(v)$
3. Label compression
      • Sort all of the strings $s_i(v)$ for all $v$ from $G$ and $G_0$ in ascending order.
      • Map each string $s_i(v)$ to a new compressed label, using a function $f : \sum * \to \sum$
   such that $f(s_i(v)) = f(s_i(w))$ if and only if $s_i(v) = s_i(w)$.
4. Relabeling
      • Set $l_i(v) := f(s_i(v))$ for all nodes in $G$ and $G'$
```

6.1.3 Weisfeiler-Lehman Graph Kernels

Firstly, the Weisfeiler-Lehman (WL) algorithm is a graph labeling method that determines vertex ordering based on graph topology. In the WL we must add on to our definitions of a graph as they do in Shervashidze et al., [85]. In their paper, Shervashidze et al. define a graph as a triplet (V, E, λ) where V and E have their standard definition of vertices and edges, and λ is a function that assigns labels to vertices in the graph. They describe the Weisfeiler-Lehman test of graph isomorphism algorithm, which is defined in Algorithm 1.

When the WL algorithm is applied to graph kernels, it allows one to determine the isomorphic relationship in the graph itself more efficiently. This determination is done by comparing the labels present in the final iterations of the algorithm. A speed-up in the determination of similarity between structures by a significant margin and in 1 they also show multiple different algorithmic tweaks that can be performed for other result types. Only the vertices and the edge in between them are required to update the label in these algorithms. Still, all edges themselves are treated pretty uniformly, not considering the type of relationship it may be. This point is where our concept of the Weisfeiler-Lehman relationship graphs kernels comes into play.

6.2 Weisfeiler-Lehman Relationship Graph Kernel

In the WLRGK, we require a directed multigraph where each edge between vertices represents a specific relationship between the vertices in question. A directed multigraph is chosen not only because an undirected multigraph can be fully represented in a directed multigraph, but also a directed multigraph would be better at representing diverse one-way relationships that can exist in many general spaces. The WLRGK still uses the WL labeling technique on a graph kernel, but updates the algorithm to relabel based on the pattern of relationships that is being searched against. This update would allow us to create new topologies based on the relationship patterns of the specific edges between vertices, giving way to finding more profound, more meaningful patterns in the relationship between nodes in a graph. It is to be noted that a standard WL Graph Kernel would provide the same results as a WLRGK with every relationship chosen each time.

6.3 Experiments

When applying the WLRGK to the information stored in our system, we would determine the similarity between proteins because of the information they hold and the topology of their graph kernels and relationship sub-graphs. For this research, an ideal system would be to train and test our WLRGK on a database of protein relationships peer-reviewed and robust. This dataset is one of the desired outputs of our overall research. At the time of writing, this dataset does not currently exist to the author's knowledge. As proof of concept, this analysis is suggested to be performed on publicly available social media information. In an experiment, first, studying and understanding the types of relationships available in the dataset must be done; then, a multigraph is created. It is also suggested that the graph should be stored in a graph database like neo4j. Moreover, a tool that allows you to do this in a distributed fashion like GraphFrames is also heavily suggest.

In my research, the dataset that was used as the Stack Exchange cstheory meta dump from July 2019 that dates to August 2010. There is already a system of users who express positive and negative sentiments about the material being discussed through upvoting and downvoting, which gives three levels of relationship; sentiment analysis of the text can also be accounted for to provide different levels of relationship. The initial test shows favorable results, but during writing this document had not been deeply studied. This analysis suggests that what is performed is a controlled test where WL Graph Kernel and WLRGK are used to analyze the system. Then edges are removed to determine if both systems would predict what type of relationship would be missing.

PART 7

CONCLUSION

This research has shown that producing a software suite of these novel applications is quite feasible and should be pursued, improved upon, and maintained. Many of the algorithms studied were proof-of-concept tools that validated that these novel concepts work well, but they were not designed to operate as optimally as possible. Each algorithm discussed can be parallelized, made, distributed, and integrated into a web application, allowing for the creation of a central database that could be added to and accessed by the public and grown to the point where the data it provides should be vetted and improved. Each tool discussed needs a wealth of real world samples to create a valid data set to be studied; even considering the GA generates many potential instances.

7.0.1 Future Works

Soon, several of these applications will be applied to different data sets to be peer-reviewed individually. A full analysis and comparison of the cRBM will be performed to find the optimal clique size can generally be determined to balance the information gained versus time complexity can be determined. A comparison of the RBM and cRBM will be made using the MNIST handwriting dataset, seeing that the RBM has been thoroughly tested on this dataset in several papers and tutorials. We will soon also use stacked Ising RBM and cRBM to create deep belief networks for its analysis to determine if each layer can be analyzed to determine any currently unknown patterns about these secondary structures. The Weisfeiler-Lehman Relationship Graph Kernel's application and results will be applied to the Stack Exchange cs theory data dump and a smaller database of the molecular interactions of proteins. Our database's growth can be applied to our interaction network, where we can then classify the type of relationships we stored based on these results. There are also plans on applying the WLRGK to criminal and spam activity on apps like Telegram to see if anonymous users can be classified based on responses and message patterns.

7.0.2 Outside Questions and Application

It is well worth stating that the research done in this document, although applied to proteins, can have applications in several other areas. The Shape Doc Algorithm has drawn much interest in the area of geology and the studying of information in underground caves. The idea of the cRBM, if feasible for the system, could be applied to any deep learning tool that uses convolutional neural networks. The WLRGK has deep levels of application and integration with analysis tools like sentiment analysis as well as word-to-vector tools that could lead to a diverse set of problems in its own right in the field of language analysis. Additionally, the idea of distributing the tools used in this research amongst a parallel network could significantly speed up the timing limitations and further advance the benefit of each algorithm. When writing this document, there was a personal focus on the advancement of medicines and the study of genetic evolution; that does not limit the scope of which these tools can be applied.

REFERENCES

[1] T. D. K. A. Bode, W., "The refined 1.9-a x-ray crystal structure of d-phe-pro-arg chloromethylketone-inhibited human alpha-thrombin: structure analysis, overall structure, electrostatic properties, detailed active-site geometry, and structure-function relationships." *Protein Sci.*, vol. 1, no. 4, pp. 426–471, 1992.

[2] F. L. Custódio, H. J. Barbosa, and L. E. Dardenne, "A multiple minima genetic algorithm for protein structure prediction," *Applied Soft Computing*, vol. 15, pp. 88–99, 2014.

[3] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of molecular biology*, vol. 48, no. 3, pp. 443–453, 1970.

[4] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences," *Journal of molecular biology*, vol. 147, no. 1, pp. 195–197, 1981.

[5] T. Blundell, B. Sibanda, M. Sternberg, and J. Thornton, "Knowledge-based prediction of protein structures," *Nature*, vol. 326, p. 26, 1987.

[6] N. Mansour, F. Kanj, and H. Khachfe, "Enhanced genetic algorithm for protein structure prediction based on the hp model," *Search Algorithms Appl*, vol. 3, p. 69, 2011.

[7] T. Dandekar and P. Argos, "Ab initio tertiary-fold prediction of helical and non-helical protein chains using a genetic algorithm," *International Journal of Biological Macromolecules*, vol. 18, no. 1, pp. 1–4, 1996.

[8] S. Crivelli, E. Eskow, B. Bader, V. Lamberti, R. Byrd, R. Schnabel, and T. Head-Gordon, "A physical approach to protein structure prediction," *Biophysical journal*, vol. 82, no. 1, pp. 36–49, 2002.

[9] K. Olszewski, L. Piela, and H. Scheraga, “Mean field theory as a tool for intramolecular conformational optimization. 3. test on melittin,” *The Journal of Physical Chemistry*, vol. 97, no. 1, pp. 267–270, 1993.

[10] M. Ben-David, O. Noivirt-Brik, A. Paz, J. Prilusky, J. L. Sussman, and Y. Levy, “Assessment of casp8 structure predictions for template free targets,” *Proteins: Structure, Function, and Bioinformatics*, vol. 77, no. S9, pp. 50–65, 2009.

[11] R. Unger and J. Moult, “Finding the lowest free energy conformation of a protein is an np-hard problem: proof and implications,” *Bulletin of Mathematical Biology*, vol. 55, no. 6, pp. 1183–1198, 1993.

[12] ——, “Genetic algorithms for protein folding simulations,” *Journal of molecular biology*, vol. 231, no. 1, pp. 75–81, 1993.

[13] L. W. S. Wang, Renxiao; Lai, “Further development and validation of empirical scoring functions for structure-based binding affinity prediction,” *Journal of Computer-Aided Molecular Design*, vol. 16, pp. 11–26, 2002.

[14] M. O. Taha, Y. Bustanji, A. G. Al-Bakri, A.-M. Yousef, W. A. Zalloum, I. M. Al-Masri, and N. Atallah, “Discovery of new potent human protein tyrosine phosphatase inhibitors¡ i¿ via¡/i¿ pharmacophore and qsar analysis followed by¡ i¿ in silico¡/i¿ screening,” *Journal of Molecular Graphics and Modelling*, vol. 25, no. 6, pp. 870–884, 2007.

[15] O. Trott and A. J. Olson, “Autodock vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading,” *Journal of computational chemistry*, vol. 31, no. 2, pp. 455–461, 2010.

[16] R. D. Taylor, P. J. Jewsbury, and J. W. Essex, “A review of protein-small molecule docking methods,” *Journal of computer-aided molecular design*, vol. 16, no. 3, pp. 151–166, 2002.

[17] D. A. Pearlman, D. A. Case, J. W. Caldwell, W. S. Ross, T. E. Cheatham III, S. DeBolt, D. Ferguson, G. Seibel, and P. Kollman, "Amber, a package of computer programs for applying molecular mechanics, normal mode analysis, molecular dynamics and free energy calculations to simulate the structural and energetic properties of molecules," *Computer Physics Communications*, vol. 91, no. 1, pp. 1–41, 1995.

[18] B. R. Brooks, R. E. Bruccoleri, B. D. Olafson, D. J. States, S. Swaminathan, and M. Karplus, "Charmm: A program for macromolecular energy, minimization, and dynamics calculations," *Journal of computational chemistry*, vol. 4, no. 2, pp. 187–217, 1983.

[19] G. Jones, P. Willett, R. C. Glen, A. R. Leach, and R. Taylor, "Development and validation of a genetic algorithm for flexible docking," *Journal of molecular biology*, vol. 267, no. 3, pp. 727–748, 1997.

[20] T. J. Ewing and I. D. Kuntz, "Critical evaluation of search algorithms for automated molecular docking and database screening," *Journal of Computational Chemistry*, vol. 18, no. 9, pp. 1175–1189, 1997.

[21] B. Kramer, G. Metz, M. Rarey, and T. Lengauer, "Part 1–docking and scoring: Methods development-ligand docking and screening with flexx," *Medicinal Chemistry Research*, vol. 9, no. 7-8, pp. 463–478, 1999.

[22] H. A. Gabb, "Ftdock (v1. 0)," 1997.

[23] A. Z. Dudek, T. Arodz, and J. Galvez, "Computational methods in developing quantitative structure-activity relationships (qsar): a review," *Combinatorial chemistry & high throughput screening*, vol. 9, no. 3, pp. 213–228, 2006.

[24] Z. F. G. G. T. B. H. W. I. S. P. B. H.M. Berman, J. Westbrook, "The protein data bank," 2000.

[25] A. Pullman and B. Pullman, “Molecular electrostatic potential of the nucleic acids,” *Quarterly reviews of biophysics*, vol. 14, no. 03, pp. 289–380, 1981.

[26] N. Krasnogor, W. Hart, J. Smith, and D. Pelta, “Protein structure prediction with evolutionary algorithms,” 1999.

[27] A. L. Patton, W. F. Punch III, and E. D. Goodman, “A standard ga approach to native protein conformation prediction.” in *ICGA*, 1995, pp. 574–581.

[28] S. Schulze-Kremer, “Genetic algorithms for protein tertiary structure prediction,” in *Machine Learning: ECML-93*. Springer, 1993, pp. 262–279.

[29] J. T. Pedersen and J. Moult, “Genetic algorithms for protein structure prediction,” *Current Opinion in Structural Biology*, vol. 6, no. 2, pp. 227–231, 1996.

[30] S. Sun, “Reduced representation model of protein structure prediction: statistical potential and genetic algorithms,” *Protein Science*, vol. 2, no. 5, pp. 762–785, 1993.

[31] K. A. Dill, K. M. Fiebig, and H. S. Chan, “Cooperativity in protein-folding kinetics.” *Proceedings of the National Academy of Sciences*, vol. 90, no. 5, pp. 1942–1946, 1993.

[32] L. Toma and S. Toma, “Contact interactions method: A new algorithm for protein folding simulations,” *Protein Science*, vol. 5, no. 1, pp. 147–153, 1996.

[33] K. Yue and K. A. Dill, “Forces of tertiary structural organization in globular proteins,” *Proceedings of the National Academy of Sciences*, vol. 92, no. 1, pp. 146–150, 1995.

[34] R. F., “Two theorems of statistical separability in the perceptron, mechanization of thought processes,” *Proceedings of a symposium on the mechanisation of thought processes*, pp. 421–456, 1959.

[35] S. K. Kushwaha and M. Shakya, “Neural network: A machine learning technique for tertiary structure prediction of proteins from peptide sequences,” in *Advances in Computing, Control, & Telecommunication Technologies, 2009. ACT’09. International Conference on*. IEEE, 2009, pp. 98–101.

[36] N. Qian and T. J. Sejnowski, "Predicting the secondary structure of globular proteins using neural network models," *Journal of molecular biology*, vol. 202, no. 4, pp. 865–884, 1988.

[37] C. M. Bishop, *Neural networks for pattern recognition.* Oxford university press, 1995.

[38] L. H. Holley and M. Karplus, "Protein secondary structure prediction with a neural network," *Proceedings of the National Academy of Sciences*, vol. 86, no. 1, pp. 152–156, 1989.

[39] D. Kneller, F. Cohen, and R. Langridge, "Improvements in protein secondary structure prediction by an enhanced neural network," *Journal of molecular biology*, vol. 214, no. 1, pp. 171–182, 1990.

[40] P. Stolorz, A. Lapedes, and Y. Xia, "Predicting protein secondary structure using neural net and statistical methods," *Journal of Molecular Biology*, vol. 225, no. 2, pp. 363–377, 1992.

[41] B. Rost and C. Sander, "Prediction of protein secondary structure at better than 70% accuracy," *Journal of molecular biology*, vol. 232, no. 2, pp. 584–599, 1993.

[42] ——, "Combining evolutionary information and neural networks to predict protein secondary structure," *Proteins: Structure, Function, and Bioinformatics*, vol. 19, no. 1, pp. 55–72, 1994.

[43] J.-M. Chandonia and M. Karplus, "New methods for accurate prediction of protein secondary structure," *Proteins: Structure, Function, and Bioinformatics*, vol. 35, no. 3, pp. 293–306, 1999.

[44] G. Pollastri and A. Mclysaght, "Porter: a new, accurate server for protein secondary structure prediction," *Bioinformatics*, vol. 21, no. 8, pp. 1719–1720, 2005.

[45] G. Pollastri, D. Przybylski, B. Rost, and P. Baldi, "Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and pro-

files," *Proteins: Structure, Function, and Bioinformatics*, vol. 47, no. 2, pp. 228–235, 2002.

[46] K. Lin, V. A. Simossis, W. R. Taylor, and J. Heringa, "A simple and fast secondary structure prediction method using hidden neural networks," *Bioinformatics*, vol. 21, no. 2, pp. 152–159, 2005.

[47] L. J. McGuffin, K. Bryson, and D. T. Jones, "The psipred protein structure prediction server," *Bioinformatics*, vol. 16, no. 4, pp. 404–405, 2000.

[48] C. Cole, J. D. Barber, and G. J. Barton, "The jpred 3 secondary structure prediction server," *Nucleic acids research*, vol. 36, no. suppl 2, pp. W197–W201, 2008.

[49] J. A. Cuff and G. J. Barton, "Application of multiple sequence alignment profiles to improve protein secondary structure prediction," *Proteins: Structure, Function, and Bioinformatics*, vol. 40, no. 3, pp. 502–511, 2000.

[50] S. K. Riis and A. Krogh, "Improving prediction of protein secondary structure using structured neural networks and multiple sequence alignments," *Journal of Computational Biology*, vol. 3, no. 1, pp. 163–183, 1996.

[51] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[52] N. Cristianini and J. Shawe-Taylor, "An introduction to support vector machines," 2000.

[53] X. Tang, L. Zhuang, J. Cai, and C. Li, "Multi-fault classification based on support vector machine trained by chaos particle swarm optimization," *Knowledge-Based Systems*, vol. 23, no. 5, pp. 486–490, 2010.

[54] V. Vapnik, "Statistical learning theory. 1998," 1998.

[55] ——, *The nature of statistical learning theory.* springer, 2000.

[56] T. S. Furey, N. Cristianini, N. Duffy, D. W. Bednarski, M. Schummer, and D. Haussler, “Support vector machine classification and validation of cancer tissue samples using microarray expression data,” *Bioinformatics*, vol. 16, no. 10, pp. 906–914, 2000.

[57] B. SchiilkopP, C. Burgest, and V. Vapnik, “Extracting support data for a given task,” 1995.

[58] S. Tong and E. Chang, “Support vector machine active learning for image retrieval,” in *Proceedings of the ninth ACM international conference on Multimedia.* ACM, 2001, pp. 107–118.

[59] S. Tong and D. Koller, “Support vector machine active learning with applications to text classification,” *The Journal of Machine Learning Research*, vol. 2, pp. 45–66, 2002.

[60] S. Hua and Z. Sun, “A novel method of protein secondary structure prediction with high segment overlap measure: support vector machine approach,” *Journal of molecular biology*, vol. 308, no. 2, pp. 397–407, 2001.

[61] J. J. Ward, L. J. McGuffin, B. F. Buxton, and D. T. Jones, “Secondary structure prediction with support vector machines,” *Bioinformatics*, vol. 19, no. 13, pp. 1650–1655, 2003.

[62] J. Guo, H. Chen, Z. Sun, and Y. Lin, “A novel method for protein secondary structure prediction using dual-layer svm and profiles,” *PROTEINS: Structure, Function, and Bioinformatics*, vol. 54, no. 4, pp. 738–743, 2004.

[63] J. Casbon, “Protein secondary structure prediction with support vector machines,” *Master’s thesis, University of Sussex*, 2002.

[64] B. Yang, Q. Wu, Z. Ying, and H. Sui, “Predicting protein secondary structure using a mixed-modal svm method in a compound pyramid model,” *Knowledge-Based Systems*, vol. 24, no. 2, pp. 304–313, 2011.

[65] H. Kim and H. Park, “Protein secondary structure prediction based on an improved support vector machines approach,” *Protein Engineering*, vol. 16, no. 8, pp. 553–560, 2003.

[66] H.-J. Hu, Y. Pan, R. Harrison, and P. C. Tai, “Improved protein secondary structure prediction using support vector machine with a new encoding scheme and an advanced tertiary classifier,” *NanoBioscience, IEEE Transactions on*, vol. 3, no. 4, pp. 265–271, 2004.

[67] C. Chen, L. Chen, X. Zou, and P. Cai, “Prediction of protein secondary structure content by using the concept of chou’s pseudo amino acid composition and support vector machine,” *Protein and peptide letters*, vol. 16, no. 1, pp. 27–31, 2009.

[68] H.-B. Shen and K.-C. Chou, “Pseaac: a flexible web server for generating various kinds of protein pseudo amino acid composition,” *Analytical biochemistry*, vol. 373, no. 2, pp. 386–388, 2008.

[69] C. Y. Fai, R. Hassan, and M. S. Mohamad, “Protein secondary structure prediction using optimal local protein structure and support vector machine.” *International Journal of Bio-Science & Bio-Technology*, vol. 4, no. 2, 2012.

[70] P. Chatterjee, S. Basu, M. Kundu, M. Nasipuri, and D. Plewczynski, “Psp_mcsvm: brainstorming consensus prediction of protein secondary structures using two-stage multiclass support vector machines,” *Journal of molecular modeling*, vol. 17, no. 9, pp. 2191–2201, 2011.

[71] M. N. Nguyen, J. C. Rajapakse *et al.*, “Multi-class support vector machines for protein secondary structure prediction,” *Genome Informatics Series*, pp. 218–227, 2003.

[72] N. Abu-halaweh and R. W. Harrison, “Identifying essential features for the classification of real and pseudo micrornas precursors using fuzzy decision trees,” in *Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), 2010 IEEE Symposium on.* IEEE, 2010, pp. 1–7.

[73] W. Zhong, J. He, and Y. Pan, "Multiclass fuzzy clustering support vector machines for protein local structure prediction," in *Bioinformatics and Bioengineering, 2007. BIBE 2007. Proceedings of the 7th IEEE International Conference on.* IEEE, 2007, pp. 21–26.

[74] L. A. Zadeh, "Fuzzy logic, neural networks, and soft computing," *Communications of the ACM*, vol. 37, no. 3, pp. 77–84, 1994.

[75] C.-F. Lin and S.-D. Wang, "Fuzzy support vector machines," *Neural Networks, IEEE Transactions on*, vol. 13, no. 2, pp. 464–471, 2002.

[76] J.-S. Jang, "Anfis: adaptive-network-based fuzzy inference system," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 23, no. 3, pp. 665–685, 1993.

[77] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *Systems, Man and Cybernetics, IEEE Transactions on*, no. 1, pp. 116–132, 1985.

[78] N. Abu-halaweh and R. W. Harrison, "Practical fuzzy decision trees," in *Computational Intelligence and Data Mining, 2009. CIDM'09. IEEE Symposium on.* IEEE, 2009, pp. 211–216.

[79] M. Umano, H. Okamoto, I. Hatono, H. Tamura, F. Kawachi, S. Umedzu, and J. Kinoshita, "Fuzzy decision trees by fuzzy id3 algorithm and its application to diagnosis systems," in *Fuzzy Systems, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the Third IEEE Conference on.* IEEE, 1994, pp. 2113–2118.

[80] K. A. De Jong, "Analysis of the behavior of a class of genetic adaptive systems," 1975.

[81] S. W. Mahfoud, "Crowding and preselection revisited," *Urbana*, vol. 51, p. 61801, 1992.

[82] A. Fischer and C. Igel, "An introduction to restricted boltzmann machines," 01 2012, pp. 14–36.

[83] H. Larochelle, M. Mandel, R. Pascanu, and Y. Bengio, "Learning algorithms for the classification restricted boltzmann machine," *J. Mach. Learn. Res.*, vol. 13, no. null, p. 643669, Mar. 2012.

[84] G. E. Hinton, *A Practical Guide to Training Restricted Boltzmann Machines.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 599–619. [Online]. Available: https://doi.org/10.1007/978-3-642-35289-8_32

[85] N. Shervashidze, P. Schweitzer, E. J. van Leeuwen, K. Mehlhorn, and K. M. Borgwardt, "Weisfeiler-lehman graph kernels," *Journal of Machine Learning Research*, vol. 12, no. 77, pp. 2539–2561, 2011. [Online]. Available: http://jmlr.org/papers/v12/shervashidze11a.html

[86] D. Haussler, "Convolution kernels on discrete structures," 1999.

[87] R. I. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete structures," in *In Proceedings of the ICML*, 2002, pp. 315–322.