October University for Modern Sciences and Art

Faculty of Computer Science

Graduation Project

Protein Cavity Prediction

Abstract

After reading and searching in the same area of our work which introduced us to several approaches which allowed us to determine our system approach to achieve our goal to try to find a way to help in different applications like drug designing , protein and protein interaction, and disease – protein interaction to find where the ligand will bind with the protein.

Our project will be able to read PDB protein files, remove from it the dummy atoms that are not supported and extract the protein 8 properties. Added to that, The voxelization of the protein to have same shape of (16,16,16) for any protein to be able to proceed to our CNN model which will predict the binding site from the 3d structure of the protein.

Table of Contents

Table of Figures

Table of Tables

Chapter 1:
Introduction

1.1 Introduction

Each year we get introduced to many new diseases and new drugs, however do we know how the drugs work to help us heal from these diseases and how it interacts with it? So, to be able to know and understand the drug interactions, we need to be able to understand its structure and how it affects the properties in our human protein structure and to know the site of interaction for the protein so that the drug be effective [12]. According to this fact, we need to obtain the correct site for the interaction to occur and to be able to identify it. This site is known as cavity or binding site. In addition, the evaluation and identification of the cavity plays a main role in understanding the interactions of the protein, which is highly used for drug designing. We also need to know that the molecule or atom which binds with the protein is also known as Ligand [8].

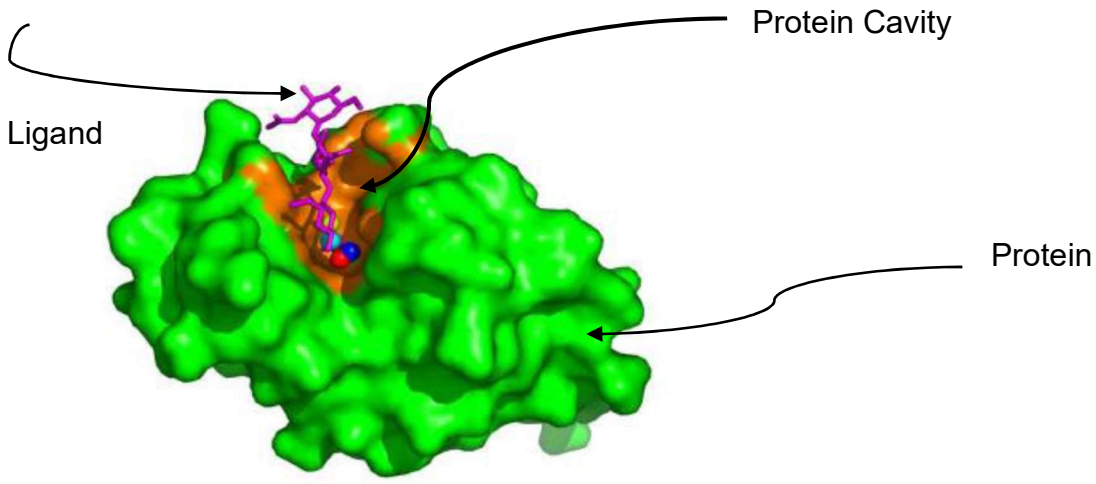


Figure 1: The Ligand binding with the protein through its cavity [4].

Nevertheless, each protein may have multiple cavities, and to be able to identify the correct cavity is not easy as each protein has its own structure and protein properties. Therefore, the identification and evaluation of the protein binding-cavities surface are the initial steps to understand protein structure and help in characterizing the binding site shape and know it's surrounding which will play a

role for variety of applications to predict the cavity [11]. According to that, our light will be spotted onto the proteins, ligand, and we will exclude the other cells' receptors from our consideration. Lately, the understanding of the protein cavity helped in developing drugs and knowing how diseases interact with human-protein structure.

1.2 Problem statement

Protein cavity prediction is an identified problem in bioinformatics and it is important as it allow us to understand the interactions of the protein and what ligands can bind with it, which will allow us to know more about diseases, drug designing, and understanding human-protein structure. Moreover, many researchers tried to identify and understand the structure of the cavity and there were some who have achieved the correct binding site for the protein by using one of the algorithms that were developed over the years which are geometric, artificial intelligence, molecules energy, or statistical [6].

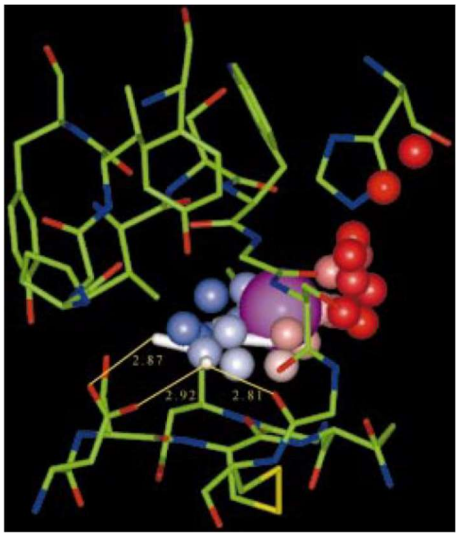


Figure 2: Finding the cavity using energy of the atoms [1].

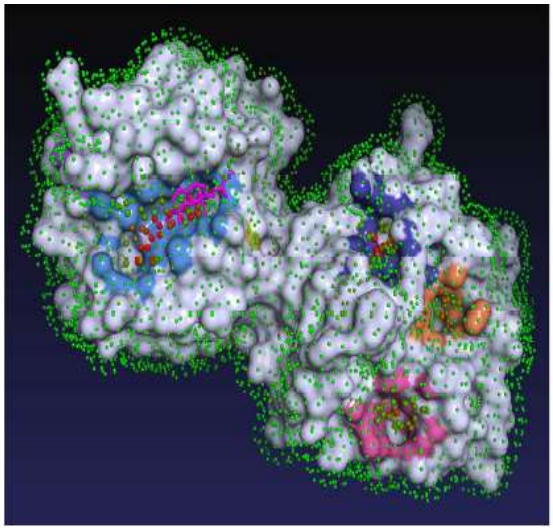


Figure 3: Finding cavity using Artificial intelligence [7].

1.3 Objective

This project will allow us to predict the correct and most accurate cavity of the protein that will make interaction with other ligand, which may help in multiple

applications like drug designing or diseases' actions with the human protein. Therefore, the key success of the system will be identifying the correct cavity of the protein [12].

1.4 Motivation

The understanding of the protein cavity and identifying protein cavity will be huge interest as it will allow the understanding of the diseases how it interacts with the human body and how each drug may affect it. In addition to, it will be useful and helpful in rational drug designing, knowing side-effects that could happen, and fragment-based drug discovery [7]. Added to that, my own motivation to choose this topic is due to of my interest in biology and how our body interact with the taken drugs, and to be able to use my computational skills in an interest is a motivation for myself.

1.5 Thesis layout

At the end, the first chapter will provide a main idea about the problem and the aim of this project. Secondly, the second chapter will provide a literature review and the background of the previous work made by other researchers in the same area. Thirdly, the third chapter will provide the methods and materials that will be used in this project. Added to that the fourth chapter will demonstrate how the system was developed. Moreover, chapter five will show our results and evaluation of our system. At last we will conclude our work in chapter six.

Chapter 2: Background and Literature Review

2.1 Background

As mentioned before, there are four different techniques to be used to predict the protein cavity. One of these techniques is geometric approach which may be implemented by multiple algorithms. The first algorithm in the geometric approach is called Roll. Roll is used to detect protein cavities by rolling a probe sphere on the protein surrounding without allowing the overlap with the protein cell wall, as well as the rolling direction will be controlled based on the inner border tracing as will be shown in figure 4 [14].

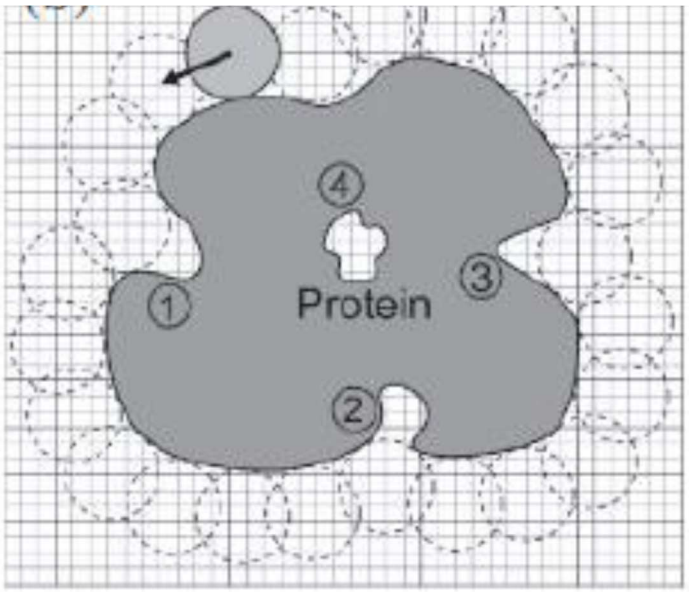


Figure 4: Roll Algorithm [14].

In addition to that, there is another algorithm in geometric approach is called PASS (Putative Active Sites with Spheres). It is designed to fill the protein structure cavities with a set of spheres and to identify a few of these spheres as binding site points which are the most likely points to be center of the binding using the Connolly sphere geometry as shown below in figure 5 [1].

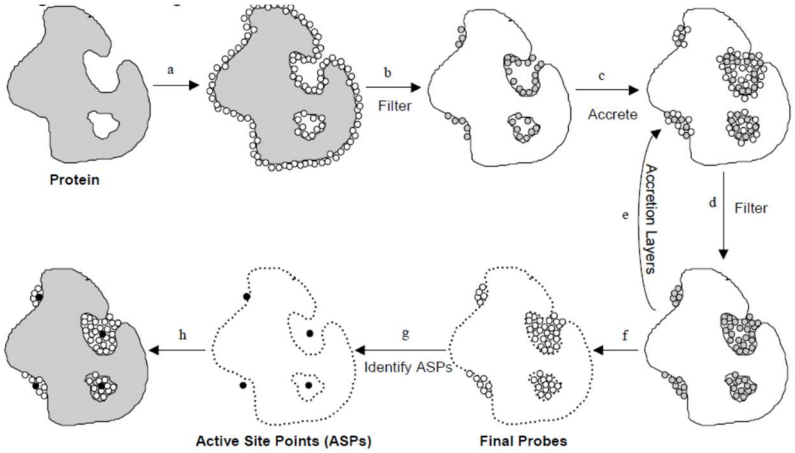


Figure 5: PASS Algorithm [1].

Added to that, in the artificial intelligence approach there are multiple algorithms that can be implemented. One of these algorithms is the random forests, which it works by putting layer of points on the protein solvent accessible surface, design atomic feature vectors for the exposed protein atoms and then classify and cluster the points [7].

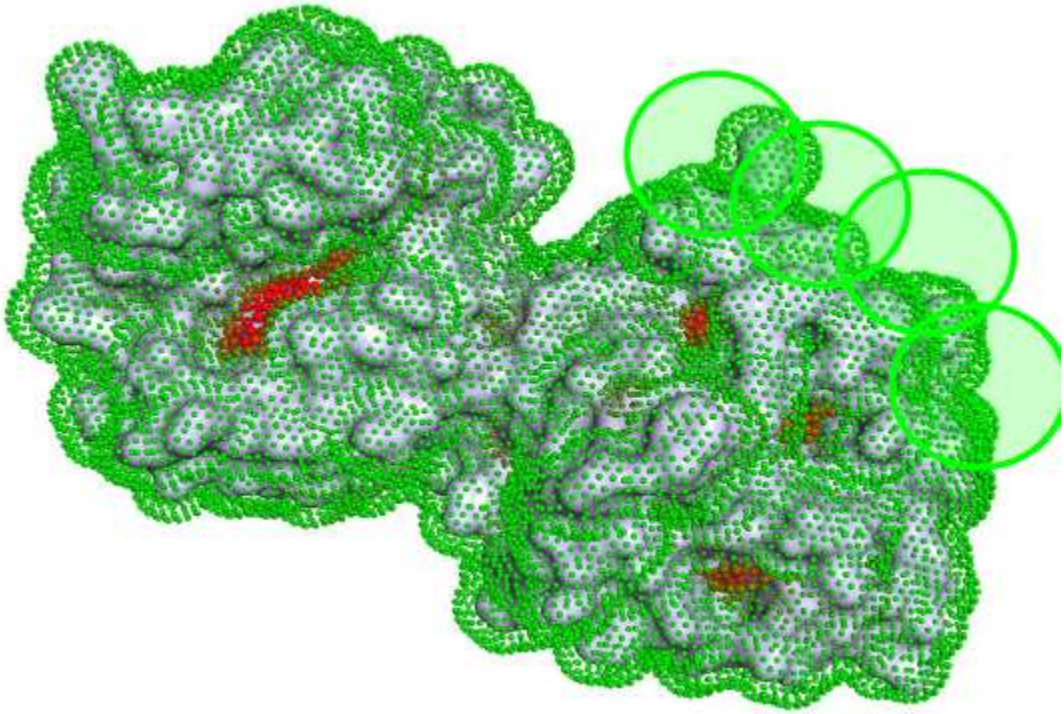


Figure 6: Random forests Algorithm [7].

Lastly, in the statistical approach, there is algorithm called 3D probability distributions, which is used to identify the atoms which will be able to bind on the protein surface according to the predicted confidence level for each protein surface atoms as will be shown in figure 7 in which **A.** shows the protein atoms types, **B.** Shows the confidence level. Therefore, this method will be also useful in knowing the protein structures with the ligand binding which will be useful in the drug design applications [5].

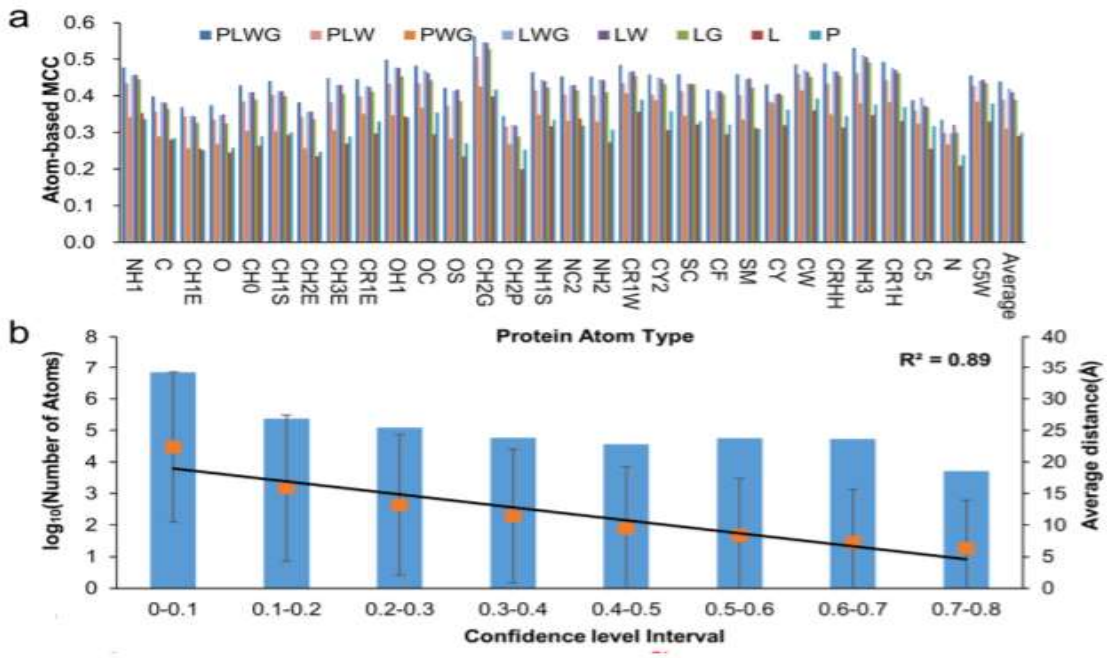


Figure 7: 3D probability Algorithm [5].

2.1.1 Approach 1: Machine Learning

Machine Learning (ML) is seen as a part of artificial intelligence (A.I), which is the study of computer algorithms to improve the experience automatically and to adapt itself over the time as its precept more data from its surrounding and its interactions.

2.1.1.1 Algorithm: Random Forests

Random forests is one of algorithms that lays under the umbrella of Machine Learning, and it used frequently with classifications and regression

tasks. Added to that, it is known that it is flexible and easy to use as it produces without tuning the parameter and usually obtain good results. Accordingly, it is one of the most used algorithms in machine learning as it is powerful, fast, and can deal with highly correlated variables therefore, it is used fully in bioinformatics due to the data used as protein and atoms as it will work as a tree method which will distribute the atoms to sub samples until we retrieve the final predication [7].

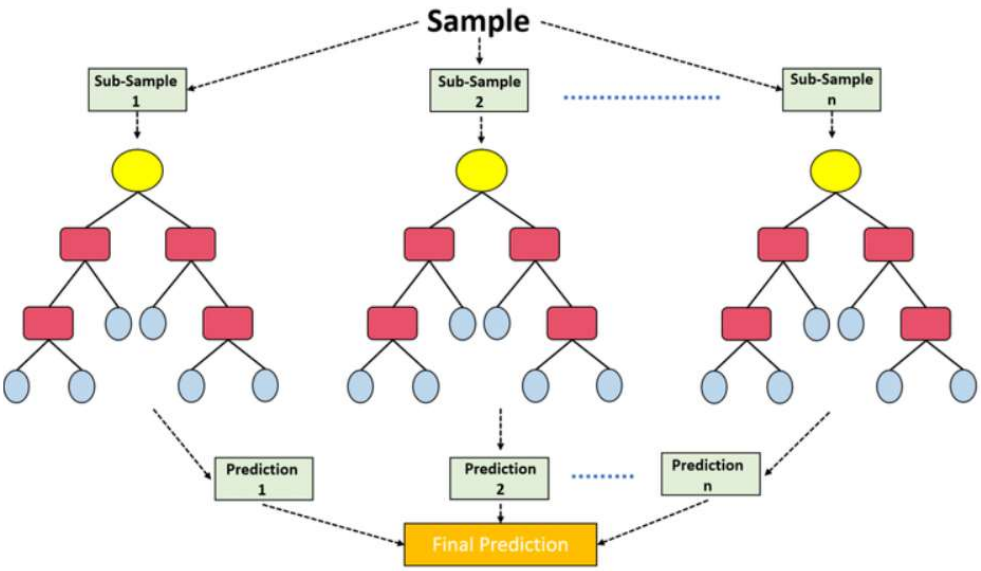


Figure 8: Random Forests Example [9].

2.1.2 Approach 2: Deep Learning

The deep learning approach is also seen as a part of the artificial intelligence (A.I), and it makes the same function of human brain, which is processing data and creating map to use for its decision making. Furthermore, in deep learning it is able to learn without supervision and can work on unstructured and unlabeled data [6].

2.1.2.1 Algorithm: Convolution neural networks

Convolution neural networks (CNN) is widely applied in image recognition, computer vision, bioinformatics, and other artificial intelligence

research fields. Added to that images it models to train and test each input image that will pass through chain of convolution layers with kernels (known also as filters), pooling, fully linked layers, and apply SoftMax function to classify an object and assign it with probabilistic values between 0 and 1 [3]. However, in bioinformatics our input is a protein most of time used as PDB extension instead of an image with details as atoms and it coordinates in the protein.

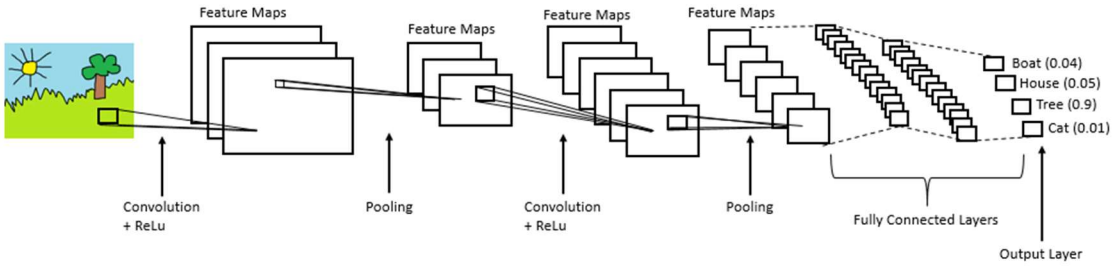


Figure 9: CNN Example [3].

2.1.3 Approach 2: Geometric Algebra

Almost all interactions of objects in virtual 3D world is based on calculations performed using linear algebra, however linear algebra is not enough for high level language for geometric programming. Therefore, Geometric Algebra represents an alternative for linear algebra limitations in the high-level programming and it is known to be compact, time effective, and performance enhanced way to represent the geometry of 3D objects on computers. Accordingly, it is used to study the surface of the protein to be able to predict its active binding sites [11].

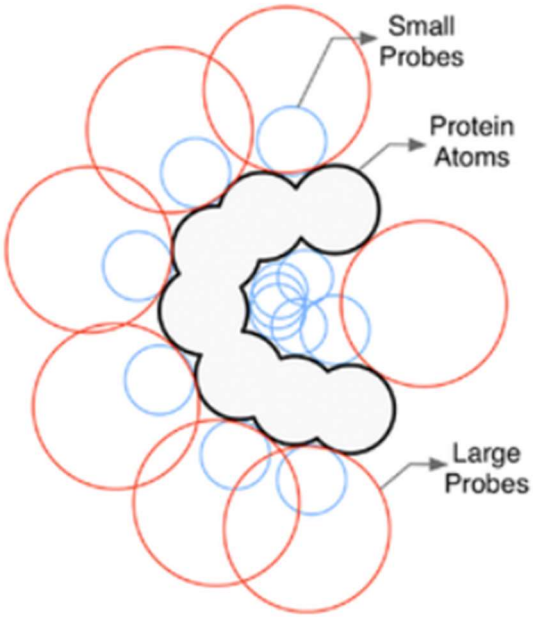


Figure 10: Geometric Example of using geometry shapes in predicting the cavity [14].

2.2 Previous Work

In this part, we will be introducing couple of researches that were made by other researchers that are interested in this area and where able to achieve the goal to predict the cavity of the protein.

2.2.1 Research 1: P2Rank: machine learning based tool for rapid and accurate prediction of ligand binding sites from protein structure

2.2.1.1 Strategy & Structure

P2rank is based on classification of points which is evenly distributed on protein's solvent accessible surface (SAS). Initially, we start by putting layer of points on the (SAS) surface then, we use feature vectors for the exposed atoms of the protein that was inherited from the table of features of the amino acids atoms and aggregate the feature vectors of the Connolly points with respect to the distance based on weight function. Secondly, we classify the points as it is either ligandable

or un-ligandable with score of 0 or 1 in which 0 will be shown as green points while 1 will be shown as red points on the surface. Lastly, they start to cluster the Connolly points which are ligandable which will form the prediction of the possible cavity (Binding Site), and then they are ranked according to the cumulative ligandability scoring [7].

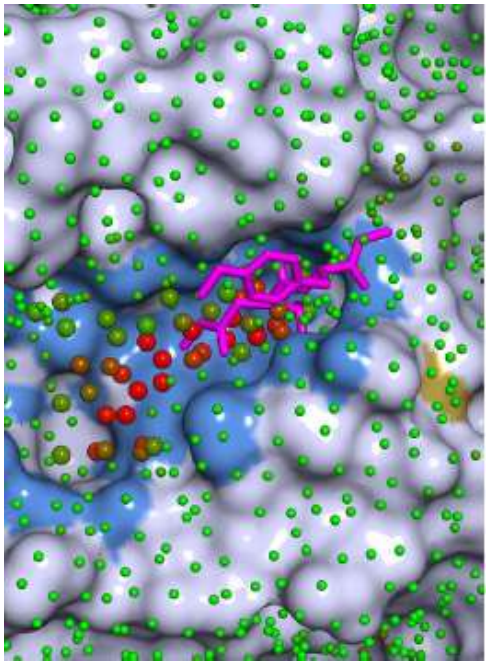


Figure 11: Putting points on protein solvent accessible surface [7].

Table 1: Atom Features [7].

Element	Description
C	*Non H-bonding aliphatic carbon*
A	*Non H-bonding aromatic carbon*
NA	*Acceptor 1 H-bond nitrogen*
NS	*Acceptor S Spherical nitrogen*
OA	*Acceptor 2 H-bonds oxygen*
OS	*Acceptor S Spherical oxygen*
SA	*Acceptor 2 H-bonds sulfur*

HD	*Donor 1 H-bond hydrogen*
HS	*Donor S Spherical hydrogen*
MG	*Non H-bonding magnesium*
ZN	*Non H-bonding zinc*
MN	*Non H-bonding manganese*
CA	*Non H-bonding calcium*
FE	*Non H-bonding iron*

2.2.1.2 Data

P2rank used multiple datasets to be able to train and evaluate their work. They used CHEN11, JOINED, COACH420, and HOLO4K [7].

CHEN11: is a dataset which consists of 251 proteins consists of 476 ligands and it was constructed from LBS Researchers and it was used here as a source for efficiency of the data [2] and it was used for the training of the model [7].

JOINED: it consists of proteins from several smaller datasets that were used in previous studies and it was used to optimize the parameters of the algorithm [7].

COACH420: consists of 420 single chain protein used in another research test set [13] and removed proteins that were available in the other used datasets [7].

HOLO4K: Is a huge dataset of protein-ligand containing multi-chain structures which it can be directed collected from PDB and it was used for the unbiased results [7].

2.2.1.3 Method Evaluation

The P2rank is a command line which is lightweight in the sense it doesn't depend in any other tools which will make it fast and powerful tool to be used. Added to that, it doesn't require any variable scaling while it deals with highly correlated variables [7].

2.2.1.4 Results Evaluation

P2rank is one of the algorithms that have best ratio between accuracy and speed; the accuracy of performance on COACH420 database is 78% and on HOLO4K is 74%, and it is light weighted and fast to get the test result [7].

2.2.2 Research 2: Fast Prediction and Visualization of Protein Binding Pockets with PASS

2.2.2.1 Strategy & Structure

PASS (Putative Active Sites with Spheres) is a simple tool that uses geometry to characterize regions that has engulfed volume in proteins surface and to let us know the positions that are most likely to be the active binding site. Firstly, the algorithm takes protein as an input and it checks the hydrogen percentage to decide to remove them or add them to the parameters, and then it puts first layers of spheres around the surface of the protein. Secondly, they perform multiple filters to check the radius, and volume in respect to the protein atoms, Thirdly, spheres are checked if there is any sphere overlap with each other and if there is an overlap, the spheres loops again, and they are re-distributed until the spheres are correctly drawn around the protein surface [1].

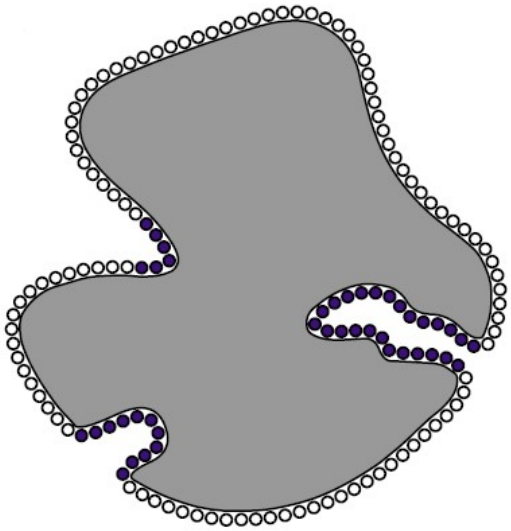


Figure 12: Spheres distributed around the protein surface [1].

At last, the low solvent exposure spheres in that area are kept as they are potential to be the cavity of the protein and then trial are made to get the centered sphere of the spheres to be the center of the ligand that will bind with the protein which will be called as Active Site Points (ASPs) [1].

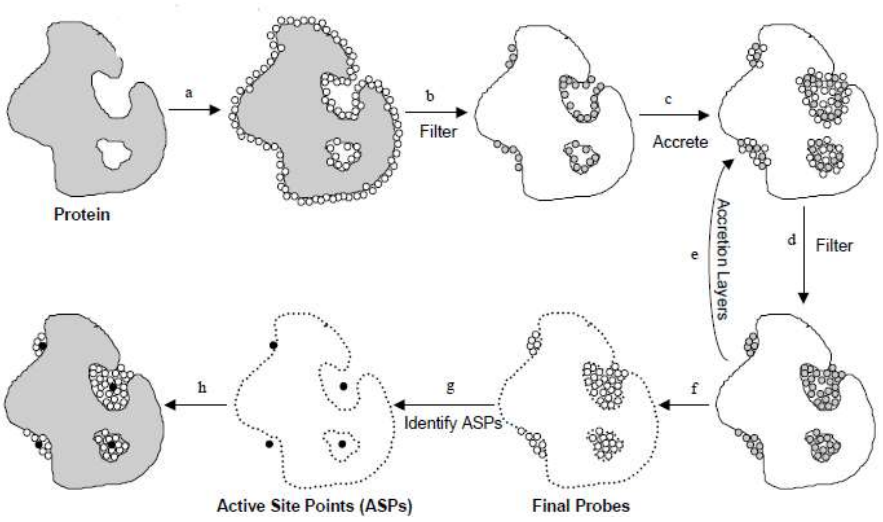


Figure 13: PASS Algorithm [1].

2.2.2.2 Data

This research uses PDB (protein data bank) proteins as a dataset to train and test on it, which is commonly used for most of the projects as it is efficient and accurate and can easily be found and collected from PDB website with all the data needed like atoms type and their coordinates [1].

2.2.2.3 Method Evaluation

In this method, the usage of geometry from the known atomic coordinates

reduces the irregularity of the geometric surface which will make better performance by using grids/voxels. On the other hand, the usage of a grid makes the system storage consumes memory unnecessarily which makes its CPU time not stable and always takes time to predict the binding site [1].

2.2.2.4 Results Evaluation

Pass can correctly predict the binding site in 17 trails from 21 trail which is approximately 80% success rate which is good with similarity to

other methods and is acceptable for the success of cavity detection algorithm [1].

2.2.3 Research 3: Roll a new algorithm for the detection of protein pockets and cavities with a rolling probe sphere

2.2.3.1 Strategy & Structure

In this research the main concept of the algorithm is to generate a cover surface that is called probe surface which will be on the surrounding of the protein to identify the region between the protein surface and probe surface as to be a possible cavity area. Firstly, Sphere probe is implemented on the protein grid and as it encounters first protein point, the probe starts to roll along the protein surface without overlapping with the protein until it makes full cycle and returns to the starting points. Moreover, Roll can determine the pockets with respect to the volume and shape. However, the probe radius must be adjusted to have the correct cover around the protein surface to ensure it find the cavities. Secondly, the ranking of the predicted binding sites was depending on the volume depth of the binding site [14].

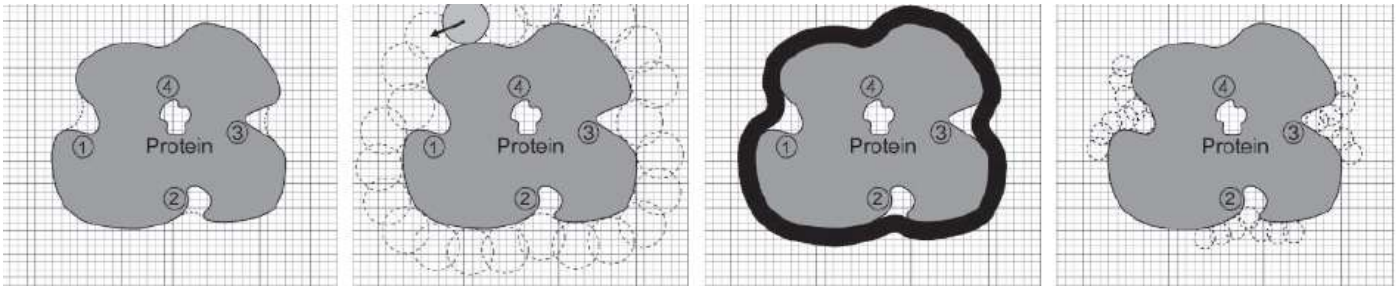


Figure 14: Roll Algorithm [14]

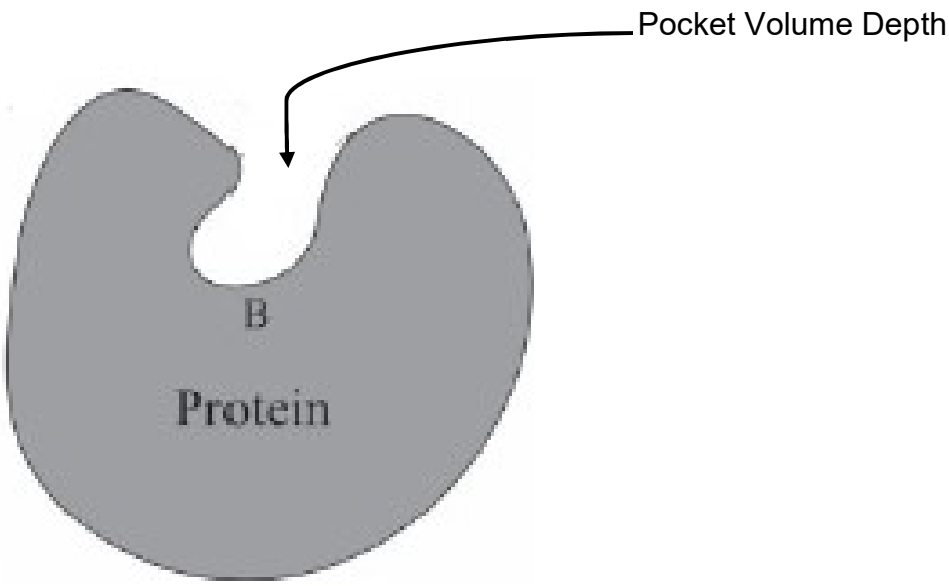


Figure 15: Protein Pocket volume Depth [14].

2.2.3.2 Data

The data was downloaded from Protein Data Bank (PDB) containing set of 48 bound or unbound structures and it was used for training and testing to be compared with other methods, and this dataset is most efficient as it has protein and its cavity place for training to have the best outcome [14].

2.2.3.3 Method Evaluation

Roll algorithm had the advantage to have successful predication of the cavity in difficult cases as some proteins have very shallow binding sites which is difficult to predict while using sphere algorithm by other methods. However, the usage of the grid system is relying on the adjustment of the protein in the 3D grid, as the roll is performed on 3D grid cut into 2D parts then it starts to roll on them, the inner tracing border won't take affect by the orientation which will affect the success rate [14].

2.2.3.4 Results Evaluation

Roll algorithm in this research was successful in predicting cavity and identifying protein pockets and was implemented in tool called POCASA, which has achieved success rate 75% to 77% for the 48 bound/unbound structures [14].

Chapter 3: Material and Methods

3.1 Materials

In this section, we will describe the data that will be used in the project, the tools, programming language, and the environment of the computer that will run the project.

3.1.1 Data

The data ScPDB can be downloaded from PDB website which is available as CSV files that is going to be used for this project. Furthermore, data consist of total of 9190 structures, and 12 structure are ignored as they contain multiple wrong entries. The ScPDB dataset also contains clustering of the binding sites for each available structure [6].

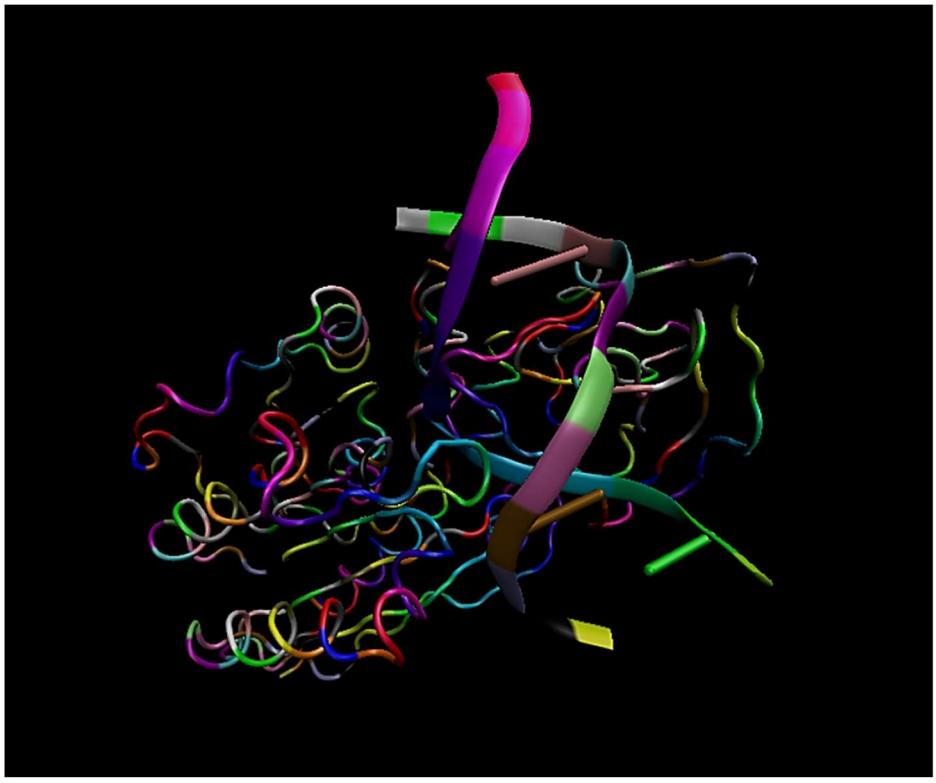


Figure 16: Protein example from the dataset visualized by VMD

1	C	20.8860	30.6720	1.9640	C.2	1 ILE4	0.0000
2	O	21.4020	30.5640	3.0740	O.2	1 ILE4	0.0000
3	CA	21.6880	30.3590	0.7110	C.3	1 ILE4	0.0000
4	N	22.9840	31.0900	0.7920	N.pl3	1 ILE4	0.0000
5	CB	21.8960	28.8220	0.6070	C.3	1 ILE4	0.0000
6	CG1	22.4760	28.4460	-0.7600	C.3	1 ILE4	0.0000
7	CD1	23.9400	28.7860	-0.9410	C.3	1 ILE4	0.0000
8	CG2	20.5850	28.0870	0.8170	C.3	1 ILE4	0.0000
9	HA	21.1816	30.6766	-0.0907	H	1 ILE4	0.0000
10	H	23.7789	30.6533	1.2133	H	1 ILE4	0.0000
11	HB	22.5399	28.5373	1.3172	H	1 ILE4	0.0000
12	HG12	21.9547	28.9294	-1.4632	H	1 ILE4	0.0000
13	HG13	22.3677	27.4596	-0.8837	H	1 ILE4	0.0000
14	HD11	24.2360	28.5088	-1.8551	H	1 ILE4	0.0000
15	HD12	24.0690	29.7718	-0.8335	H	1 ILE4	0.0000
16	HD13	24.4820	28.3020	-0.2540	H	1 ILE4	0.0000
17	HG21	20.7402	27.1017	0.7463	H	1 ILE4	0.0000
18	HG22	20.2232	28.3032	1.7238	H	1 ILE4	0.0000
19	HG23	19.9277	28.3718	0.1193	H	1 ILE4	0.0000
20	C	17.7890	30.2730	3.2730	C.2	2 ARG5	0.0000
21	O	17.2150	29.6100	2.3990	O.2	2 ARG5	0.0000
22	CA	18.7590	31.3990	2.9010	C.3	2 ARG5	0.0000
23	N	19.6280	31.0600	1.7800	N.am	2 ARG5	0.0000
24	CB	17.9900	32.6790	2.5910	C.3	2 ARG5	0.0000
25	CG	17.1590	33.2100	3.7410	C.3	2 ARG5	0.0000
26	CD	16.8540	34.6910	3.5730	C.3	2 ARG5	0.0000
27	NE	18.0130	35.5290	3.8770	N.pl3	2 ARG5	1.0000
28	CZ	18.9230	35.9230	2.9900	C.cat	2 ARG5	0.0000
29	NH1	18.8310	35.5560	1.7190	N.pl3	2 ARG5	0.0000
30	NH2	19.9270	36.6980	3.3760	N.pl3	2 ARG5	0.0000
31	HA	19.3401	31.5765	3.6952	H	2 ARG5	0.0000
32	H	19.2683	31.1210	0.8489	H	2 ARG5	0.0000
33	HB2	17.3775	32.4963	1.8219	H	2 ARG5	0.0000
34	HB3	18.6499	33.3841	2.3314	H	2 ARG5	0.0000
35	HG2	17.6630	33.0770	4.5944	H	2 ARG5	0.0000
36	HG3	16.2977	32.7035	3.7802	H	2 ARG5	0.0000
37	HD2	16.1066	34.9383	4.1896	H	2 ARG5	0.0000
38	HD3	16.5748	34.8575	2.6273	H	2 ARG5	0.0000
39	HE	18.1306	35.8288	4.8237	H	2 ARG5	0.0000
40	HH11	19.5203	35.8576	1.0603	H	2 ARG5	0.0000
41	HH12	18.0723	34.9776	1.4192	H	2 ARG5	0.0000
42	HH21	20.6126	36.9959	2.7118	H	2 ARG5	0.0000

Figure 17: Dataset CSV file example for one of the proteins with its coordinates and atoms.

3.1.2 Tools

Unity: It gives the user the ability to create experiences in both 2d space and 3d space and can implement object on these spaces, and the engine offers a primary scripting API in C#.

C#: Is a programming language that can be used to perform wide range of tasks and objectives in variety of professions.

Anaconda: Is a platform for applying Machine learning and Deep learning models.

Spyder: It is the scientific python development environment (IDE) that is included with anaconda.

Python: open source programming language which helps developers to work on their project effectively.

VMD: Visual Molecular Dynamics (VMD) is molecular modelling which is used to visualize protein and to analyze it.

3.1.3 Environment

Local CPU, intel i7 processor with 4 cores, 8 Logical processors and 16GB Ram.

Local GPU Nvidia GeForce GTX 1060.

3.2 Methods

In this part we will explain the project methodology, and the used approaches and algorithms that will be implemented.

3.2.1 System architecture Overview

Firstly, we get the protein from the PDB dataset and start to voxelize the protein to extract the protein 8 features occupancy to be our input for our CNN network, and then we will use the CNN model and train it to obtain the prediction of the correct cavity.

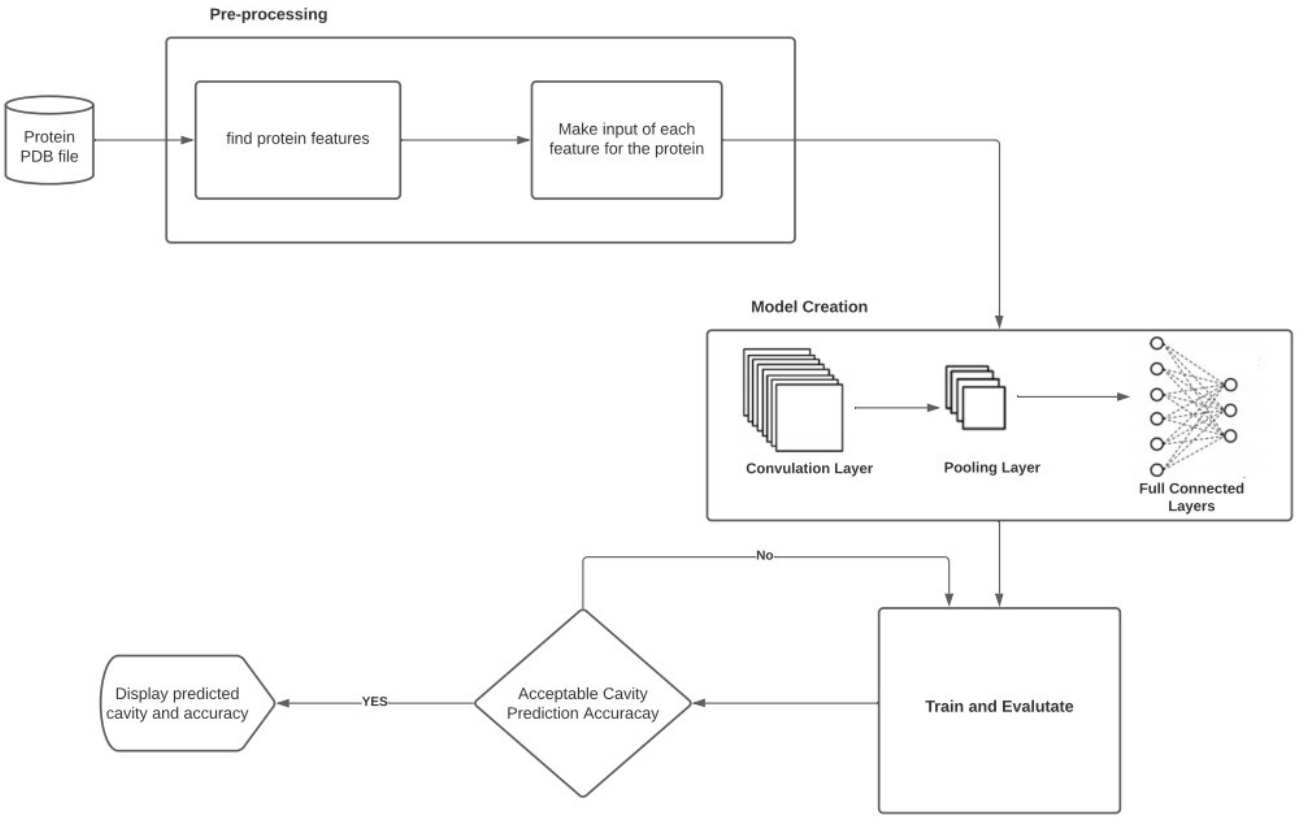


Figure 18: CNN Architecture overview

Chapter 4: System Implementation

4.1 System Development

The system is being built by multiple phases to reach the goal of the system, the initial phase of the system was to understand the dataset which contains different number of atoms and features. Moreover, we used unity to be able to visualize the data, this which have allowed us to understand the 3d space of the protein atom coordinates and the connection that occurs between the atoms and how the protein structure looks like, which also allowed us to understand how the binding site may occur according to protein solvent protein [PSP] “technique” and helped to validate that the protein is read correctly by the system. Secondly, I used spyder to read the data and obtain the atom coordinates using HTMD tool which helped in importing the data successfully and was implemented using a sample small protein and started to perform the voxelization on the whole protein and put each atom in its voxel space and index which will help to procced and gain the 8 features that will be the most important.

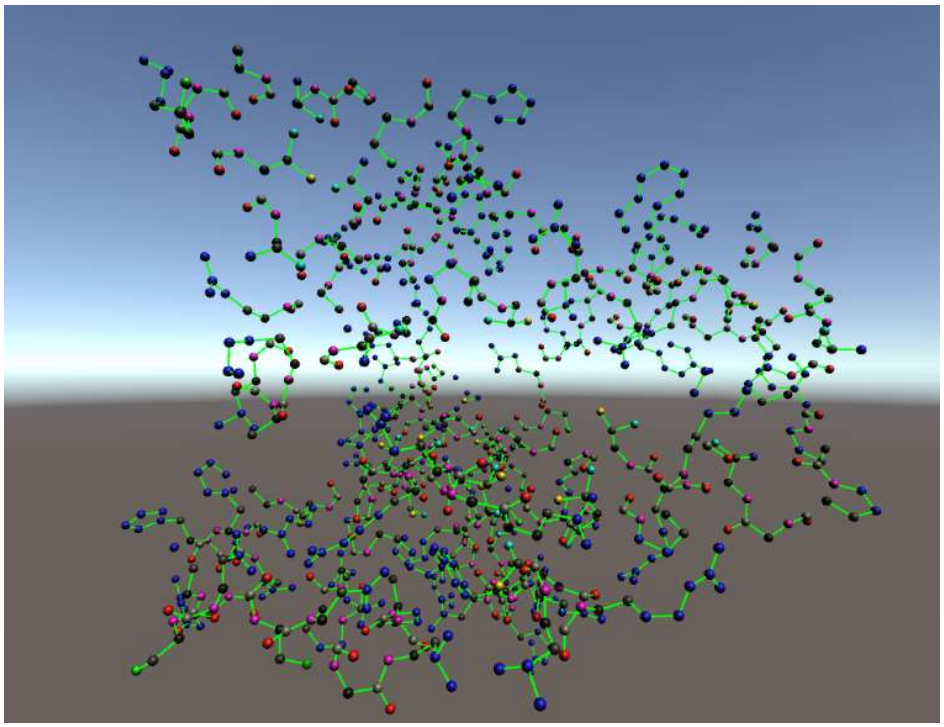


Figure 19:Protein visualization by Unity

Table 2: Atoms coloring scheme created

Atoms	Color
CA	Black
C	Gray
O	Red
N	Magenta
OG	Green
CG1	Yellow
CG2	Cyan
Water	Blue

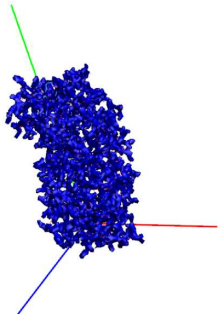


Figure 20:Visual representation of Hydrophobic [6].

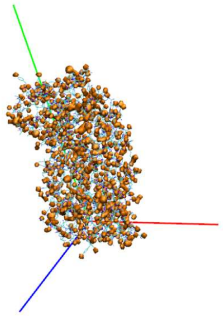


Figure 21:Visual representation of Hydrogen Acceptor [6].

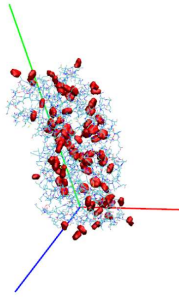


Figure 22:Visual representation of Aromatic [6].

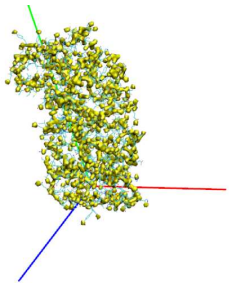


Figure 23:Visual representation of Hydrogen Donor [6]

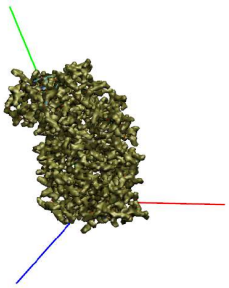


Figure 24:Visual representation of Positive ionizable [6].

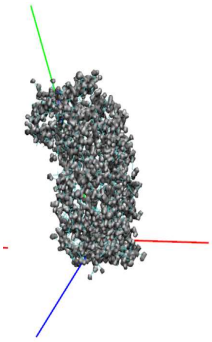


Figure 25:Visual representation of Negative ionizable [6].

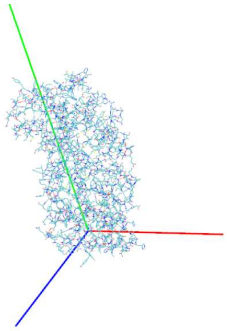


Figure 26:Visual representation of Metal [6].

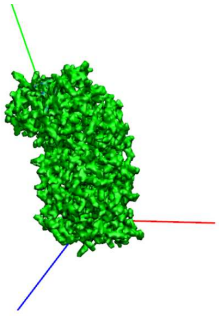


Figure 27:Visual representation of Volume [6].

Lastly, we will start to insert these features to simple CNN model using keras to test if it reads the data correctly and recognize the protein features and number of the voxels which we made the size to be (16x16x16) on the entire dataset to have the same size shape which accordingly we can start to train on the dataset and to check that the layers are able to read the 8 features for each voxel and successfully output all the voxels either they are binding sites or not and we visualize it using VMD.

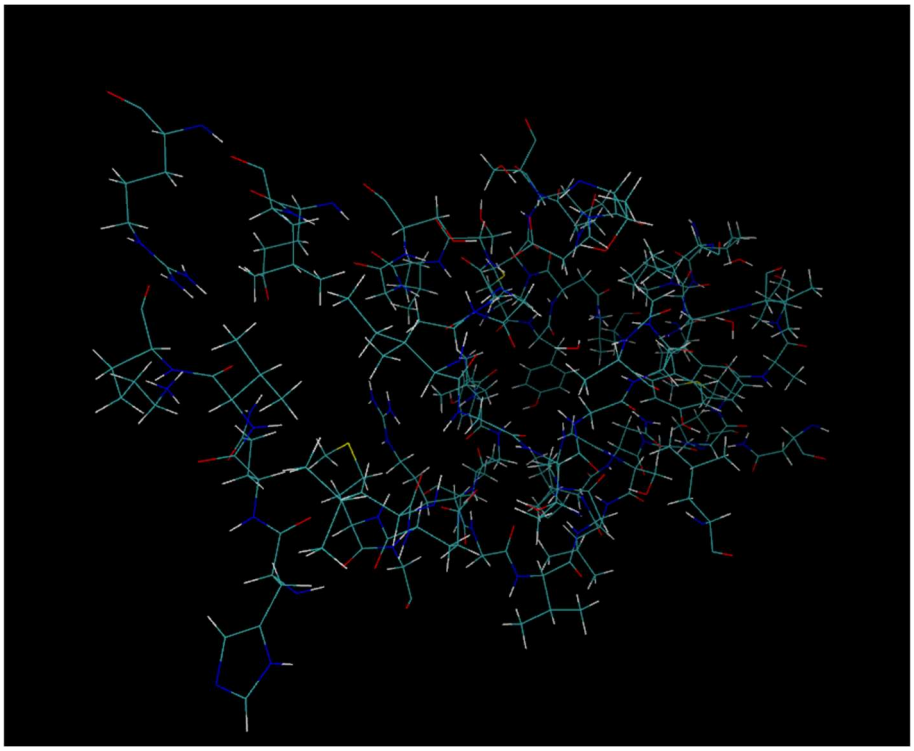


Figure 28: Visual Representation of Cavity using VMD.

4.2 System Structure

In this section, it will be responsible to show an overview of the final system we developed and how the data flows through the system and how each stage is responsible of what action to do. Moreover, in the subsections of this parts, we will try to demonstrate system using class diagram for the used classes throughout the project which will help us in showing a clear overview of the system.

4.2.1 System Overview

The developed system consists of 3 main stages. Firstly, it is responsible for reading the data, prepare it and pre-process for the next stages. Secondly, we voxelize and extract feature from the data. Ending by entry of the data to the CNN model.

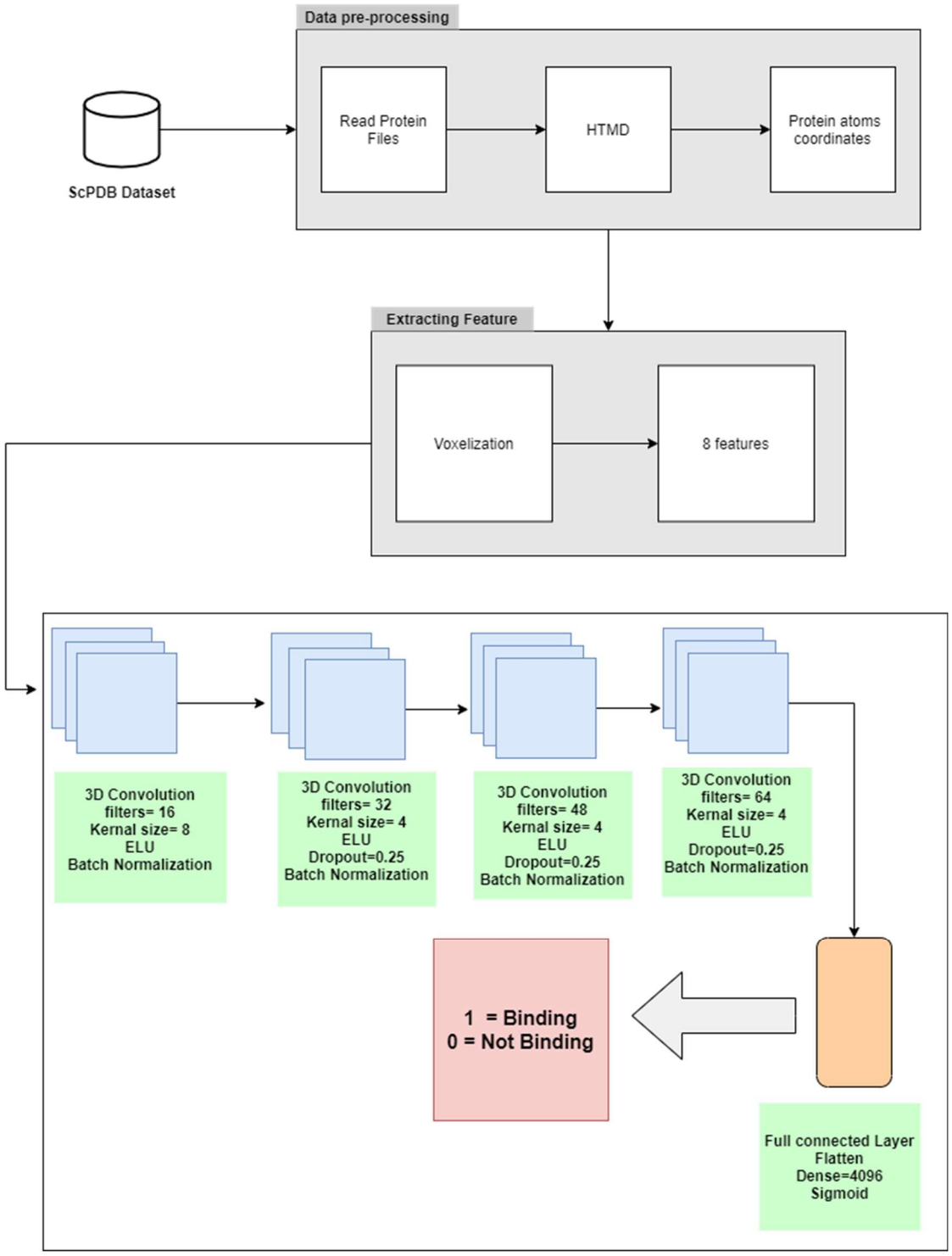
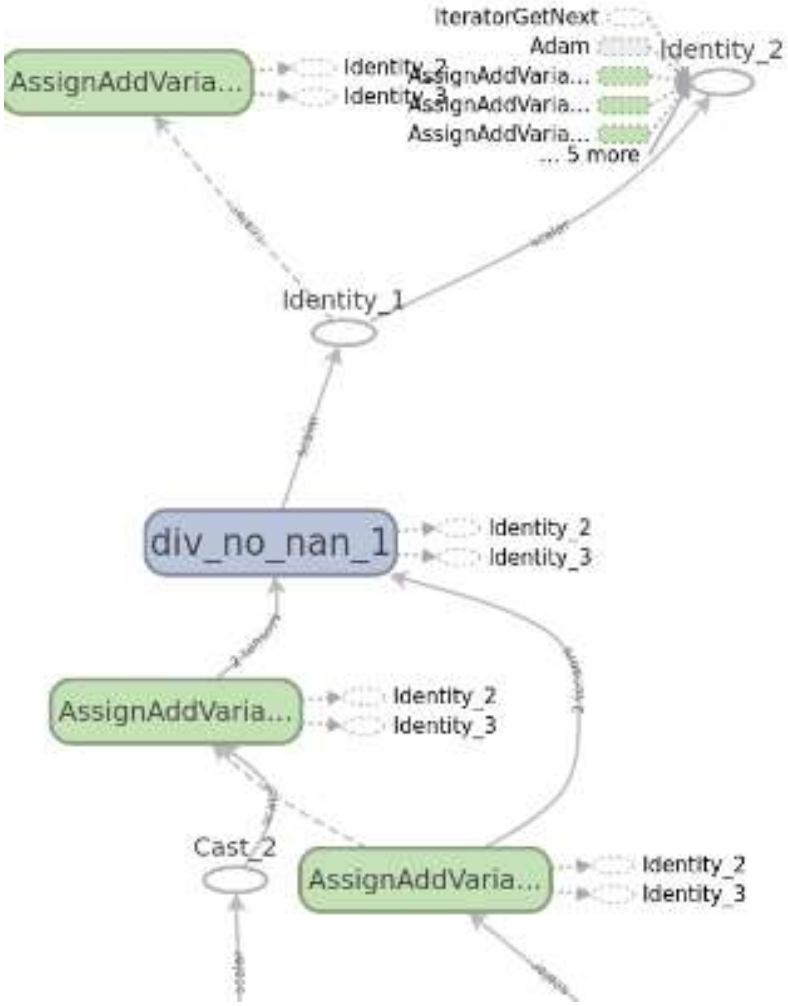


Figure 29:System Architecture.

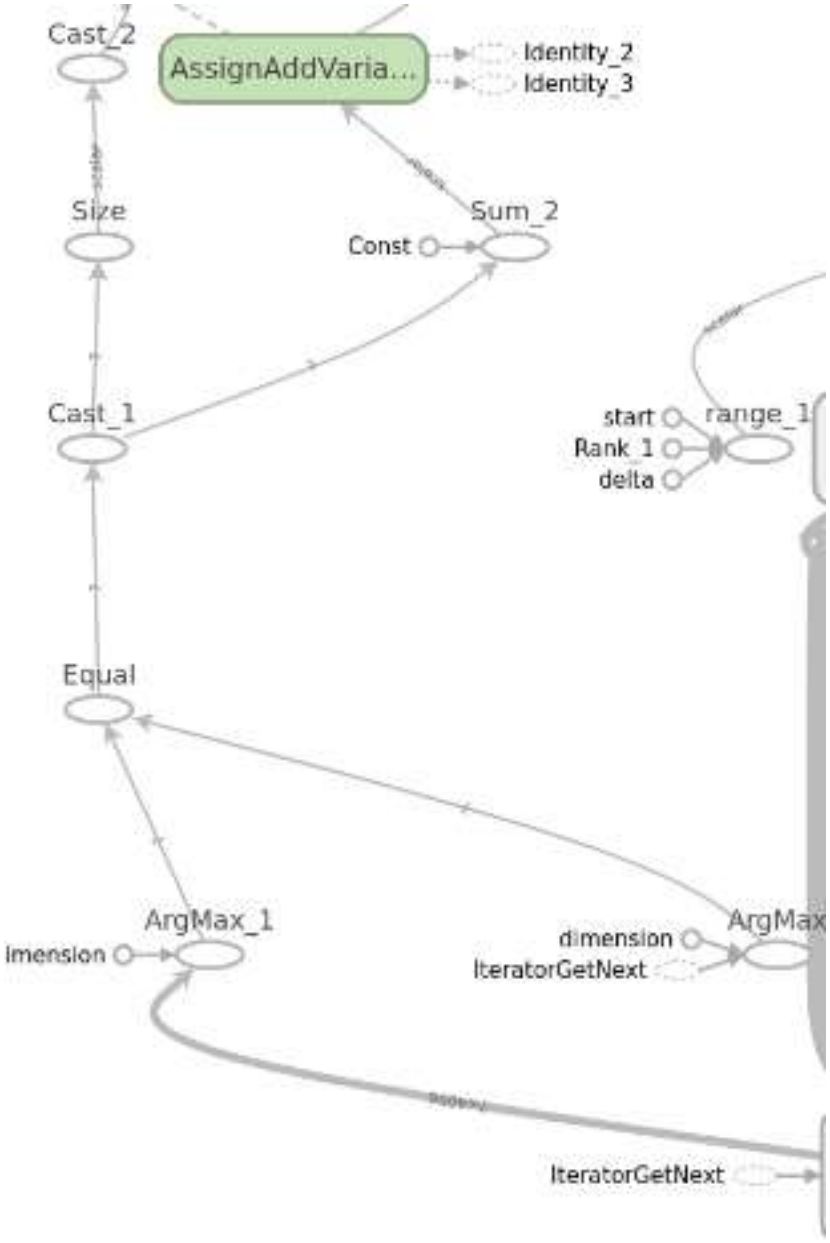
Lastly, after the model outputs the results, we will see each binding voxel atoms and try to visualize it using VMD to validate the model if it is learning correctly from the input data.

4.2.2.1 Tensor Board

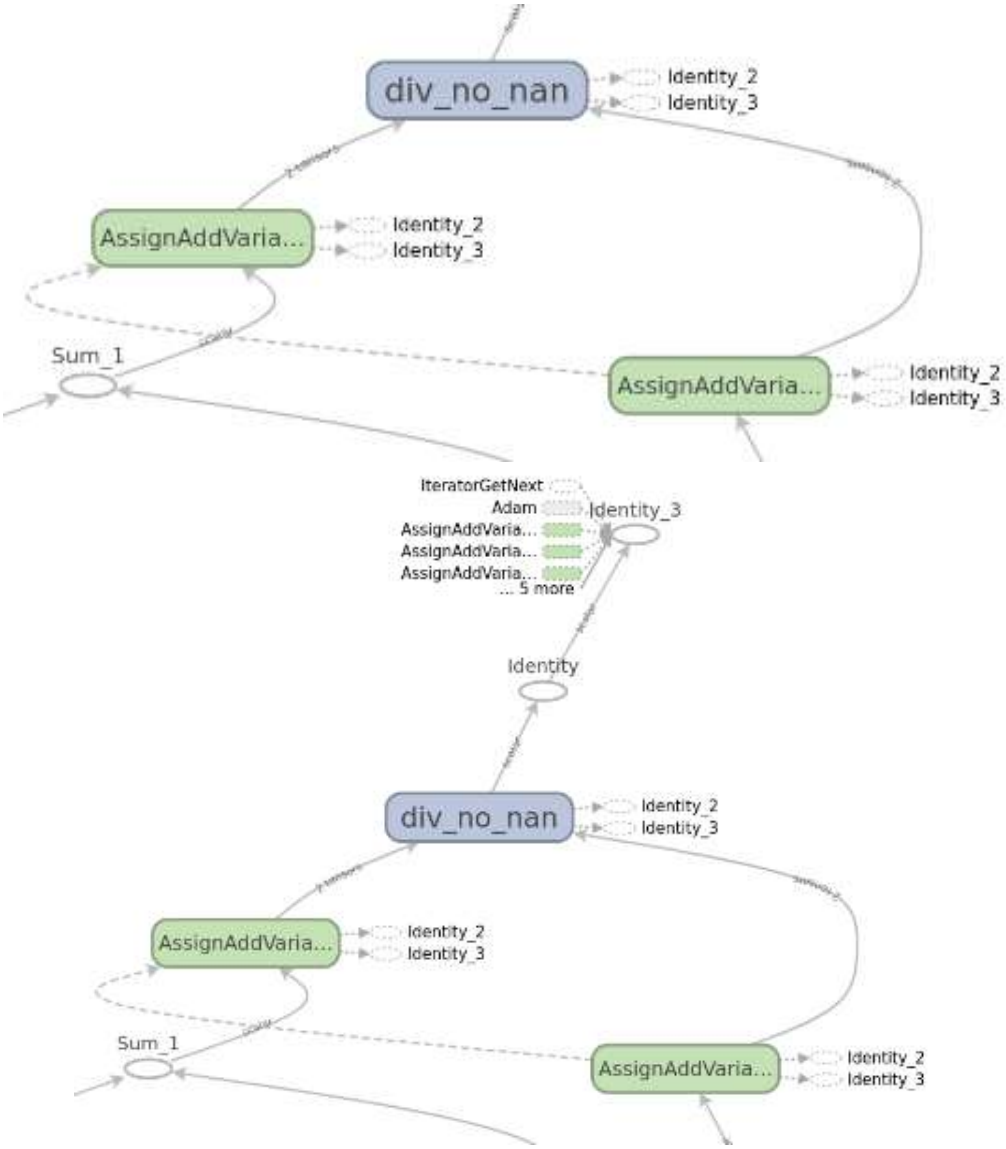
In this section we will demonstrate the whole structure of the CNN using the tensor board, which it will describe the convolution layers, the drop out layers, the flatten layers with dense, and the activation functions.



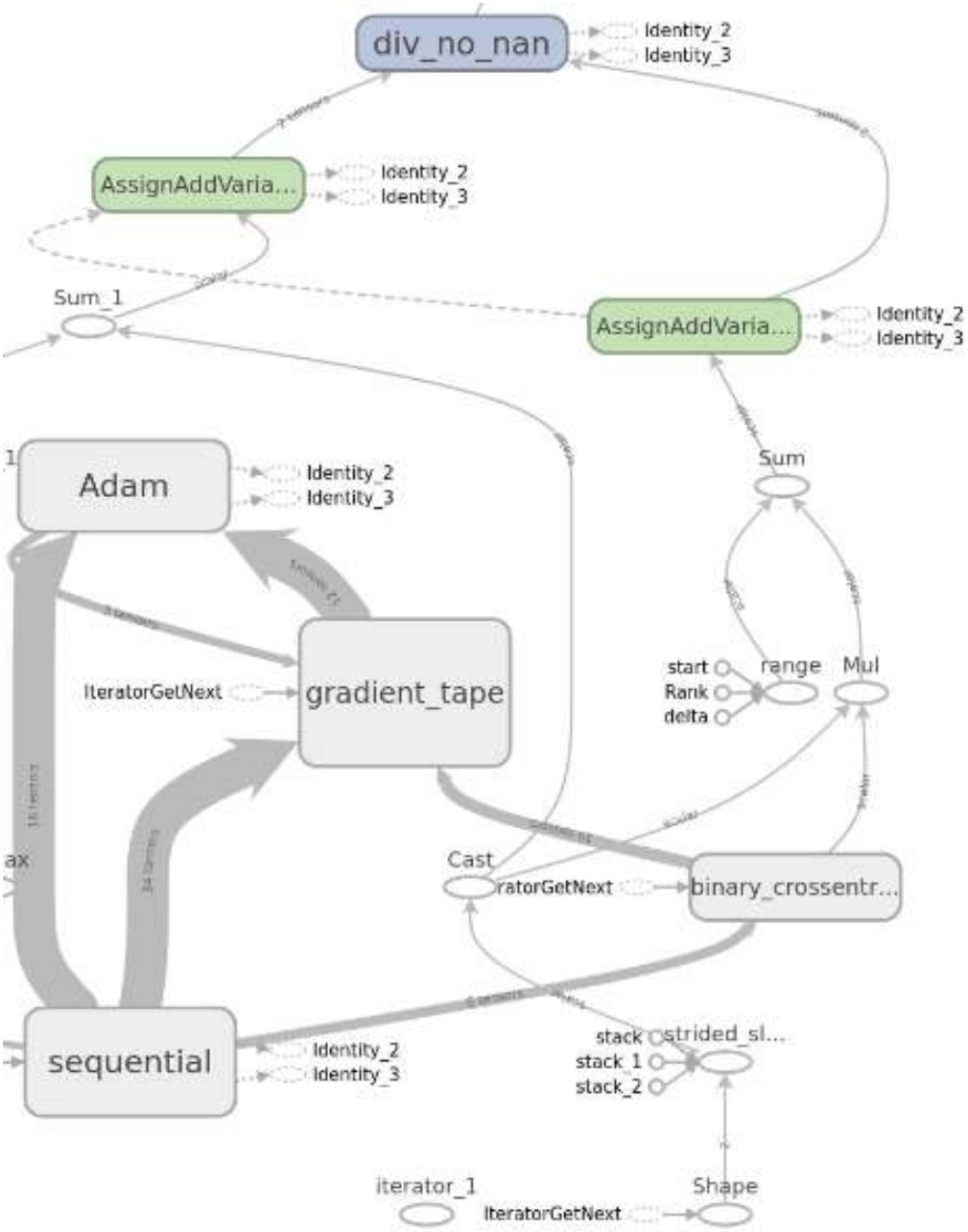
The second part of the tensorboard showing the flow of the model in detail.



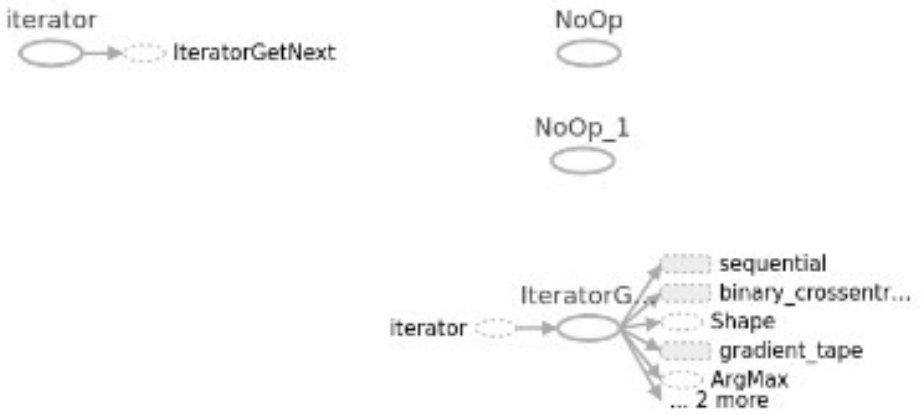
The third part of the tensorboard showing the flow of the model in detail.



The fourth part of the tensorboard showing the flow of the model in detail.



The last part of the tensorboard showing the flow of the model in detail



4.3 System Running

4.3.1 Reading the dataset

The dataset each folder consists of the protein text file and its site text file, and these files consist of the coordinates of each atom in the protein and their 3D location in space. To validate this data, we need to visualize it using VMD as unity will show the atoms overriding each other it will not be sufficient to understand the visualizing site and protein itself together. Therefore, the structure of the protein I had to use VMD to check if each protein has its corresponding site drawn above it on the same 3D space coordinates as will be shown in the next figure (Protein drawn by cartoon, and the site is drawn using lines).

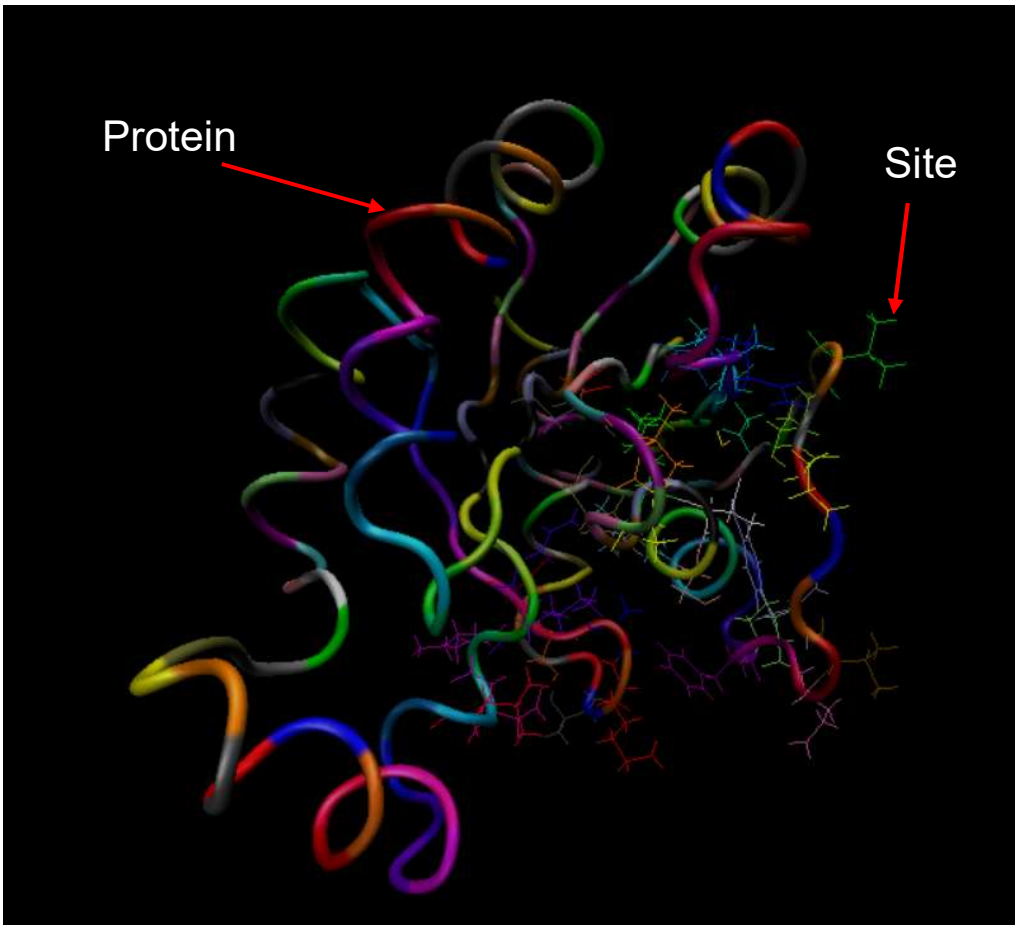


Figure 30: Protein and Site Visualized by VMD

Therefore, now as I validated that each protein and its site are drawn on the same 3D space I proceeded to the next phase which was reading this protein file into my system and here starts the potential challenges as we explore the protein files as many structures only include information about their biological data which means that the protein will have dummy atoms which will need to be removed and to be successfully read to the system and I used helper tool (HTMD) to help me read the molecules of the protein which reads the protein file without depending on any force field, and all operations are done to the 3D structure of the protein as this tool gets access to the protein attributes for example; atom name, residues name, charge of the atom, coordinate of the atom, and the shape of the protein. For

example, if we took protein “3PTB” and printed the file read by the tool it will show the number of atoms and number of atoms.

```
>>> mol = Molecule( '3PTB', name='Trypsin' )
>>> print(mol)
Molecule with 1701 atoms and 1 frames
```

Figure 31: 3PTB protein loaded by HTMD

Accordingly, the coordinates of atoms are read by making the number of atoms * 3 (X,Y,Z) * number of the frames to be able to have the correct coordinates of the protein. Added to that, propka tool helped me in predicting pKa values of the ionizable groups in protein complex based on their 3D structure which was needed to be done to be able to load the file successfully with the correct coordination, and then it detects the hydrogen bonds and try to optimize these bonds. When this step is done, propka starts to put the force field and naming scheme to the protein atoms which will allow us now to start performing any task on the protein as it is biologically and chemically optimized.

```
Optimization progress:
  Detecting potential hydrogen bonds:
    0% |                    | 100%
       ********************
  Optimizing hydrogen bonds:
    0% |                    | 100%
       ********************
Applying the forcefield to the protein...Done.
Applying the naming scheme to the protein...Done.
```

Figure 32: Propka optimization process.

4.3.2 Voxelization and Extracting Features

We started by going through the dataset and drawing a bounding box which takes a buffer of 1Å [Angstrom] to make sure all atoms are taken to the next step which is to voxelize each protein by vector of (16,16,16) which stands

for 4096 voxels fixed implemented on the protein to have the same input shape when passing to the mode and to have it optimized correctly. Moreover, we had to calculate the voxel size as not all protein has same number of atoms so for successful voxelization, we had to obtain the difference between min and max coordinates and then divide it by the number of each voxel for each coordinate direction and use this voxel size to know each voxel coordinates. Added to that, we started to perform the extraction of the 8 features for each voxel which are Hydrophobic, Aromatic, Hydrogen acceptor, Hydrogen donor, Positive ionizable, Negative ionizable, Metal, and Volume. Furthermore, we calculated the occupancy of each feature and obtained the maximum occupancy for the 8 features for each voxel, as well we had to calculate the center of each protein that will be essentially used in the next steps.

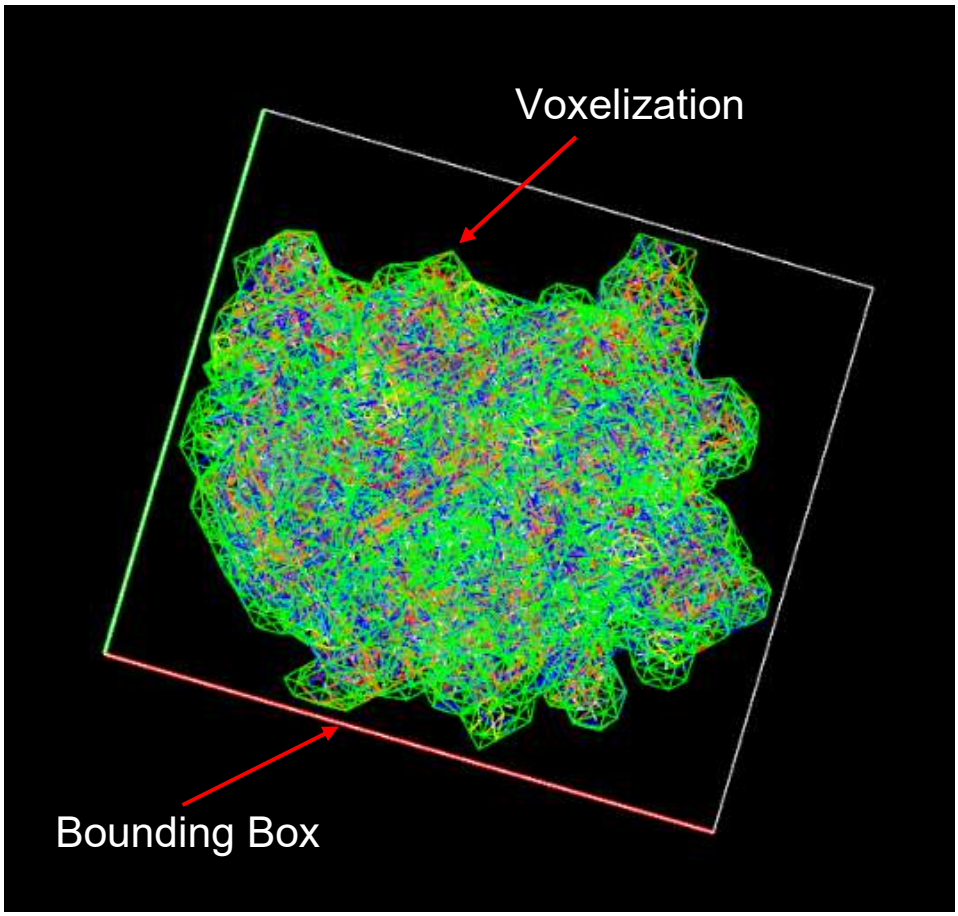


Figure 33: Voxelization and Bounding box created on protein sample.

4.3.3 Creating Train and Validation Data

Firstly, we had to go for each protein and calculate the voxel size as each protein has different number of atoms. Accordingly, the next step was to go through protein atoms coordinates and check each atom is in which voxel and give the atom the voxel index. Added to that, we read also the site atoms coordinates and check which atom is available in the protein and in the site and if the condition is true, we say that the voxel this atom belongs to is binding. Lastly, we create csv file containing each voxel and if it is binding or not and this will become the validation data, and for analysis we made file knowing the number of binding sites and non-binding site in each protein, with the percentage of the binding sites in each protein.

Number of binding Voxels	153
Total Number of Voxels	4096
percent of the binding voxels	4%

Figure 34: Protein Analysis.

Atom Number	X-coord	Y-coord	Z-coord	Voxel-x	Voxel-y	Voxel-z	Isbinding	Vox Number
1031	-9.144	31.128	19.031	-3.44563	13.35278	5.895087	0	168
1019	-9.084	31.448	19.976	-3.42302	13.49005	6.187813	0	184
1030	-9.808	32.025	20.355	-3.69583	13.73756	6.305213	0	184
935	0.611	28.404	25.628	0.230236	12.18428	7.93859	1	1178
940	1.922	28.063	27.169	0.724244	12.03801	8.415934	0	1179
1153	1.971	30.744	7.018	0.742709	13.18806	2.173912	0	1187
1151	2.42	29.68	8.966	0.9119	12.73164	2.777329	0	1188
1079	0.688	29.714	12.39	0.259251	12.74623	3.837956	0	1189
1080	0.675	29.354	11.214	0.254352	12.5918	3.473675	0	1189
1091	0.917	30.97	12.751	0.345542	13.285	3.94978	0	1189
1055	1.668	29.681	15.397	0.628533	12.73207	4.769411	0	1190
1077	0.504	29.225	14.88	0.189916	12.53646	4.609264	0	1190
1078	0.42	28.708	13.509	0.158264	12.31469	4.184579	0	1190
1100	0.906	31.211	13.721	0.341397	13.38838	4.250249	0	1190
1053	0.493	31.011	17.166	0.185771	13.30259	5.317381	0	1191
1054	1.663	30.203	16.843	0.626649	12.95599	5.217328	0	1191
1057	1.79	29.059	17.852	0.674504	12.46525	5.529877	0	1191
1065	-0.035	30.801	17.989	-0.01319	13.21251	5.572315	0	1191
1067	1.923	29.464	18.757	0.724621	12.63898	5.810213	0	1192
992	0.954	31.228	22.611	0.359485	13.39568	7.004036	0	1193
936	0.525	28.672	24.426	0.19783	12.29925	7.566256	1	1194
934	1.878	28.707	26.405	0.707664	12.31426	8.179276	1	1195
937	1.789	30.109	26.995	0.674128	12.91567	8.362035	1	1195
942	0.835	30.295	27.23	0.314643	12.99545	8.434829	0	1195
943	2.096	30.764	26.305	0.789811	13.19664	8.148299	0	1195
945	2.357	29.607	28.913	0.88816	12.70033	8.956159	0	1196
949	2.217	30.86	30.675	0.835406	13.23782	9.50196	0	1196
1092	1.184	32.006	11.758	0.446153	13.72941	3.642186	0	1205
1095	0.749	33.368	12.29	0.282237	14.31366	3.806979	0	1205

Figure 35: Created Data.

4.3.4 Visualizing Results

The output from the model will be if the voxel number and if it is binding or not therefore, we had to check each voxel and what are the atoms existing in it and then merge them together and visualize it by loading the whole protein but only choose the atoms that exists in the binding sites to be only seen and hide the other atoms of the non-binding site to view the expected and put it in comparison with the output.

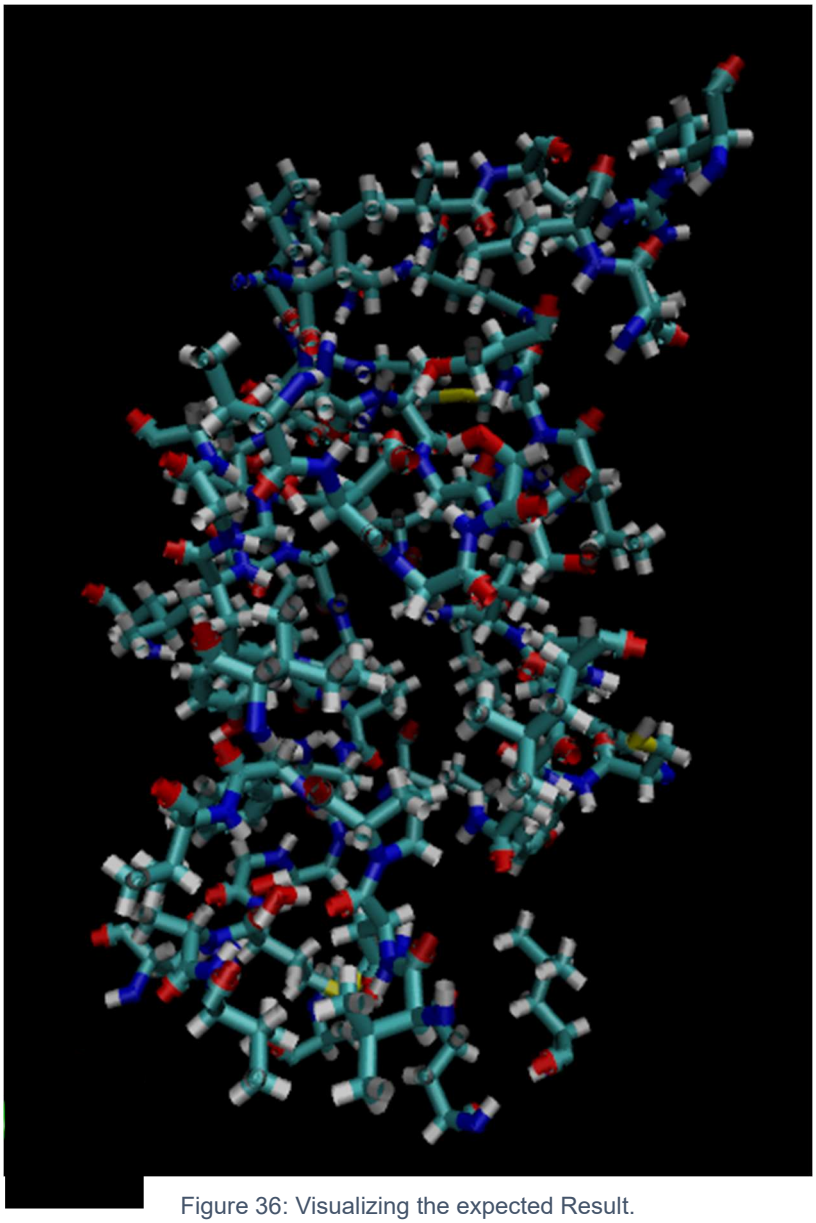


Figure 36: Visualizing the expected Result.

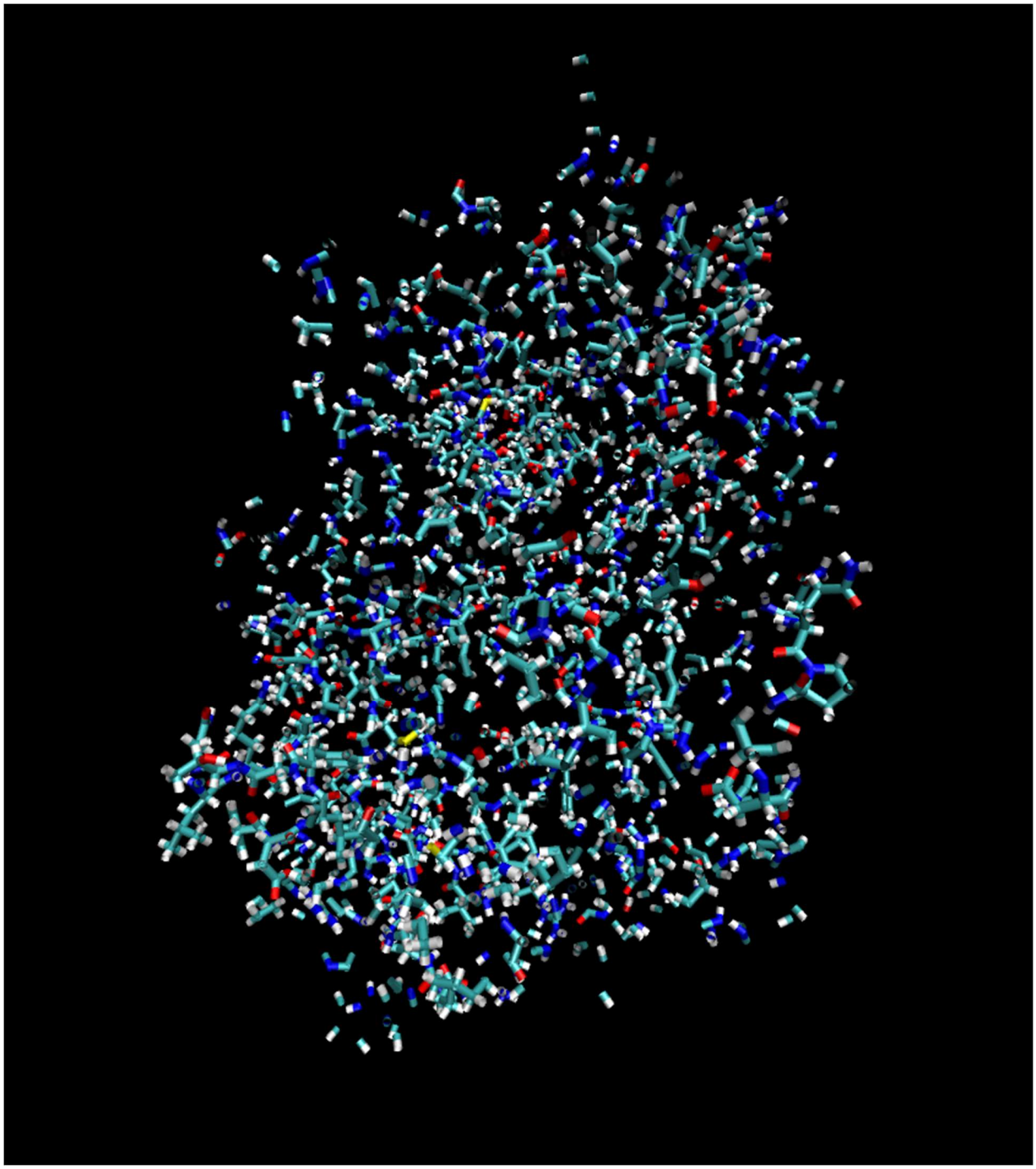


Figure 37: Visualizing the Output Result

Chapter 5: Results and evaluation

5.1 Testing Methodology

The used methodology in CNN model to classify a voxel if it is binding or not, which will be validated using the created csv of the positive voxels and compare it with the output from model, However, due of the in-balancing of the voxels we are implementing K-Fold technique.

5.2 Results

Expected 1cbo_1

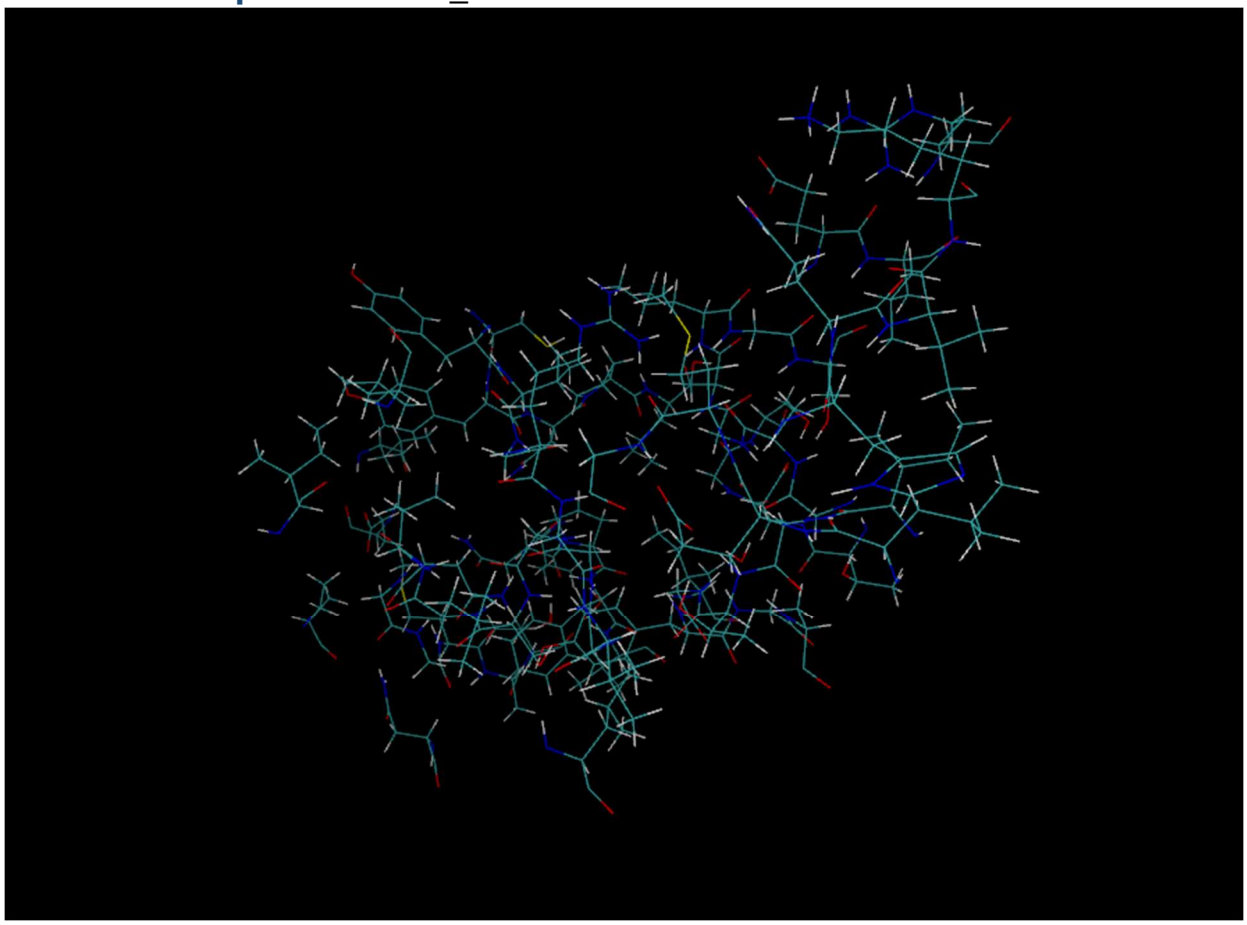


Figure 38: Expected Site

Output from model 1cbo_1 **After 800 Epochs**

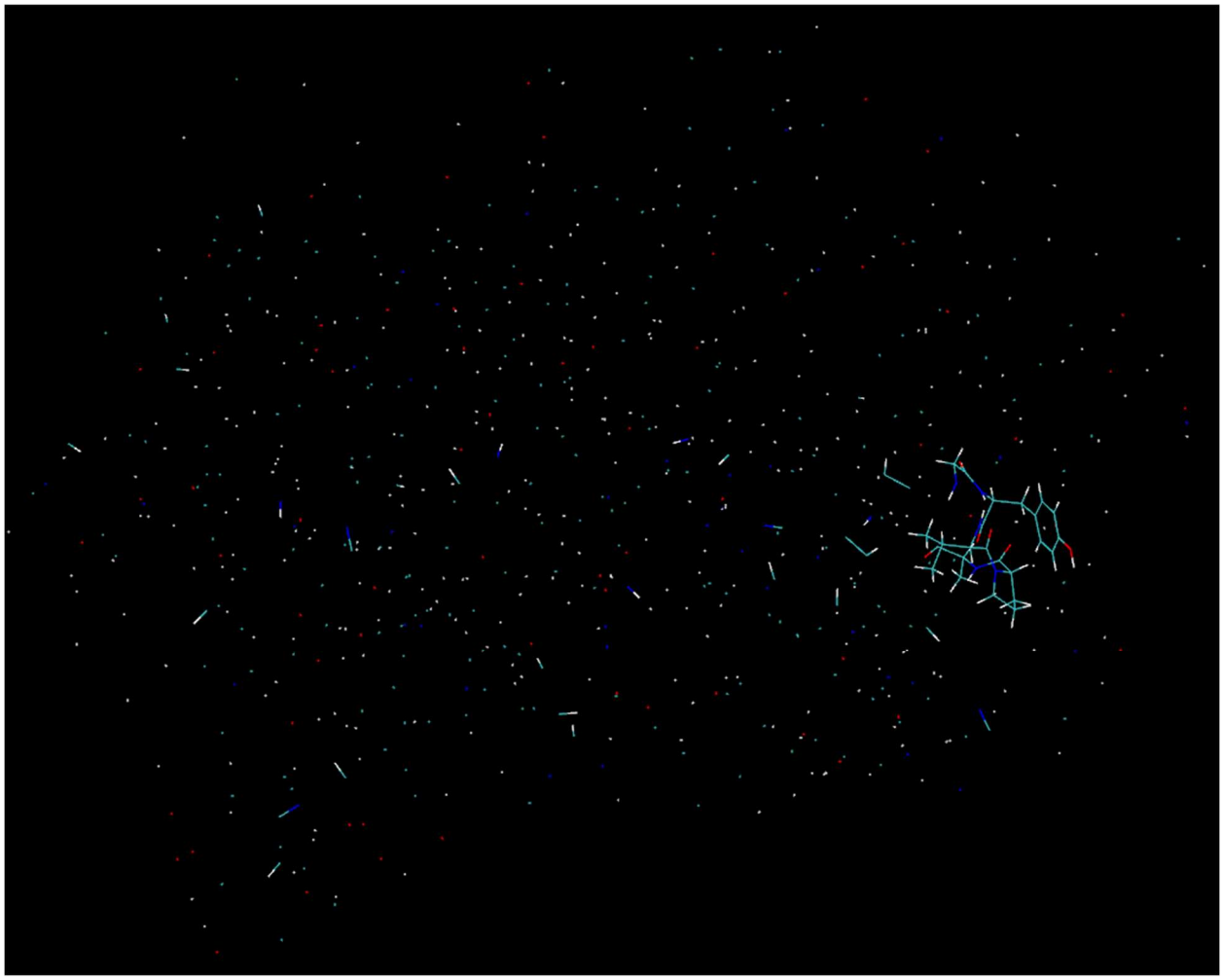


Figure 39: Predicted Site.

Expected 3a0i_1

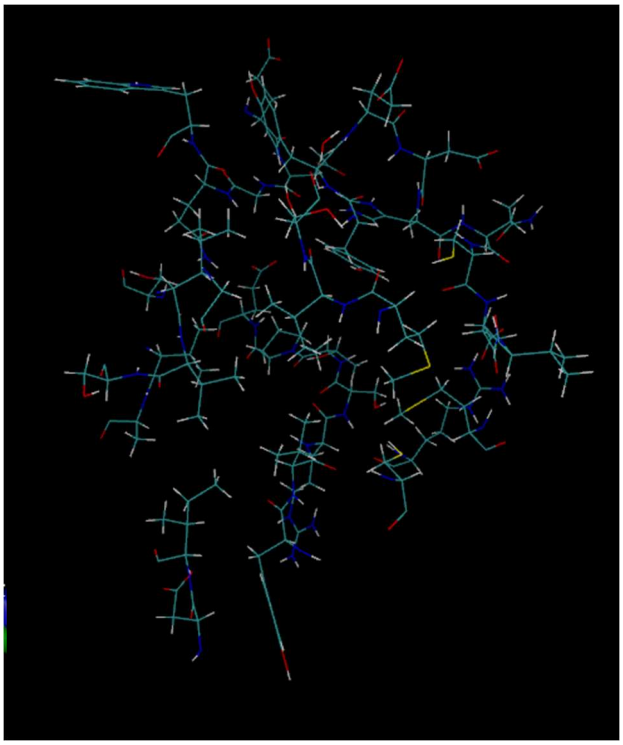


Figure 40: Expected Site.

Output from model 3a0i_1 After 800 Epochs

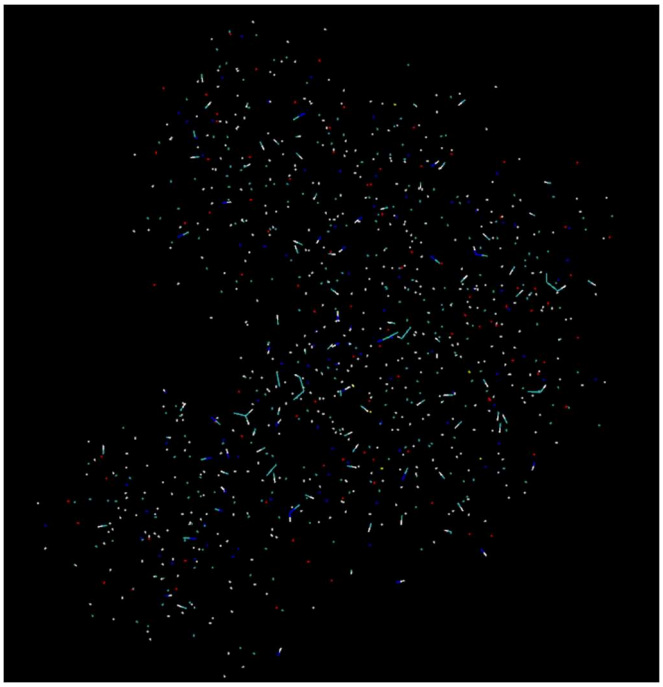


Figure 41: Predicted Site

Expected 3a0t_1

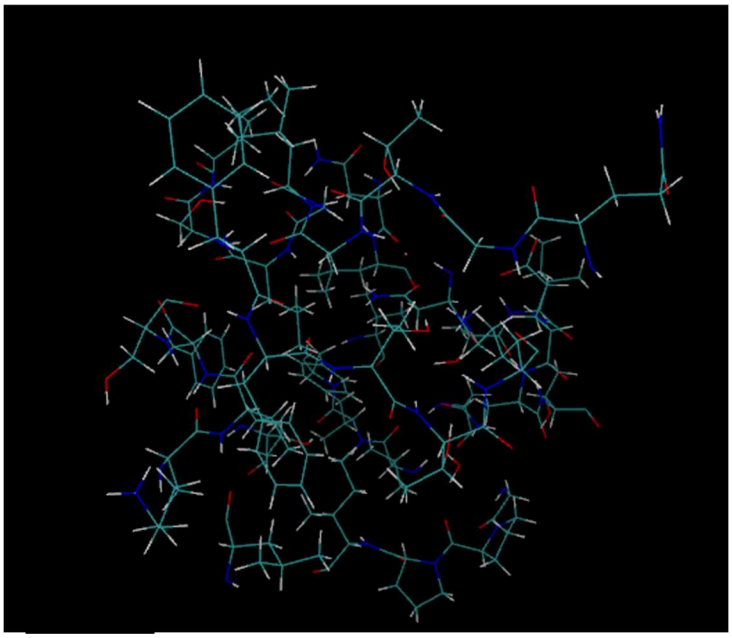


Figure 42: Expected Site.

Output from model 3a0t_1 After 800 Epochs

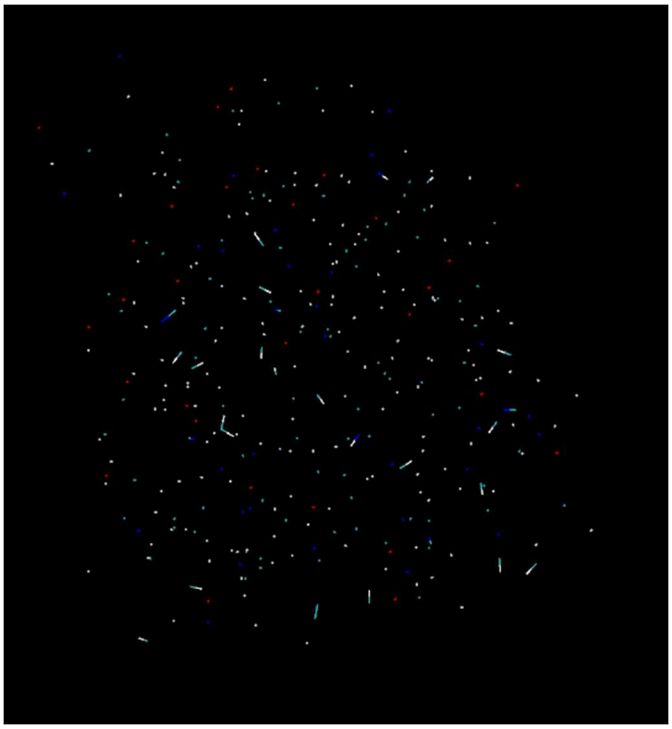


Figure 43: Predicted Site.

5.2.1 Limitations

The dataset was one of the challenges that was faced in this project as we had to make multiple preprocessing on it to fit our system. Firstly, the protein had many dummy atoms which was not possible for HTMD to read the protein, there was unsupported atoms, and there were other errors such as:

- termini
- multiple chains
- nucleic acids
- coupled titrating residues
- Disulfide bridge detection

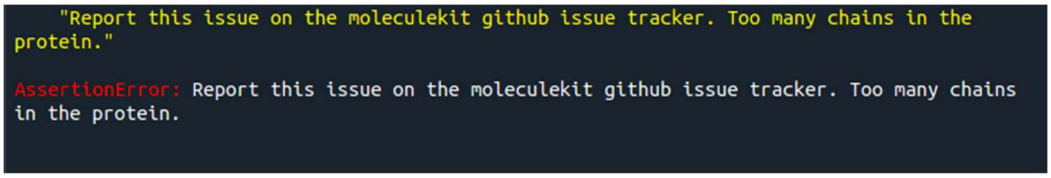


Figure 44: Example of the faced errors.

Accordingly, we had to go through each protein individual and run the reading code then we check each protein what is the faced error and what are the dummy atoms that exist (-1 written in cell if protein does not contain dummy atoms). Furthermore, we created csv for each protein taken in the system what are the atoms that we should and if the protein is supported to be read into the system.

Protein Name	Removed Dummy atoms	Error If Found	Taken to Dateset
1a2b_1	HOH	-	TRUE
1a2n_1	HOH	-	TRUE
1a4i_1	HOH,NDP	-	TRUE
1a4L_2	HOH,DCF,GLN	Too many chains in protein	FALSE
1a4r_1	HOH,GDP	-	TRUE
1a4w_1	HOH,NA5	-	TRUE
1a4z_4	HOH,NAD,SM5	-	TRUE
1a5b_1	HOH,PLP	-	TRUE
1a5s_1	HOH,PLP,NA2,SER	-	TRUE
1a5u_5	HOH,NA4	-	TRUE
1a7k_4	HOH,NAD	-	TRUE
1a7x_1	HOH	-	TRUE
1a8g_1	HOH	-	TRUE
1a8k_1	HOH	-	TRUE
1a8p_1	HOH	-	TRUE
1a8r_2	HOH,GTP	-	TRUE
1a8t_1	HOH,LYS	Too many chains in protein	FALSE
1a9c_10	HOH,GTP	-	TRUE
1a9m_1	HOH	-	TRUE
1a9p_1	HOH	-	TRUE
1a9q_1	HOH	-	TRUE
1a9r_1	HOH	-	TRUE
1a9s_1	HOH	-	TRUE
1a9t_1	HOH	-	TRUE
1a9x_1	HOH,ADP,CL1	-	TRUE
1a9y_1	HOH,NA4,NAD	-	TRUE
1a9z_1	HOH,NA4,NAD	-	TRUE
1a26_1	HOH	-	TRUE
1a27_1	HOH,NAP	-	TRUE
1a29_1	HOH,TFP	-	TRUE
1a42_1	HOH,PHE	-	TRUE
1a50_1	HOH,NA2,PLP	-	TRUE
1a59_1	-1	-	TRUE
1a69_2	HOH,FMB,THR	-	TRUE
1a71_2	HOH,NAD	-	TRUE
1a72_1	HOH	-	TRUE
1a80_1	HOH	-	TRUE
1a94_1	HOH	-	TRUE
1aa6_1	HOH,MGD,SF4	-	TRUE
1acj_1	HOH	-	TRUE
1ad3_2	HOH,NAD	-	TRUE
1ad5_2	HOH,ANP	Segment error (was 9 in the protein) propka library predicted it to be 7	FALSE
1adb_1	HOH,CND	-	TRUE
1adc_1	HOH,PAD	-	TRUE
1adf_1	HOH	-	TRUE
1ads_1	HOH	-	TRUE
1ae1_2	HOH,NAP	-	TRUE
1ae8_1	HOH,NAG	-	TRUE
1af0_1	HOH	-	TRUE
1af7_1	HOH	-	TRUE
1afe_1	HOH,NAG	-	TRUE
1afs_1	HOH,NAP,TES	-	TRUE
1agn_3	HOH,NAD	-	TRUE
1agw_1	HOH,SU2,THR	Too many chains in protein	FALSE
1ah0_1	HOH,NAP	-	TRUE
1ah3_1	HOH,NAP	-	TRUE
1ah4_1	HOH	-	TRUE
1ahb_1	HOH	-	TRUE
1ahg_2	HOH	-	TRUE
1ahh_2	HOH,NAD	-	TRUE
1ahi_2	HOH,CHO,NAI	-	TRUE
1ahn_1	HOH	-	TRUE
1ai0_3	HOH,IPH	-	TRUE
1ai9_2	HOH,NDP	-	TRUE
1aiy_6	HOH,IPH	-	TRUE
1aj0_1	HOH,SAN	-	TRUE
1aj0_2	HOH,PH2	-	TRUE
1aj2_1	HOH	-	TRUE
1aj8_1	HOH,COA	-	TRUE
1ajv_1	HOH	-	TRUE
1ajx_1	HOH	-	TRUE
1aka_1	HOH	-	TRUE
1akb_1	HOH	-	TRUE
1akc_1	HOH	-	TRUE
1ake_1	HOH,AP5	-	TRUE
1akr_1	HOH	-	TRUE
1aku_1	HOH	-	TRUE
1akw_1	HOH	-	TRUE

Figure 45: Csv file created to identify each protein dummy atoms.

5.3 Evaluation

5.3.1 Accuracy Evaluation

Before applying K-fold technique for the model accuracy

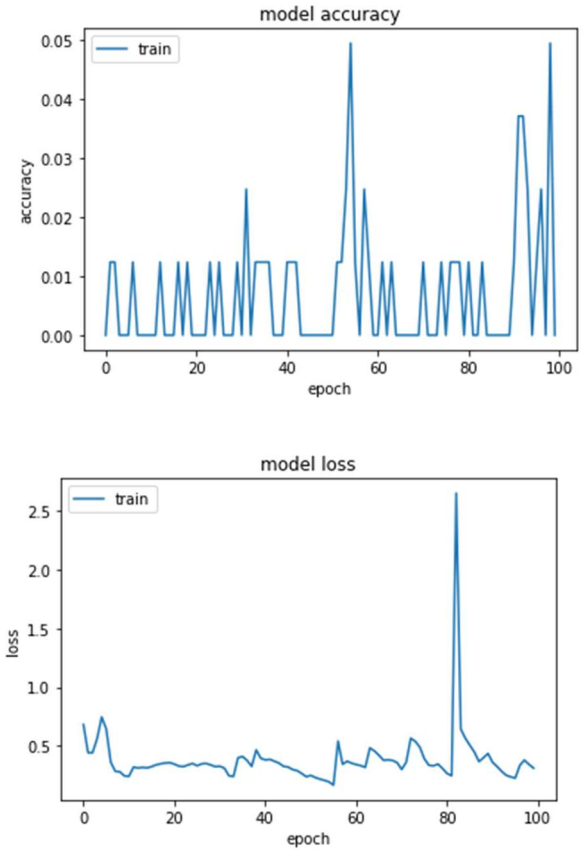


Figure 46: Model accuracy and loss.

After applying K-fold technique for the model accuracy:

First try of 5 folds and 10 epochs on 50 protein:

```
Average scores for all folds:
> Accuracy: 0.0 (+- 0.0)
> Loss: 35.70107877254486
------------------------------------------------------------------------
```

Second try of 10 folds and 25 epochs on 50 protein:

```
Average scores for all folds:
> Accuracy: 0.7692307978868484 (+- 1.538461595773697)
> Loss: 0.27665482759475707
------------------------------------------------------------
```

5.3.2 Time Performance

According to the help tools that was not supported we could not use Google Colab which was going to enhance the performance, we had to use spyder and keras on local pc CPU running on ubuntu. The time taken to voxelize the 100 protein it took approximately 14 hrs. Added to that, to read the dataset to use it for the model it takes approximately 2 hrs., and the model takes 17s per epoch with 3s/step.

Chapter 6: Conclusion and Future Work

6.1 Conclusion

In conclusion, in our thesis we developed system that was able to read protein and extract its features. Moreover, after reading and searching in the same area of our work which introduced us to several approaches which allowed us to determine our system approach to achieve our goal, that was applying deep learning approach “CNN” that is considered to be an appropriate approach for our protein 3D structure and the extracted features to achieve the result of obtaining the binding site for the protein.

6.2 Problem Issues

6.2.1 Technical issues:

Firstly, as mentioned before we used “HTMD” as helper tool to read the protein, however this tool is not supported by windows which caused us to use ubuntu to allow the installation for this tool. Secondly, HTMD has tool that allows it to optimize the PH of the protein “Propka” which was not working on google colab, causing us to develop our system locally on ubuntu on CPU.

At the end, it caused us to not be able to implement our solution on the whole dataset as it starts to crash and the kernel restarts.

6.2.2 Data issues:

Added to that, proteins that passes the 20,000 atoms the voxelization starts to have difficulty to run it with other protein in the loop function as the memory reaches its max and the voxelization fails as well as the kernel fails, and it restart itself.

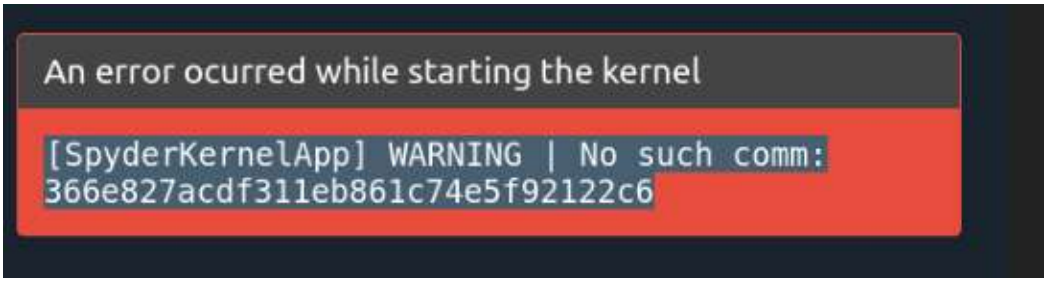


Figure 47: Kernel fail after memory reach its maximum.

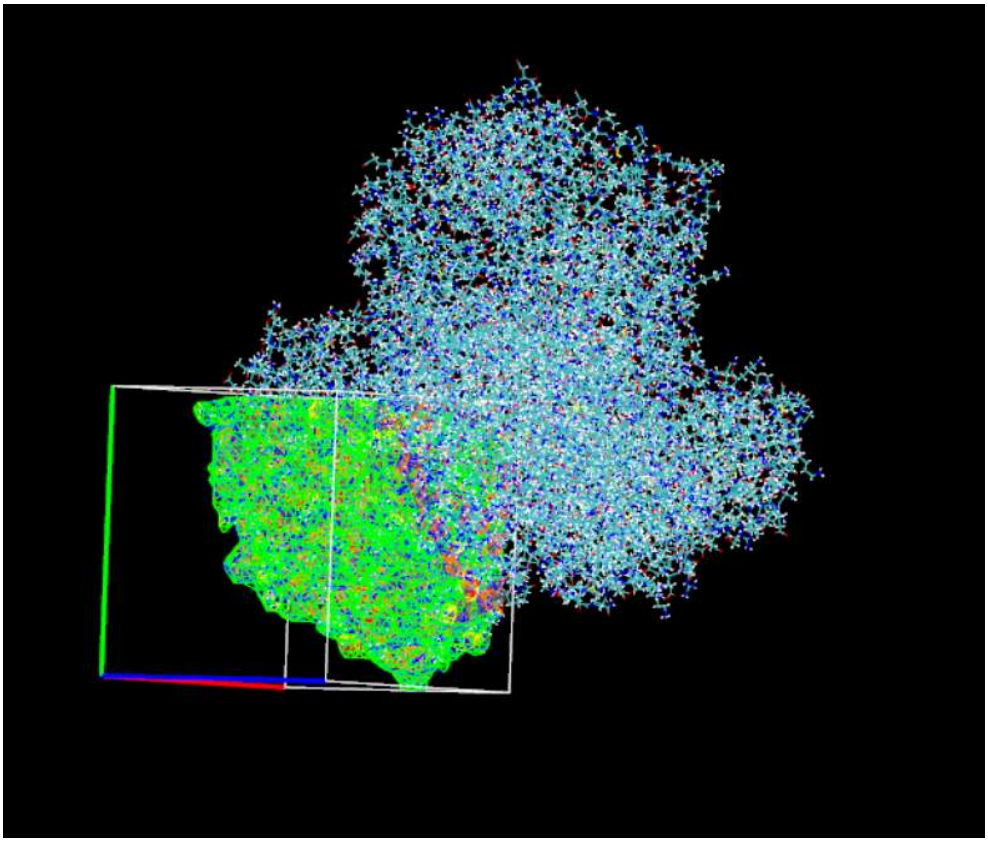


Figure 48:Visualization of protein voxel after it failed.

In addition to, we had in-balance in the voxels as from 4096 there may be less than 10% binding site which affects the accuracy of the model, which causes accuracy of 0% as it is not learning from enough data.

Number of binding Voxels	153
Total Number of Voxels	4096
percent of the binding voxels	4

Figure 49: Binding Voxels percentage of protein 1a2b_1.

6.2.3 Scientific issues:

To allow the model to train for as many epochs as we wanted we needed to save the weights to be able to load it and continue training after resting the used laptop or to load the weights for future predict for the protein which was not possible as the file was too large, and it failed to save it.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv3d_44 (Conv3D)           (None, 8, 16, 16, 16)     131088
_________________________________________________________________
batch_normalization_33 (Batc (None, 8, 16, 16, 16)     64
_________________________________________________________________
conv3d_45 (Conv3D)           (None, 8, 16, 16, 32)     32800
_________________________________________________________________
dropout_33 (Dropout)         (None, 8, 16, 16, 32)     0
_________________________________________________________________
batch_normalization_34 (Batc (None, 8, 16, 16, 32)     128
_________________________________________________________________
conv3d_46 (Conv3D)           (None, 8, 16, 16, 48)     98352
_________________________________________________________________
dropout_34 (Dropout)         (None, 8, 16, 16, 48)     0
_________________________________________________________________
batch_normalization_35 (Batc (None, 8, 16, 16, 48)     192
_________________________________________________________________
conv3d_47 (Conv3D)           (None, 8, 16, 16, 64)     196672
_________________________________________________________________
dropout_35 (Dropout)         (None, 8, 16, 16, 64)     0
_________________________________________________________________
flatten_11 (Flatten)         (None, 131072)            0
_________________________________________________________________
dense_11 (Dense)             (None, 4096)              536875008
=================================================================
Total params: 537,334,304
Trainable params: 537,334,112
Non-trainable params: 192
```

Figure 50: Model parameters

6.3 Future Work

For our future work, we want to our system to include another feature that we give it ligand “small protein” which interacts with another protein so after identifying the binding site we be able to identify which ligand is allowed to bind correctly which will allow our system to be applied in multiple ways. In addition to that, if we found fund or team to work with we can build our system on server to be able to solve our size of parameters limitation.

Reference list:

1. Brady Jr., G., & Stouten, P. (2000). Fast prediction and visualization of protein binding pockets with PASS. Journal Of Computer-Aided Molecular Design, 14(4), 383-401. doi: 10.1023/a:1008124202956.

2. Chen, K., Mizianty, M., Gao, J. and Kurgan, L., 2011. A Critical Comparative Assessment of Predictions of Protein-Binding Sites for Biologically Relevant Organic Compounds. Structure, 19(5), pp.613-621.

3. Convolutional Neural Networks and Caffe | Details | Hackaday.io. (2021). Retrieved 6 February 2021, from https://hackaday.io/project/26979-vision-based-grasp-learning-for-prosthetics/log/65975-convolutional-neural-networks-and-caffe.

4. Huang, B., & Schroeder, M. (2006). LIGSITEcsc: predicting ligand binding sites using the Connolly surface and degree of conservation. BMC Structural Biology, 6(1), 19. doi: 10.1186/1472-6807-6-19

5. Jian, J., Elumalai, P., Pitti, T., Wu, C., Tsai, K., Chang, J., Peng, H. and Yang, A., 2016. Predicting Ligand Binding Sites on Protein Surfaces by 3-Dimensional Probability Density Distributions of Interacting Atoms. PLOS ONE, 11(8), p.e0160315.

6. Jiménez, J., Doerr, S., Martínez-Rosell, G., Rose, A., & De Fabritiis, G. (2017). DeepSite: protein-binding site predictor using 3D-convolutional neural networks. Bioinformatics, 33(19), 3036-3042. doi: 10.1093/bioinformatics/btx350

7. Krivák, R. and Hoksza, D., 2018. P2Rank: machine learning based tool for rapid and accurate prediction of ligand binding sites from protein structure. *Journal of Cheminformatics*, 10(1).

8. Mason, S., Chen, B., & Jagodzinski, F. (2018). Exploring Protein Cavities through Rigidity Analysis. Molecules, 23(2), 351. doi: 10.3390/molecules23020351

9. Merghadi, Abdelaziz & Yunus, Ali P. & Dou, Jie & Whiteley, Jim & Thaipham, Binh & Bui, Tien & Avtar, Ram & Boumezbeur, Abderrahmane & Pham, Binh. (2020). Machine learning methods for landslide susceptibility studies: A comparative overview of algorithm performance. Earth-Science Reviews. 10.1016/j.earscirev.2020.103225.

10. Sonka,M. et al. (1998) Image Processing, Analysis, and Machine Vision. Chapter 5, Border tracing, 2nd edn. Pws Pub Co., pp. 142.

11. Weisel, M., Proschak, E., & Schneider, G. (2007). PocketPicker: analysis of ligand binding-sites with shape descriptors. Chemistry Central Journal, 1(1). doi: 10.1186/1752-153x-1-7

12. Xu, Y., Wang, S., Hu, Q., Gao, S., Ma, X., & Zhang, W. et al. (2018). CavityPlus: a web server for protein cavity detection with pharmacophore modelling, allosteric site identification and

covalent ligand binding ability prediction. Nucleic Acids Research, 46(W1), W374-W379. doi: 10.1093/nar/gky380

13. Yang J, Roy A, Zhang Y (2013) Protein-ligand binding site recognition using complementary binding-specific substructure comparison andsequence profile alignment. Bioinformatics 29(20):2588–2595

14. Yu, J., Zhou, Y., Tanaka, I. and Yao, M., 2009. Roll: a new algorithm for the detection of protein pockets and cavities with a rolling probe sphere. Bioinformatics, 26(1), pp.46-52.