Desta, Israel
May 6, 2019

Utilizing Data Mechanics Principles to Pre- and Post-process Data for Protein-Protien Docking

Abstract

Protein-Protien Docking is a computational tool that is used to predict how two proteins interact and uses a great deal of data. While the process of docking itself is beyond the scope of the class, the pre and post processing steps involves a great deal of data manipulation. As such, in this project, tools from the relational paradigm were used to make the said steps more efficient and general. Tools such as selection, projection, aggregation and more were used. After docking, the results were analyzed with 4 widely accepted evaluative scores. The second part of the project implemented Z3 solver to find scaling factors for 2 of the evaluative scores as implemented on a recent research paper. Furthermore, correlation coefficients and the corresponding p values for between each of the 4 scores were calculated to identify the most important aspect for predicting the right structure of protein. Lastly, to better understand the features that are important for increasing prediction accuracy, the data was visualized using Chart.js and web server building tool flask.

Introduction

Proteins in the human body are essential biological components for the survival, maintenance, immunization and every other core processes of human life. Proteins are able to coordinate and implement all of this by interacting with other proteins by forming protein complexes – a single entity of one or more proteins. Understanding how, or in what way, these proteins interact is essential for pharmaceutical and fundamental research. Since the experimental methodologies to verify a protein complex, are extremely tedious, time consuming and expensive, computational tools have become more and more needed. ClusPro is an automated webserver, open for academic users, that docks two proteins given by the user. While most parts of the process of docking two or more proteins is beyond the scope of CS504 and some do involve data mechanics.

However, due to the complexity of the software and time constraints, this project focuses on the pre and post processing of data for and from ClusPro. Therefore, it is important to understand just the inputs and outputs of the software. ClusPro requires just two inputs: protein 1 and protein 2 which are the component proteins of the complex. These two proteins are text files that outline the different atoms, the coordinates and physico-chemical properties of each atom. ClusPro eventually outputs three types of files: i) 30 predicted models of the protein complex (protein 1 and 2 bound at some particular location), ii) a cluster file outlining 1000 models and the more than 30 clusters that each belong to, and iii) an ftfile that contains 6 different energy potentials that are calculated for 70,000 models.

As a researcher in the lab that developed ClusPro, my goal is to improve the accuracy of the software. In order to work towards that goal, current performance of the software should be evaluated using a well-accepted benchmark set of protein complexes with the true protein structures of the complexes as well as the component proteins already solved. In order to check the accuracy of ClusPro's predictions, the predictions need to be compared with the true structures of the complex from the benchmark set.

In order to compare them with the true structures, the chains of the component proteins need have the same names as those same proteins in the true structure. So, it is important that the names of the chains are mapped to the true chains for all proteins in the benchmark set. After mapping the true chain names of each protein, the input files ought to be changed to the true chain names. After submitting the component proteins to ClusPro and obtaining the 3 aforementioned files, there are several post analyses steps that will be useful for improving ClusPro's predictions. Some of the features that are important are the ranges of energy potentials from the 70,000 models, cluster populations (counts) of the 30 clusters mentioned above and the true accuracy score of each top model using the 4 evaluative scores that are independent yet highly correlated. The evaluative scores also give us a hint about the weakness and potential improvement points for the docking tool used.

Doing the pre- and post-processing steps mentioned in the previous paragraph for a researcher, like me, who is aiming to improve ClusPro is very tedious and not very efficient. In order to make this process, more efficient and informative, different tools of data mechanics ought to be used. In the project described in this report, I tried to showcase how the above steps could be made more efficient. Furthermore, the results are visualized in a way that potentially highlight otherwise hidden trends or patterns.

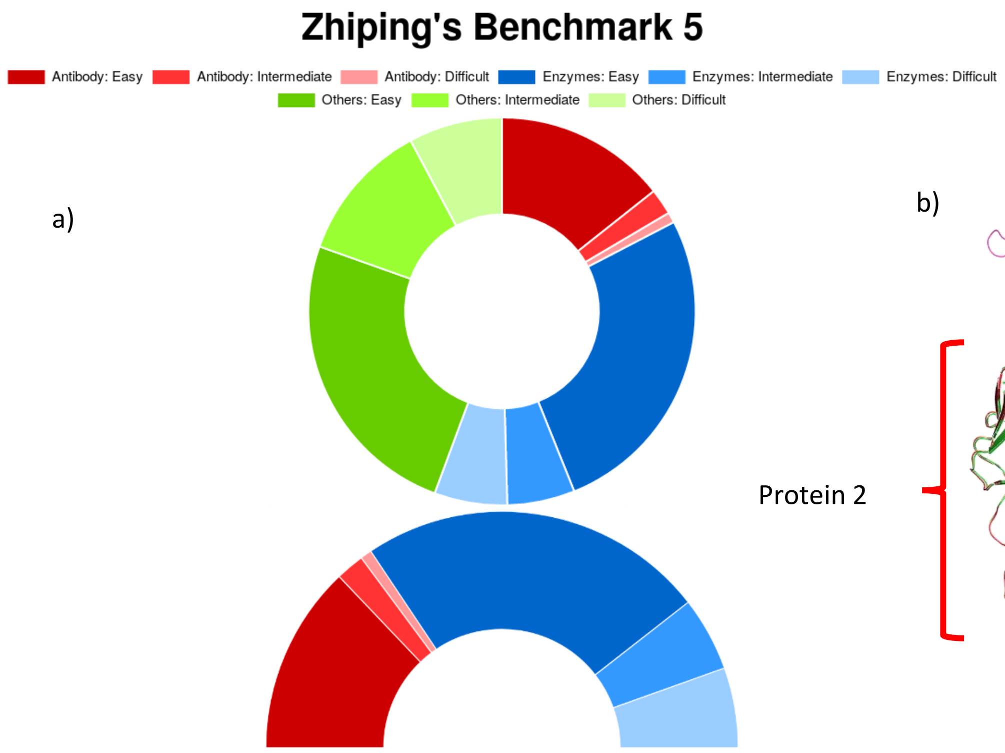
Methods

Workflow of Project 1: relational paradigm

0. loadData.py loads all 5 data sets from the 3 data portals: RSCB PDB, Weng Lab at Umass, and datamechanics.io
1. Obtain the well accepted complex benchmark set from Zhiping's lab website. This benchmark set contains 230 complexes of different types. Since ClusPro uses different parameters for the different classes and difficulty levels of proteins, it is important to keep the classes and difficulty levels in mind. The number for each type are included in a Chart.js graphic that might be useful for users in figure 1.a. The information obtained also includes the category of the complex, the chains for the receptor and the ligand, the change area of interface (ASA), and the interface root mean square distance between bound and unbound complexes (iRMSD). # clusterBM5.py a. I use k-means clustering to see if the 230 complexes can be divided into different groups by ASA and iRMSD. b. I also get the mapping of protein ID to chain ID of each complex
2. Due to time constraints and complexity, I chose to do the post-processing steps only one of the cases: 1AHW which is rendered using PyMOL in figure 1.b. I use the Protein ID to chain ID map to change the chain IDs of the unbound proteins that make up 1AHW. (I also do that same for the bound just in case the chain IDs are different). # renamePDBfile.py
3. (I dock the unbound proteins of 1AHW using ClusPro (this is done outside of the project and involves no transformations)

4. I use the major output of the docking step which is an ft file. This file outlines the coordinates and the different types of energy values. I find the range of the energy values from the 70,000 rotation lines. # findEnergyRange.py
5. The other output from the docking step is the cluster file which gives the cluster centers and members of the top 1000 rotation lines in the ft file mentioned in (4). For this file, I count the number of members in each cluster. # countClusterMembers.py
6. Finally, after docking, we need to check if our simulation is correct or not. For that, one way is to check the bound 1AHW. Another way is to see how homologous complexes interact. So, for that I pulled proteins with similar sequences from the protein data bank and selected for similar complexes. # findHomologs.py

a)



b)

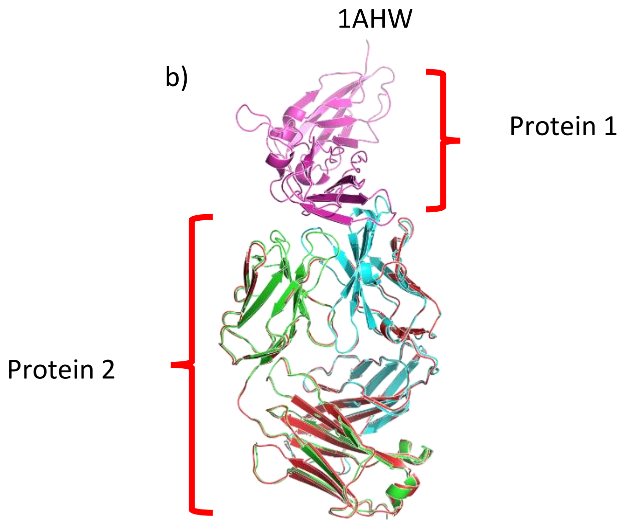


Figure 1: a) different types of proteins in the Zhiping's widely accepted benchmark set [3] and b) a PyMOL rendering of 1AHW, a protein complex from Zhiping's set: the protein that I chose to process for post-docking analysis

Desta, Israel
May 6, 2019

Workflow of Project #2: Z3Solver and statistical analysis

The DockQ program, which is a widely recognized software to evaluate accuracy, oututs 4 different scores for each ClusPro prediction: fnat, irmsd, Lrmsd and dockQ scores. fnat, irms and Lrms are somehow manipulated to obtain the final dockQ score which is used to tell how accurate the prediction is. In the first part of the project 2, I try to find the scaling factors for irms and Lrms that enables us to obtain dockQ from fnat, irms and Lrms by framing the problem as a constraint satisfaction problem with the following constraint:

$$TrueDockQ \; - 0.05 \leq \frac{fnat \; + \; \dfrac{1}{1+\left(\dfrac{iRMSD}{i}\right)^2} + \dfrac{1}{1+\left(\dfrac{LRMSD}{j}\right)^2}}{3} \leq TrueDockQ + 0.05$$

Equation A: constraint used for solving the scaling factors for iRMSD and LRMSD

Z3 solver from Microsoft Research was used for this problem. The dataset that I used was the text file which was obtained after running DockQ program on the top predicted models of 38 of the 230 complexes in Zhiping's benchmark set. The text file contained the indentifiers of the model, fnat, irmsd, lrmsd and dockQ scores. To this end the script loadData2.py loads the evaluation results from DockQ program that has the fnat, irmsd, lrmsd, and dockq scores of each of ClusPro's predictions. Then it uses relational building blocks to filter only the 4 scores and the identifying keys (proteinName, predictionRank, energyRank and coefficientNumber). The second script, constStats.py then the filtered dockQ dataset from repo and Finds the scaling factors of irmsd and lrmsd that yields the dockQ score that satisfy the constraint outlined above in equation A for all 96000+ predictions

The second goal which of the three fundamental scores are the most important for better dockQ scores. To that end, the correlation coefficients and the corresponding p-values of the four evaluative scores were calculated. Only the p-values are deposited in the repo as that tells all. constStats.py was written to retrieve the filtered dockQ dataset from repo and finds the p-value for the correlation coefficients of each of the 4 scores with each other.

Project #3: Chart.js and Flask

For project 3, the goal was to visualize different parts of the project for improved understanding for the users. For all three visualizations, Chart.js was used to plot the graphics.[2] Finally, flask API was used to build a webserver which users can access all three visualizations, and one of the raw data utilized to build the visuals.[1]

Visualization #1: Chart.js

For predictive software such as ClusPro, it is important to have a set of test cases to compare performances. One well accepted benchmark set with 230 proteins is one by Zhiping Weng lab.[3] Due to the differences in nature between the different classes and difficulty levels of proteins, it is important to evaluate ClusPro's performance on each differently as it is necessary to use different parameters for docking. With figure 1a., clients can see the spread of data, and if linked to the source link of the benchmark, can be very informative. The chart is interactive and can be made full and half/circle. Users can also remove different types of proteins to take a closer look at other types of proteins.

Visualization #2: Chart.js

The clustering that was obtained using relational tools in project #1, is an ideal candidate for an interactive scatter plot as it can show the clear boundaries between the different levels of difficulty. The x-axis represents changes of structure on the interface atoms before and after binding. The y-axis represents changes in surface area of the interface atoms before and after binding, also known as ΔASA. These numbers are calculated via comparing to the true structure and are obtained from Weng lab.[3] Users can float their cursor to look at individual points and see the actual ΔASA and ΔIRMSD values.

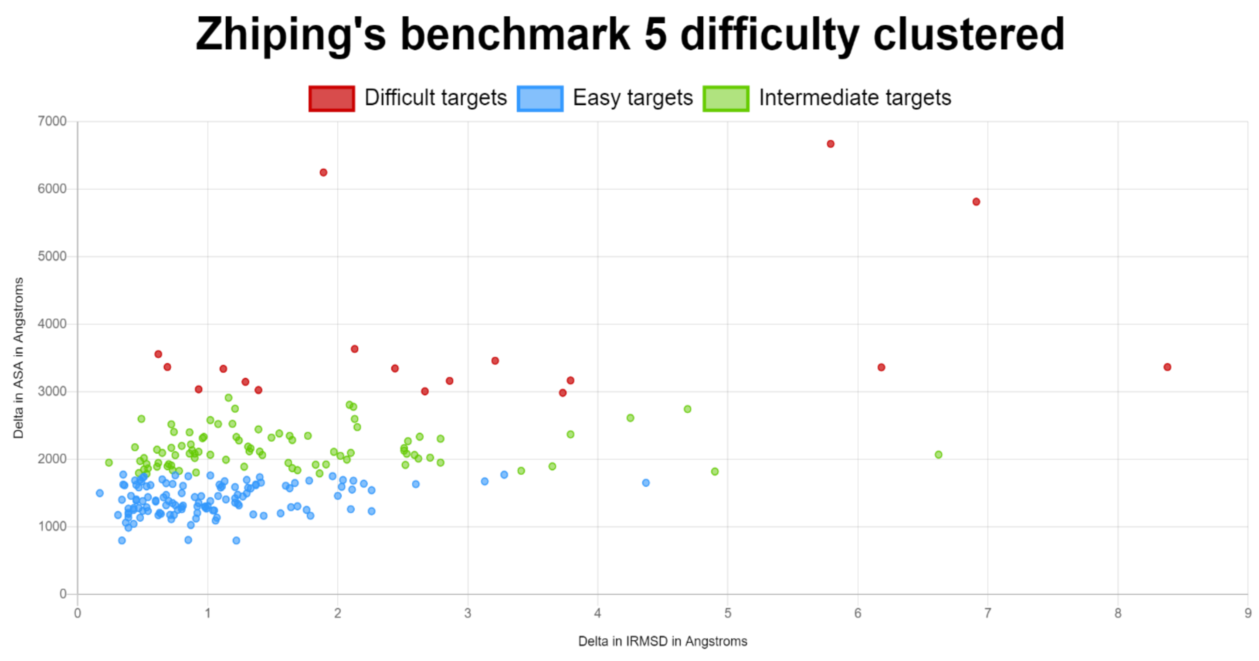


Figure 2: Scatter plot showing proteins clustered by difficulty level using change in IRMSD dna interface surface area as the only two parameters.

Visualization #3

The third visualization is 3D bubble chart with xaxis, yaxis and the radius of the bubbles conveying the cluster size (cluster rank), the accuracy score (dockQ) and model rank (based on energy value) respectively. As it stands, graph uses data from two proteins and 5 different models each. Users can remove data sets and hover their cursors to know the actual x, y and r values. Though the goal was to pull from a json file that had all the proteins and all models, due to time constraints that was not achieved.

Provenance

For all the first two projects, provenance was utilized to track data usage and transformations. The provenance diagrams consisting of all activities, entities and algorithms are shown in the provenence.html for project 1 and 2 .

Results and Conclusions

As shown in figure 2, the most defining characteristic that differentiates difficulty of proteins seems to be change in surface area of the interface. As shown, ΔASA above 3000 generally become a difficult target. However, note how the boundary between intermediate and easy is not that clear. While the K-means clustering resulted in 10 true positives for difficult cases out of 18, and 97 true positives for easy targets out of 121 predictions, the clustering did not do so well with intermediate cases. This is a good result since we only used 2 parameters and other features such as protein size might also have been used by Weng lab to categorize the proteins.[4] However, an 80% accuracy for a simple clustering method is significant.

After getting a map of the bound proteins to their true chain names, 1AHW was used as a sample for renaming and analyzing the docking results. I counted the number of members in each cluster: [{'Center 605': 43}, {'Center 365': 20}, {'Center 24': 77}, {'Center 466': 26}, {'Center 555': 10}, {'Center 316': 75}, {'Center 770': 39}, {'Center 993': 13}, {'Center 942': 25}, {'Center 68': 11}, {'Center 251': 11}, {'Center 432': 21}, {'Center 969': 30}, {'Center 987': 3}, {'Center 678': 38}, {'Center 216': 28}, {'Center 434': 8}, {'Center 607': 38}, {'Center 789': 8}, {'Center 370': 27}, {'Center 453': 10}, {'Center 155': 21}, {'Center 576': 34}, {'Center 650': 31}, {'Center 898': 8}, {'Center 16': 70}, {'Center 828': 26}, {'Center 816': 15}, {'Center 694': 31}, {'Center 785': 5}, {'Center 264': 10}, {'Center 889': 71}, {'Center 983': 15}, {'Center 335': 28}]. The ranges of the different types of potentials were also calculated from the ftfile (the main output of ClusPro) using relational tools for 1AHW predictions. The ranges for the 6 energy potentials across the 70,000 predictions were found to be: Van Der Waals – repulsive: 572.011, Van Der Waals – attractive: 2276, electrostatic potential: 0.7162, born potential: 3.518 and DARS potential: 1012.6 and the total energy potential: 280.41. Finally, potential homologs to 1AHW were also pulled from the Protein Data Bank (PDB) using similarity features. All of these were done using relational tools and coding as well as time of computation were improved.

In the second project, antibody-antigen complexes were chosen out of 230 proteins benchmark set of which there were 38. After running the evaluative program, DockQ, on all the top predictions of ClusPro for 33 of the 38 proteins, the evaluative scores were further analyzed.

The constraint outlined in equation A was enforced for every single prediction of each protein which means for about ~30 models for each protein which is solved 105 times comes to ~100,000 constraints. Basu et al., in their search for a single score that captured fnat, IRMSD and LRMSD, used tedious brute force method to find the scaling factors for iRMSD and LRMSD. The authors used "a grid search … for all pairs of d1 and d2 in the range of 0.5 to 10Å for d1, and 0.5 to 5Å for d2 in steps of 0.5."[3] Granted, they did not have a DockQ score already to compare it to. With Z3 solver, the tedious grid search was made more efficient and found that the iRMS scaling factor was ~1.4 and LRMS scaling factor was 8.66 which are almost exactly as the true results reported by Basu et al.[4] Lastly, the correlation coefficients between the four evaluative scores for assessing accuracy was calculated as well the corresponding p-values. As shown in Table 1, interface RMSD (iRMS) shows a perfect correlation with DockQ, which is the ultimate score. This result is intuitive as the interface where the two proteins interact is more important to predict correctly than the orientation of one of the proteins.

Table 1: P-values of correlation coefficients of the four evaluative scores that are used to assess accuracy of ClusPro's predictions

	FNAT	**iRMS**	**LRMS**	**DockQ**
FNAT	0	0.023	0.106	0.004
iRMS	0.024	0	0.0185	0
LRMS	0.121	0.018	0	0.006
DockQ	0.005	0	0.011	0

In addition to visualizing the input data set and parts of project 1, a third visual feature was added to the project with Chart.js. As pointed out in the introduction, ClusPro ranks its models based on energy first but then clusters the top 1000 and then ranks the clusters based on population size. The top models it retrieves are the geometric centers of those clusters.

Though, this methodology of picking predictions has been proven to be good, it is not perfect. In order to perfect that methodology or potentially develop a new one the interplay between cluster size, energy and accuracy needs to be understood. Therefore, a bubble chart was plotted using Chart.js with the energy ranking divided by 50 was used as radius, x-axis shows the relative cluster size (cluster rank) instead of absolute cluster size while y-axis represented dockQ score (accuracy of prediction). Note that smaller bubbles should, as per ClusPro's hypothesis, be more accurate. As seen, the trend is not exactly in line with that hypothesis which makes sense. With more data points entered, a better understanding of the relationships between the three features can be obtained.

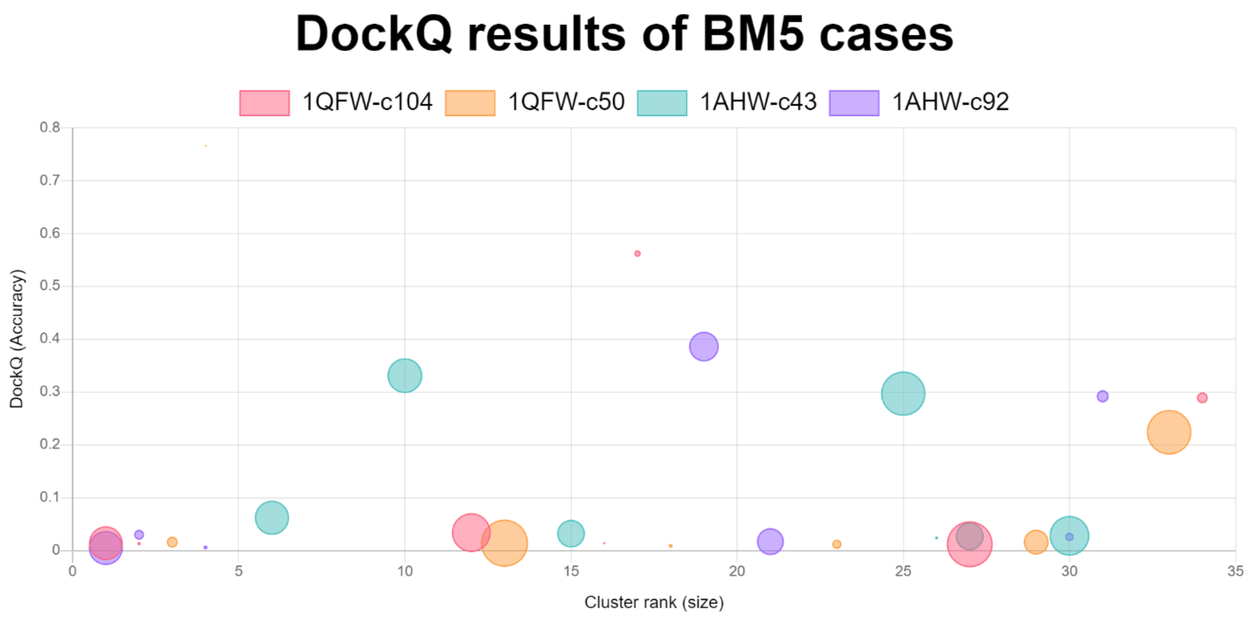
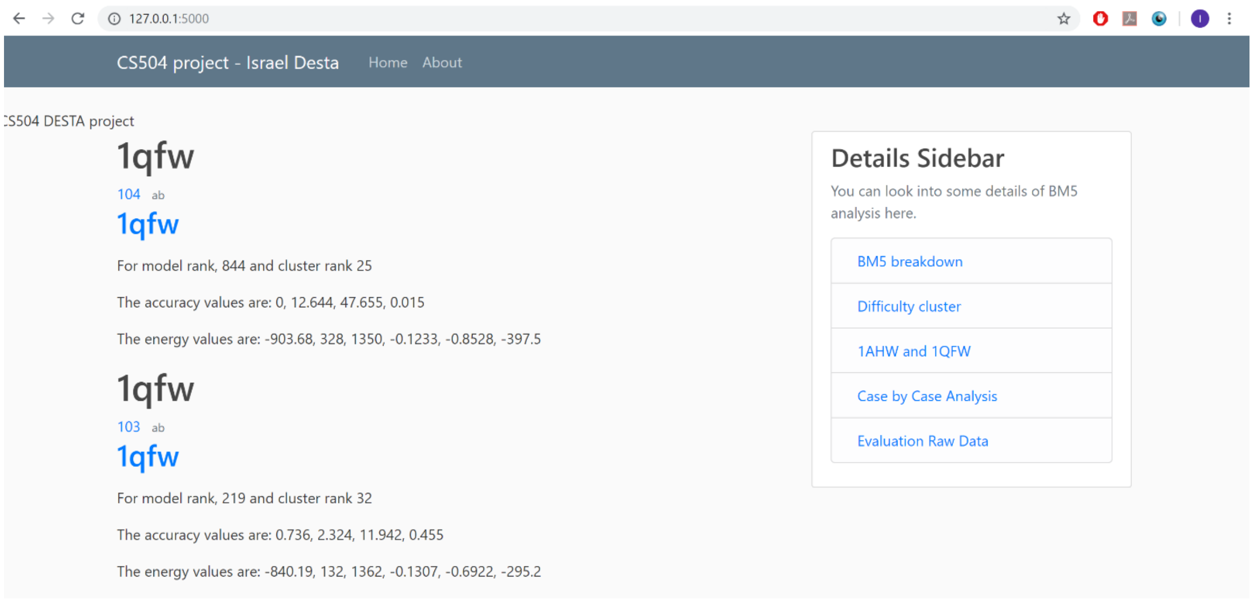


Figure 3: Bubble chart showing the interplay between cluster size, energy and accuracy.

Finally, the three visualizations, all done using Chart.js, are showcased in the last feature of this project which is a web application. The webApp allows users to switch between the different visualizations easily in addition to the interactive nature of the charts themselves. For figure 3, a web service with a RESTful web API was also attempted. A json file was uploaded to the server (also can be viewed by users), so that the figure 3 can use that json data. However, due to a bug that I was unable to fix, I have not been able to plot the data points. However, the code and layout is included in the submission.



1qfw

104 ab

1qfw

For model rank, 844 and cluster rank 25

The accuracy values are: 0, 12.644, 47.655, 0.015

The energy values are: -903.68, 328, 1350, -0.1233, -0.8528, -397.5

1qfw

103 ab

1qfw

For model rank, 219 and cluster rank 32

The accuracy values are: 0.736, 2.324, 11.942, 0.455

The energy values are: -840.19, 132, 1362, -0.1307, -0.6922, -295.2

Details Sidebar

You can look into some details of BM5 analysis here.

Figure 4: A screenshot of the web app that was written to allow users to switch between all the visuals built for this project.

Acknowledgements

I would like to thank Professor Andrei Lapets, Instructor Po-Yu Hsieh and TF Hao Chen for their support through out the semester.

Reference

[1] Schafer, Corey. "Coreymschafer - Overview". *Github*, 2019, https://github.com/CoreyMSchafer. Accessed 29 Apr 2019.

[2] "Chart.Js | Open Source HTML5 Charts For Your Website". *Chartjs.Org*, 2019, https://www.chartjs.org/. Accessed 29 Apr 2019.

[3] Vreven, T. M., et al. J. Mol. Biol. **2015**, 427, 3031-3041

[4] Basu, S. and Wallner, B. PLoS One. **2016**, 11, e0161879