Short Term Project Report
December 20, 2020

Computational affinity estimation of protein-protein complexes

Alexander Shumilov, IST

1 Project description

This project was originally initiated as thesis project, but due to circumstances thesis topic was changed and work on this topic was stopped. In this short term project initial developments were brought to some reasonable conclusion.
Project goal - create reliable model for prediction of protein-protein complexes affinity. To be more specific, the task is to solve regression task for particular energy functional dependent on distances between Coulomb matrices, that are generated based on normal and mutated protein. The correct solution gives a mechanism of differentiation of a mutated protein-protein complex with stable structure from non-stable one.

2 Preliminaries

2.1 Data collection

The data source for this project is Structural database of Kinetics and Energetics of Mutant Protein Interactions, or SKEMPI [1]. SKEMPI is a manually curated database of mutations in structurally characterised protein-protein interactions and the effect of those mutations on binding affinity and other parameters. Collected data include the PDB file, the chains of the interacting subunits, the mutation, the wildtype and mutant affinities, the reference, the names of the proteins, the temperature at which the experiment is performed, the experimental method used, notes on the entry and, when available, the association rate, dissociation rate, enthalpy and entropy.

2.2 Data preprocessing

The given raw data is not suitable for any computations and analysis yet. For example, PDB files for the interactions often contain multiple copies of the interacting proteins in the unit cell or other chains irrelevant to the interaction. Further, some PDB files contain features that are not readily parsed by some software, such as residue insertion codes or negative residue numbers. Apart from technical issues, there are problem with difference in protein description conditions. For most proteins such quantitative data as Gibbs energy is given for a different temperature factor, so recalculation is need. So main preprocessing steps are:

1. Preparation of PDB files for parsing
2. Calculation of Gibbs free energy of given proteins using temperature (298 K) and affinity data, calculation of its modification in case of mutated protein
3. For proteins with different temperature recalculation with 298 K initial conditions, marking proteins with pH different from 7.4
4. Collecting info about special cases of affinity constant value for further steps
5. Collecting general dataset statistics

2.3 Coulomb Matrix

The Coulomb matrix has recently been widely used as molecular descriptors in various ML models [4]. Given a molecule its Coulomb matrix defined by Equation 1, where Z_i is the atomic number of atom i, and R_i is its position in atomic units.

$$M_{ij}^{\text{Coulomb}} = \begin{cases} 0.5Z_i^{2.4} & \text{for } i = j \\ \frac{Z_i Z_j}{R_{ij}} & \text{for } i \neq j \end{cases} \tag{1}$$

Here, off-diagonal elements correspond to the Coulomb repulsion between atoms i and j, while diagonal elements encode a polynomial fit of atomic energies to nuclear charge.
The justification of usage of Coulomb Matrices is the certain distance measure that helps to translate this problem to programming field. This measure is a distance $d(M, M_i)$ between two matrices M, M_i - the Euclidean norm of molecules' diagonalized Coulomb matrices that is given in Equation 2, where ϵ stands for eigenvalue of a matrix.

$$d(M, M_i) = d(\epsilon, \epsilon^i) = \sqrt{\sum_j |\epsilon_j - \epsilon_j^i|} \tag{2}$$

Such form can used to compare mutated protein complexes with non-mutated ones, but in our problem the goal is to simultaneous process mutated and non-mutated protein to obtain slightly unusual rectangular Coulomb matrix - for every original protein we take each of corresponding mutants. So in this case diagonal term will look like $0.5(Z_i \cdot Z_i')^{1.4}$, where elements' charges are taken from different mutants of certain protein. Since the number of atoms in different mutants of one protein complex does not vary widely enough on the scale of the total number of atoms in a molecule, such matrices are very close to ordinary square matrices. Also important to know that in order to speed the computations instead of finding eigenvalues, singular values were found since SVD process tends to be much faster and error is relatively low.

2.4 Energy of protein-protein complex

Within suggested approach for further ML modeling, the energy of a molecule M is defined as sum over weighted Gaussians (Equation 3, [2]). As you can see this function is dependent on distance between Coulomb matrices for a given molecules.

$$E(M) = \sum_{i=1}^{N} \alpha_i \exp\left[-\frac{1}{2\sigma^2} d(M, M_i)^2\right] \tag{3}$$

From the perspective of machine learning, coefficients α_i are regression coefficients to be found and σ is length-scale parameter. So every i molecule contributes to the energy with corresponding regression coefficient α_i.
To determine unknown coefficients kernel ridge regression is suggested. This regularized model limits the norm of regression coefficients (Equation 4), thereby ensuring the transferability of the model to new compounds. For given length scale and regularization parameter λ, the explicit solution to the minimization problem.

$$\min_{\alpha} \sum_i (E(M_i) - E_i^{ref})^2 + \lambda \sum_i \alpha_i^2 \tag{4}$$

The E^{ref} mixes the Perdew–Burke-Ernzerhof (PBE) exchange energy and Hartree–Fock exchange energy in a set $3:1$ ratio, along with the full PBE correlation energy and could be computed using DFT theory.

3 Methods

3.1 Data processing

Most of preprocessing was done using specific python-compatible software Modeller. First thing to do is to generate mutants for wild type PDB files from SKEMPI database. Also notable that python package Dscribe could help to understand how to construct correct Coulomb matrix from scratch, because to make rectangular matrix for such task there are no special software and it must be done by hands.
So the workflow could be described by following points:

1. Studying the methods of processing PDB-format files and motivation of using modeller itself, working with basic scripts, first attempt to generate descriptor for certain protein
2. Implementation of walk through all original PDB files to collect any information we may need (Gibbs energy, e.t.c.)
3. ”Mutated” proteins generation via modeller. Was used one of modeller example scripts with some modifications mostly for stable work with large number of input PDB files. The script consequently takes an input PDB file from directory with SKEMPI database files and mutates a single residue with further position optimization.

The main issue however became the construction of Coulomb matrix itself. There are two proposed methods that were investigated and compared - more fast and presumably less precise method that, and more precise but significantly slower method. First is Dmitry's initial approach, were he computed Coulomb matrices considering not every protein's atom, but using average among all atoms of every residue (kind of centering). With such an assumptions results were quite good (discussed in next section). Second approach is to consider the influence of atoms, but only for neighbouring residues. This approach overall should be better, but processing time is extremely big (whole database was not process in 3 months of basically daily work of script), results are not that good. So balance between these 2 approaches should be found, which was could not be possibly done during work on this so to say side project.

3.2 Learning

The minimization problem from Equation 4 is a regression problem. The ML regression algorithm of choice for this task is widely used XGBoost. Hyperpaameters of XGBoost were tuned via GridSearch. To estimate the performance was used 5-fold cross-validation. Also there were different approaches to regression (Kernel Ridge, Extremely Randomized Tree regression) but they were proved to be insufficient for this task with comparison to XGBoost.

4 Results

First thing to check is distribution of distances (Figure 1) for some protein pairs (mutated/non-mutated) for comparison with [3] to understand that we are actually making matrices right. As you can see, the figures from experiment and referenced paper have close peaks of probability density, so the algorithm works well.

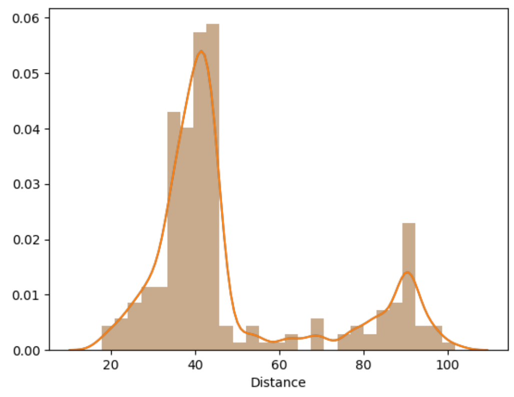

Figure 1: Distribution of distances for some protein pairs (mutated/non-mutated)

As for metrics of evaluation, MAE, RMSE and R-square correlation coefficients were chosen. MAE is a model evaluation metric often used with regression models, RMSE is standard deviation of the errors which occur when a prediction is made on a dataset and R-squared is a statistical measure of how close the data are to the fitted regression line. The errors and correlation coefficients for two approaches could be found in corresponding table below.

Approach	MAE	RMSE	R^2
"Fast"	5.16	8.71	0.68
"Precise"	9.21	13.83	0.66

Obtained errors are in good correspondence with reference paper [3]. "Good correspondence" here means the same order of errors, because despite the initial data is basically same, but the processing and ML approaches are different. Correlation coefficients in turn are not so high, but for this task result could be considered more than satisfactory. The difference in correlation coefficients is insignificant.

Corresponding visualization of algorithm accuracy could be seen in Figures 2 and 3. On the x-axis you can see referenced energies and on y-axis given the experimental outcome of the ML approach.

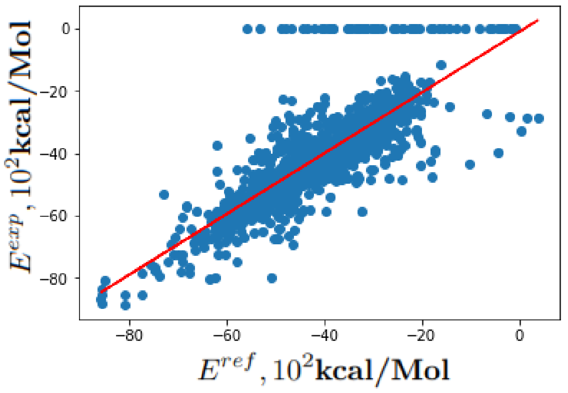

Figure 2: Correlation of DFT-PBE0 results (E^{ref}) with ML (cross-validated) based estimates (E^{est}) of atomization energies for faster Dmitry's approach

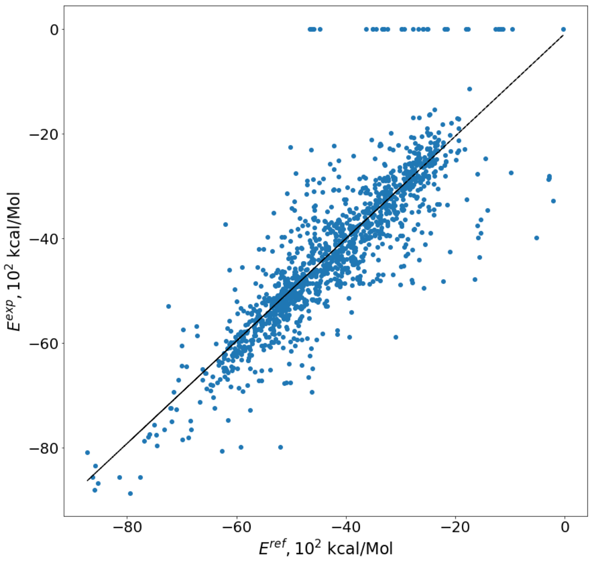

Figure 3: Correlation of DFT-PBE0 results (E^{ref}) with ML (cross-validated) based estimates (E^{est}) of atomization energies for slower but presumably more accurate approach

As you can see some of the experimental points for both approaches are invalid (zero energy). It indicates that some stage of work some files were not processed correctly. The difference between number of incorrectly processed files is explained by the nature of approaches - the slower approach is just more accurate by construction. Overall the performance looks reasonably good for both instances.

5 Conclusion

In this project was developed ML model for prediction of stability of mutated protein-protein complexes that base on description of electrostatic interactions using specific Coulomb matrices. From raw PDB files were generated possible mutants that became th basis of Coulomb matrices generation. In turn such matrices were used in further ML regression for experimental energy computation. Were tested two approaches for the files processing, Dmitry's fast and presumably less accurate and more through but more slow one. The best $R^2 = 0.68$ coefficient was obtained using fast approach, corresponding errors (MAE, RMSE) are lower as well for faster approach.

One of the most of problems currently are related of processing PDB files - some of them are not properly structured and thus processing scripts fail to obtain any viable information and throw errors. Another issue is time of processing scripts execution - proteins are relatively big, so there are matrices. What is notably there is the possibility using HPC methods to reduce it, but parallelization in python is kind of tricky. Since we are working with matrices and do not need an access to whole matrix in any given moment of time, there is pretty obvious way to go parallel - distribute sub-matrix "blocks" of initial matrices among given number of processes using so-called ghost columns, which contain matrix elements that another process may need for calculation. Of course Coulomb matrix composing itself fits parallelization concept ideally. For now attempts to run my scripts on Zhores (due to access to Zhores sandbox with 2 GPUs), but it was quite unsuccessful due to limited choice of python packages that can be loaded and overall memory restrictions (there is a lot of files to process). So if I'll be asked now what approach is better overall, I could have answered that the less sophisticated is better in this situation, since wasted time did not help to achieve much increase in accuracy.

References

[1] Justina Jankauskaitė et al. "SKEMPI 2.0: An updated benchmark of changes in protein-protein binding energy, kinetics and thermodynamics upon mutation". In: *Bioinformatics* 35 (July 2018). DOI: `10.1093/bioinformatics/bty635`.

[2] K. R. Muller et al. "An introduction to kernel-based learning algorithms". In: *IEEE Transactions on Neural Networks* 12.2 (2001), pp. 181–201. DOI: `10.1109/72.914517`.

[3] Matthias Rupp et al. "Fast and Accurate Modeling of Molecular Atomization Energies with Machine Learning". In: *Phys. Rev. Lett.* 108 (5 Jan. 2012), p. 058301. DOI: `10.1103/PhysRevLett.108.058301`. URL: `https://link.aps.org/doi/10.1103/PhysRevLett.108.058301`.

[4] Joshua Schrier. "Can One Hear the Shape of a Molecule (from its Coulomb Matrix Eigenvalues)?" In: *Journal of chemical information and modeling* 60.8 (Aug. 2020), 3804 "3811. ISSN: 1549-9596. DOI: `10.1021/acs.jcim.0c00631`. URL: `https://doi.org/10.1021/acs.jcim.0c00631`.