Predicting the Effect of Point Mutations On Protein Structural Stability

Roshanak Farhoodi
UMass Boston
100 W.T. Morissey Blvd.
Boston, Massachusetts 02125
roshanak.farhoodi001@umb.edu

Max Shelbourne
Western Washington University
516 High Street
Bellingham, WA 98225
shelbom@wwu.edu

Rebecca Hsieh
Western Washington University
516 High Street
Bellingham, WA 98225
hsiehr@wwu.edu

Nurit Haspel
UMass Boston
100 W.T. Morissey Blvd.
Boston, Massachusetts 02125
nurit.haspel@umb.edu

Brian Hutchinson†
Western Washington University
516 High Street
Bellingham, WA 98225
brian.hutchinson@wwu.edu

Filip Jagodzinski
Western Washington University
516 High Street
Bellingham, WA 98225
filip.jagodzinski@wwu.edu

ABSTRACT

Predicting how a point mutation alters a protein's stability can guide drug design initiatives which aim to counter the effects of serious diseases. Mutagenesis studies give insights about the effects of amino acid substitutions, but such wet-lab work is prohibitive due to the time and costs needed to assess the consequences of even a single mutation. Computational methods for predicting the effects of a mutation are available, with promising accuracy rates. In this work we study the utility of several machine learning methods and their ability to predict the effects of mutations. We *in silico* generate mutant protein structures, and compute several rigidity metrics for each of them. Our approach does not require costly calculations of energy functions that rely on atomic-level statistical mechanics and molecular energetics. Our metrics are features for support vector regression, random forest, and deep neural network methods. We validate the effects of our *in silico* mutations against experimental $\Delta\Delta G$ stability data. We attain Pearson Correlations upwards of 0.69.

CCS CONCEPTS

•**Computing methodologies** → *Machine learning algorithms;*
•**Applied computing** → *Bioinformatics; Molecular structural biology;*

KEYWORDS

Machine Learning; Protein Structure; Mutation; Support Vector Regression; Random Forest; Deep Neural Network; Rigidity Analysis

† Brian Hutchinson has a joint appointment with Pacific Northwest National Laboratory, 1100 Dexter Ave N., Seattle, WA 98109.

ACM-BCB'17, August 20-23, 2017, Boston, MA, USA.

978-1-4503-4722-8/17/08…$15.00
DOI: https://doi.org/10.1145/3107411.3116251

1 INTRODUCTION

The amino acid sequence of a protein determines its structure and as a result, its function. Even a single amino acid substitution can alter a protein's shape, which can be the cause of a debilitating disease. For example, mutations of α-galactosidase cause Fabry disease, a disorder that causes cardiac and kidney complications [14].

Wet-lab experiments can be used to engineer a protein with a specific mutation, and the mutant directly assessed to infer the effect of that amino acid substitution. The wild type and mutant proteins can be denatured to determine their relative unfolding rates, from which the free energy of unfolding ($\Delta\Delta G$) can be calculated; it is an indicator of whether a particular mutation is stabilizing or destabilizing, and to what degree. Existing experimental data about various mutations performed in physical proteins is available in the ProTherm database [23].

Unfortunately, conducting mutagenesis experiments on physical proteins is expensive and time consuming, and thus experimental data about the effects of mutations is limited. Therefore, computational methods can be helpful in estimating the effects of a mutation on a protein structure, and several computational methods have been developed in the past, with various degrees of success.

2 RELATED WORK

In this section we survey the existing experimental and computational work for predicting the effects of amino acid substitutions.

2.1 Experiments On Physical Proteins

Wet-lab experiments provide the gold standard for directly measuring the effect of mutations on a protein structure, measured by $\Delta\Delta G$ with respect to the wild type. Matthews *et al.* have generated many mutants of Lysozyme from the Bacteriophage T4 [2, 7, 12, 26–28]. They found that residues with high mobility or high solvent accessibility are much less susceptible to destabilizing substitutions. Although such studies provide precise, experimentally verified insights into the role of a residue based on its mutation, they are time consuming and cost prohibitive. Additionally, some mutations are so destabilizing that the mutant protein cannot be crystallized at all. Thus, only a small subset of all possible mutations can be studied explicitly.

2.2 Computational Approaches

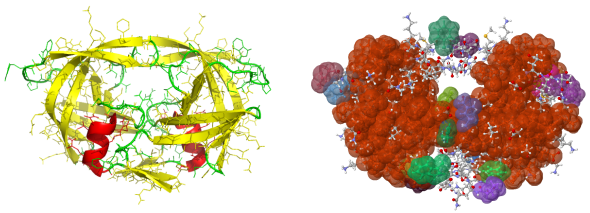
To complement and inform mutation studies performed on physical proteins, computational methods have been developed over the years. These methods strive to predict the effects of mutations. Several have high prediction and accuracy rates in the 70-80% range.

Several computational methods for assessing the effects of mutations fix the atoms in the backbone of a protein and proceed to search for the best side-chain conformation, while others utilize rotamer side chain libraries to adjust a structure in response to an amino acid substitution [11, 19, 31]. Other approaches [24] rely on heuristic energy measurements to predict the stability of proteins in which side chains are perturbed. Yet another approach [15] estimates the folding free energy changes upon mutations using database-derived potentials. Prevost [32] used Molecular Dynamics simulations to study the effect of mutating Barnase, and concluded that the major contributions to the free energy difference arose from non-bonded interactions. Thus, progress has been made in predicting the effects of mutations on protein stability. However, many such methods rely on computationally intensive energy calculations and are therefore time intensive.

2.3 Combinatorial, Rigidity Based Methods

A first generation of rigidity-based mutation analysis tools are available, but the extent of the types of *in silico* mutations that they can perform are limited. Rigidity Analysis [16] is a combinatorial technique for identifying the flexible regions of biomolecules. Figure 1 depicts the cartoon and rigidity analysis results of PDB file 1hvr of HIV-1 protease. Rigidity analysis, which identifies rigid clusters of atoms, is distinguished from most other methods because it is fast. It does not rely on homologous protein data, nor costly all-atom energy calculations.

In our previous work we used rigidity analysis to probe how a mutation **to glycine** destabilizes a protein's structure. We compared the rigidity properties of the wild type structure to the rigidity properties of a mutant that we generated *in silico* using KINARI-Mutagen [18]. On input of a Protein Data Bank (PDB) structure file, KINARI-Mutagen identifies hydrogen bonds and hydrophobic interactions. The stabilizing interactions involving the atoms of the side chain being mutated to Glycine are removed from the protein's model. This is equivalent to computationally mutating a specific residue to Glycine, the smallest amino acid which has no side chain atoms that form stabilizing bonds.



(a) Cartoon Representation **(b) Rigidity Analysis**

Figure 1: Rigidity analysis of PDB file 1hvr identifies sets of atoms belonging to rigid clusters. The largest rigid cluster is shown orange, which spans both halves of the protein. The larger rigid clusters are (cluster size : count) 11:6, 12:5, 15:2, 16:2, 19:2, 23:1 and 1371:1.

The effect of a mutation on the protein's structural stability can be correlated with its effect on a protein's rigidity. In our previous work [5, 17] we measured the effect of the mutation by recording the change in the size of the Largest Rigid Cluster (LRC) of the mutant versus the wild type (WT, non-mutated protein). The rationale was that the LRC is an indicator of the protein's rigidity or flexibility. Predictions were validated against experimentally derived $\Delta\Delta G$ unfolding measurements from the ProTherm [23] database. A negative $\Delta\Delta$G value for a mutant structure reveals that the mutant is less stable than the wild type, and thus the amino acid substitution is destabilizing.

2.4 Machine Learning Based Approaches

Machine learning (ML) is a branch of artificial intelligence involving algorithms that allow programs to classify, group, and learn from data. The regression problem proceeds via the following steps: a) represent a set of known data points as a set of *feature vectors* labeled by the corresponding output value, b) train a model that best maps inputs to the correct output, c) use the model to make predictions on a set of new (e.g. held out) data points. In Section 5 we detail each of the Support Vector Regression, Random Forest, and Deep Neural Network methods we used.

Machine learning and statistical methods have been developed to help predict the effects of mutations and to infer which residues are critical. Cheng *et al.* [10] used Support Vector Machines to predict with 84% accuracy the sign of the stability change for a protein due to a single-site mutation. Also, data of amino acid replacements that are tolerated within families of homologous proteins have been used to devise stability scores for predicting the effect of residue substitutions [35], which has been extended and implemented into an online web server [36].

In our previous work [17], we used an SVM-based model that combines rigidity analysis and evolutionary conservation, in addition to amino acid type and solvent accessible surface area, to a dataset of proteins with experimentally known critical residues. We achieved over 77% accuracy in predicting the sign of the change of stability for a single point mutation to Glycine and Alanine.

Brender, *et al* [9], have developed a scoring function that reasons about protein-protein interfaces. They used sequence- and residue-level energy potentials in conjunction with a Random Forest (RF) approach to achieve a Pearson correlation coefficient of approximately 80% between predicted and observed binding free-energy changes upon mutations. Jia, *et al* [20], have employed a variety of Machine Learning Tools to generate several models based on thermostability data for assessing the effects of single point mutations. They used 798 mutants from 51 different protein structures for which there is $\Delta\Delta G$ data, and attained accuracy rates ranging from 78-85% among SVM, RF, NBC, KNN, ANN, and PLS approaches, with the Random Forest Approach having the highest accuracy. Li, *et al* [25], developed a model based on the Random Forest algorithm for predicting thermostability changes due to amino acid substitutions. In their approach they relied on 41 features, and achieved accuracies of 79.9%, 78.2%, and 78.7% for single, double, and multiple point mutation.

3 MOTIVATION

As discussed above, existing experimental methods still provide only partial information about the effects of mutations. Computational methods can complement this information, but many existing methods are time consuming, or alternatively, their accuracy could be improved. There is a need for fast and reliable methods that can efficiently analyze the effect of a mutation of an amino acid on the structure of a protein. As already discussed, machine learning based methods have been used in the past by us and other researchers, and they are a promising avenue to explore further.

In this work, we present fast and efficient machine learning and graph theory based methods for predicting the effect of a mutation on a protein structure. Through rigidity analysis, support vector regression, random forests and deep neural networks, we predict the effect of a mutation on the $\Delta\Delta G$ of a protein. We validated our results by using experimental data from the ProTherm database. Our approach achieves strong performance in predicting the effect of a single point mutation on a protein structure, while being fast enough to run in a few minutes and sometimes seconds.

Several aspects of our work distinguish it from others. **Firstly, none of our features in use by our machine learning models require calculating energetics of various biophysical phenomena**. The calculation of our metrics does not require costly calculations based on statistical mechanics nor molecular energetics. Our features are strictly structure-based, which is a purposeful design decision to enable near real-time run-times. **Secondly, the number of data points that we use is far more than most others have used**. With 2,072 mutations for which we have experimentally derived data from ProTherm, our dataset far surpasses in size most others, many of which have fewer than 1,000. This dataset is far greater than we used in our previous work due to our recent expanded capabilities of generating mutations *in silico* [4]. **Lastly, the majority of our features in use by our models are derived from quick calculations detailing the rigidity properties of mutant, wild type pairs of protein structures**. With the exception of our past proof-of-concept work, nobody else has used rigidity metrics on a large scale to assess their use in enhancing models for predicting the effects of mutations.

4 DATA PREPARATION

Here we describe the source of our data, including how we processed the ProTherm $\Delta\Delta G$ values, how we generated *in silico* mutants, and the split of the data into training, development and testing sets for building our machine learning models. We enumerate our features and explain how each is normalized.

4.1 ProTherm Data, *in silico* mutants

We downloaded the entire ProTherm plain-text database, and identified 2,072 entries for single mutations with $\Delta\Delta G$ values. See Table 1 for a summary of the proteins and their mutations that we retained.

We used our in-house software, Protein Mutation High Throughput (ProMuteHT) [4], for quickly generating protein mutants *in silico*, with the mutations for which we had ProTherm stability data. That software is composed of several parts, including custom scripts and algorithms, integrated with off-the-shelf open access tools and libraries. It generates large-to-small (LTS) as well as small-to-large (STL) amino acid substitutions. For an LTS mutation ProMuteHT removes atoms from a PDB file to simulate a substitution. For example, mutating Leucine to Alanine involves removing from a protein structure file the CG, CD1 and CD2 atoms from the Leucine being mutated. When a small residue is mutated to a larger one, the STL module relies on the freely availably SCWRL 4.0 [22] software that makes predictions about a side chain's orientation. To account for the steric clashes that might arise due to replacing a small residue with a larger one, we used the NAMD [30] software to perform 500 energy minimization steps, requiring approximately 5 seconds.

Table 1: Summary of the data obtained from ProTherm.

Parsed Numerical Results	
Unique Proteins	44
Total Mutations	2,072
Mutated Residues that are Hydrophobic	892
Mutated Residues that are Aromatic	161
Mutated Residues that are Polar	432
Mutated Residues that are Charged	654
Mutated Residues with 0-30% SASA	932
Mutated Residues with 30-50% SASA	532
Mutated Residues with 50+ SASA% SASA	608

The rigidity of each mutant and its wild type were analyzed using the publicly available rigidity software by Fox *et. al* [13]. The rigidity output data is of the form *rigid cluster size : count*, which offers the distribution of clusters and their sizes that were identified. We used the rigidity data for each mutant, wild type pair to calculate 6 different rigidity metrics. See [3] for a complete discussion explaining the motivation and utility of these metrics at inferring the effects of mutations.

4.2 Features and Data Split

From the ProTherm data and our rigidity calculations, we derived the following 60 features:

- WT SASA: how exposed to the surface a residue is.
- WT Secondary Structure: four features indicating whether the mutation took place in a sheet, coil, turn or helix.
- Temperature and pH at which the experiment for calculating $\Delta\Delta G$ was performed.
- Rigidity Distance (RD): one of *lm, sig1, sig2, sig3, sig4, sig5*. See [3] for a full explanation.
- WT Rigid Cluster Fraction: 24 features giving the fraction of atoms in the WT that belong to rigid clusters of size 2, 3, . . . , 20, 21-30, 31-50, 51-100, 101-1000 and 1001+, respectively.
- Mutation Rigid Cluster Fraction: 24 features giving the fraction of atoms in the mutation belonging to the same set of bins as above.
- Residue type: four binary features indicating whether the residue is Charged (D, E, K, R), Polar (N, Q, S, T), Aromatic (F, H, W, Y), or Hydrophobic (A,C,G,I,L,M,P,V).

These features along with their normalization scheme and range are summarized in Table 2.

Table 2: Feature summary.

Feat. #	Name	Norm.	Range
1	WT SASA	None	$[0, 1]$
2-5	WT Secondary Structure	None	$\{0, 1\}$
6	Temperature	0-1	$[0, 1]$
7	Potential of Hydrogen (pH)	pH	$\approx [-1, 1]$
8	Rigidity Distance (RD)	Standard	$\mathbb{R}$
9-32	WT Rigid Cluster Frac.	None	$[0, 1]$
33-56	Residue Rigid Cluster Frac.	None	$[0, 1]$
57-60	Residue Type	None	$\{0, 1\}$

Normalization "0-1" refers to the mapping

$$\frac{x_i - \min(x)}{\max(x) - \min(x)} \tag{1}$$

while "pH" normalization refers to a slight variant that maps pH of 0 to -1, 7 to 0 and 14 to 1:

$$\frac{x_i - 7}{7}. \tag{2}$$

Standardization refers to normalizing the feature to be zero-mean and unit-variance:

$$\frac{x_i - \mu}{\sigma} \tag{3}$$

We estimated the feature mean (μ) and standard deviation (σ) on the training set, and used these same values to transform the development and test sets. Our $\Delta\Delta G$ labels were also standardized prior to training using the training set statistics. The $\Delta\Delta G$ values fall in the range of [-5.02, 4.55] after standardization.

Our 2,072 data points were randomly split into a training set (1,438 data points), a development set (324 data points) and a test set (310 data points), under the constraint that all instances of each WT appear in a single dataset. **This constraint was used so that we can assess how well our methods generalize to new protein wild types, and means that our reported results are more pessimistic than they would be under an unconstrained random split**.

5 METHODS : SVR, RF, DNN

In this section, we provide more details about three popular machine learning methods, Random Forest, Support Vector Regression and Deep Neural Networks, that we used for predicting $\Delta\Delta G$. That is, we solve a regression problem: our objective is building a function that minimizes the difference between calculated (predicted) and actual observed target values for all data points in the training set.

5.1 SVR

Support Vector Machines are supervised learning models that are widely used for solving classification and regression problems. In case of Support Vector Regression (SVR), the aim is to minimize the generalization error bound in order to achieve strong generalization [6]. The generalization error bound is the combination of the training error and a regularization term that controls the complexity of hypothesis space [34]. SVR works based on generating a regression function in a high dimensional feature space where the input data are implicitly mapped using a *kernel function*. The kernel function can be linear or nonlinear (polynomial, sigmoid or radial basis).

The SVR model that we used in this work for $\Delta\Delta G$ prediction were implemented using Scikit-learn [29], which is a free machine learning library for the Python programming language. Scikit-learn offers a variety of easy to use clustering, classification and regression algorithms implementations.

We used grid search to tune the parameters of the model using our training and development sets. For the SVR model we tuned three main parameters; the penalty parameter of the error term (regularization constant) C, the kernel function and the kernel coefficient γ. Using the development set, we evaluated the prediction accuracy using values chosen from range of 0.1 to 1000 for C, examined RBF, linear and sigmoid as the kernel functions and changed γ values from 0.0001 to 1. The lowest error was achieved with C equal to 150, when the RBF kernel was used and γ was set to 0.1.

5.2 Random Forest

Random forests (RFs) work by utilizing the power of decision trees. A decision tree infers decision rules from the training data to carry out a regression or classification task. The decision rules are generated based on the value of a feature that yields the best split of the training data using a metric such as information gain or Gini impurity index. A random forest is an ensemble learning method that fits a number of decision trees on multiple random sub-samples of the training set and then use averaging to boost the accuracy of the prediction and control overfitting [8]. Each tree uses a sample size the same as the original training set. The method allows these samples to be the same as the original training set (Replicate) or to be drawn with replacement (Bagging).

The random forest model was also implemented using Scikit-learn. To tune the model parameters, we examined the prediction error while changing the number of estimators (trees in the forest) from 10 to 1000. The accuracy of the model did not show improvements when beyond 150 estimators were used. For resampling method, we used bagging (Bootstrap Aggregation), where each tree in the forest was trained using random subsamples from the training set chosen with replacement. Bagging showed over 30 percent improvement in accuracy compared to Replicate. For the rest of parameters such as maximum depth of the tree and minimum number of samples required to split an internal node we used Scikit-learn default values, but plan to include these parameters in our future tuning experiments.

5.3 DNN

A Deep Neural Network (DNN) is a supervised machine learning model in which the input undergoes multiple "hidden" layers of non-linear transformation before a prediction is made. This generalizes the standard, shallow neural network which has only a single hidden layer. Predictions, y, in our DNN are made according to the following:

$$y = W_{(L)}^T h_{(L)} + b_{(L)} \tag{4}$$

$$h_{(i)} = g\left(W_{(i-1)}^T h_{(i-1)} + b_{(i-1)}\right) \text{ for } i = 1, \ldots, L \tag{5}$$

where $h_{(0)}$ denotes the input x, and our model parameters are matrices $W_{(0)}, \ldots, W_{(L)}$ and vectors $b_{(0)}, \ldots, b_{(L)}$. The hidden activation

function is denoted g; we explored two options: $g(z) = \tanh(z)$ and $g(z) = \text{ReLU}(z) = \max(0, z)$. Our model parameters are learned using first order optimization algorithms to minimize training set mean squared error (MSE).

We implemented our DNN using TensorFlow [1], an open-source machine learning toolkit. Our hyperparameters were tuned using a combination of random search and the Spearmint [33] tool. Spearmint is a Bayesian hyperparameter optimization tool, incrementally exploring the hyperparameter space with the objective of maximizing expected improvement.

We tuned several hyperparameters on the developed set: number of hidden layers (1 - 4), number of units per layer (10 - 100), learning rate (0.0001 - 0.5), weight initialization range (0.0001 - 0.1), activation function (tanh, ReLU), and the optimizer used for backpropagation (stochastic minibatch gradient descent, adam [21]).

6 RESULTS

The SVR, RF and DNN models were trained initially using the training set and the hyperparameters were tuned to optimize development set performance as described in Section 5. After the optimal parameters were identified, in a final round of training for SVR and RF, the samples in training set and development set were combined and used as the training set. For DNN, whose results were highly dependent on the initial random weight initialization, we omitted this step of combination and retraining, and simply used the best model trained on the training set alone. We repeated our experiments using the features mentioned in Section 4 for six configurations where in each configuration we included one RD measure for training. The prediction accuracy of the models on the test set was then evaluated using two metrics. We calculated Root Mean Square Error (RMSE) and the Pearson correlation coefficient between predicted and expected (actual) $\Delta\Delta G$ values. The prediction accuracy of the three models for each RD measure is presented in Table 3. As shown in the table, the RF model consistently outperformed the SVR and DNN models by having lower RMSE and higher correlation coefficients. The DNN performance typically falls in-between the other two. This relative ordering is fairly consistent regardless of the RD value used as a feature to train the models. Averaged over the six experiments (last rows of Table 3), the RF gives the best performance and the SVR gives the worst. While the RF RMSE and correlation variation over different RDs is rather small, RF generated the least RMSE (0.820) and highest correlation (0.694) when *sig2* was used in the feature set. With SVR and DNN *sig1* and lm showed the best performance among the RDs, respectively.

Table 3: Prediction accuracy (RMSE = Root Mean Square Error, C = Pearson Correlation)

RD	Accuracy Measure	SVR	RF	DNN
lm	RMSE	0.961	0.839	0.865
	C	0.534	0.673	0.647
*sig*1	RMSE	0.945	0.822	0.963
	C	0.555	0.691	0.528
*sig*2	RMSE	0.960	0.820	0.957
	C	0.534	0.694	0.537
*sig*3	RMSE	0.959	0.822	0.946
	C	0.539	0.692	0.551
*sig*4	RMSE	0.986	0.827	0.931
	C	0.500	0.687	0.571
*sig*5	RMSE	0.966	0.821	0.942
	C	0.524	0.694	0.557
Avg.	RMSE	0.963	0.825	0.934
	C	0.531	0.689	0.565

7 DISCUSSION

To assess the utility of our three models in predicting the values of $\Delta\Delta G$ due to point mutations, we compared the Pearson Correlation Coefficients of our Random Forest model (our highest scoring average) against equivalent coefficients for 12 other approaches that we found in the literature [20]. Our Pearson Correlation Coefficient value of 0.689 would rank our RF approach 9^{th} of 12, with Prethemut, ProMaya, and ELASPIC having attained higher correlation coefficient values of 0.72, 0.74, and 0.77, respectively. Understandably, any such comparison must be taken with caution, for example due to different data set sizes, different cross validation approaches, as well as data preprocessing. And although for this work we focused on a regression model rather than attempting a binary classification of the data, it is not uncommon in the literature for binary classification models to excluded neutral (0±0.5 $\Delta\Delta G$ kCal/mol) mutants. Any such similar pre-processing, which we did not do, might be employed by other methods and models attempting regression analyses, which might ultimately affect a ranking of different approaches.

Another important point worth reiterating is that none of our features were attained via direct calculations of energetic terms arising from changes in a protein's confirmation due to a mutation. Although we previously indicated that doing so was a conscious effort on our part aiming to minimize costly energy calculations, indeed excluding energy terms might be related to a possible limitation of our approach. Namely, an amino acid substitution on a protein structure might induce a destabilizing or stabilizing effect due to reasons that are not structure based, which our method would not be able to reason about because our features are all purely structural in nature.

8 CONCLUSIONS

We developed and present several machine learning based methods to predict the effects of mutations on the stability of a protein. In particular, our method predicts the change to the free energy of unfolding upon mutation ($\Delta\Delta G$), using a combination of graph based rigidity analysis and features like solvent accessible surface area (SASA), temperature, pH, and the type of mutated amino acid. We trained and tested our methods on an extensive dataset taken from the ProTherm database, which contains experimental information about point mutations. We show that our algorithm, especially the Random Forest (RF) based predictor, can predict the $\Delta\Delta G$ with high accuracy.

Our next steps involve developing methods to assess the effects of multiple point mutations. While many methods predict the effect of a single mutation on a protein structure, only very few of

them assess how multiple amino acid substitutions affect a protein's structure and stability. Additionally, since our method is very fast and efficient, requiring as little as a few seconds to conduct a computational experiment, we are developing a server which will allow users to conduct hypothesis testing about the effects of mutations. We envision that our server will run in near real-time, and thus permit high-throughput studies, enabling screening a large number of amino acid substitutions and their effect on a protein's stability.

9 CONTRIBUTIONS

Roshanak conducted the RF and SVR analysis, Max performed the DNN analysis, and Rebecca preprocessed and aggregated the ProTherm data. Brian, Nurit, and Filip supervised the work. All authors contributed to the analysis of the results and the writing of the manuscript.

ACKNOWLEDGEMENTS

The work is supported in part by NSF grant CCF-1421871 (NH). BH and MS gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X GPU used for this research.

REFERENCES

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D.Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. (2015). http://tensorflow.org/ Software available from tensorflow.org.

[2] T. Alber, S. Dao-pin, J.A. Wozniak, S.P. Cook, , and B.W. Matthews. 1987. Contributions of hydrogen bonds of Thr 157 to the thermodynamic stability of phage T4 lysozyme. *Nature* 330 (1987), 41–46.

[3] E. Andersson, R. Hsieh, H. Szeto, R. Farhoodi N. Haspel, and F. Jagodzinski. 2016. Assessing how multiple mutations affect protein stability using rigid cluster size distributions. In *Computational Advances in Bio and Medical Sciences (ICCABS), 2016 IEEE 6th International Conference on*. IEEE, 1–6.

[4] E. Andersson and F. Jagodzinski. 2017. ProMuteHT : A High Throughput Compute Pipeline for Generating Protein Mutants in silico. In *CSBW (Computational Structural Bioinformatics Workshop), in proc. of ACM-BCB (ACM International conference on Bioinformatics and Computational Biology)*.

[5] B. Akbal-Delibas, F. Jagodzinski, and N. Haspel. 2013. A Conservation and Rigidity Based Method for Detecting Critical Protein Residues. *BMC Structural Biology* 13(Suppl 1) (2013), S6.

[6] D. Basak, S. Pal, and D.C. Patranabis. 2007. Support vector regression. *Neural Information Processing-Letters and Reviews* 11, 10 (2007), 203–224.

[7] J.A. Bell, W.J. Becktel, U. Sauer, W.A. Baase, , and B.W. Matthews. 1992. Dissection of helix capping in T4 lysozyme by structural and thermodynamic analysis of six amino acid substitutions at Thr 59. *Biochemistry* 31 (1992), 3590–3596. Issue 14.

[8] L. Breiman. 2001. Random forests. *Machine Learning* 45, 1 (2001), 5–32.

[9] J. Brender and Y. Zhang. 2015. Predicting the effect of mutations on protein-protein binding interactions through structure-based interface profiles. *PLoS Computational Biology* 11, 10 (2015), e1004494.

[10] J. Cheng, A. Randall, and P. Baldi. 2006. Prediction of Protein Stability Changes for Single-Site Mutations Using Support Vector Machines. *PROTEINS: Structure, Function, and Bioinformatics* 62 (2006), 1125–1132.

[11] R.L. Jr. Dunbrack and M. Karplus. 1994. Conformational analysis of the backbone-dependent rotamer preferences of protein sidechains. *Nature Structural Biology* 1 (1994), 334–340. Issue 5.

[12] A.E. Eriksson, W.A. Baase, X.J. Zhang, D.W. Heinz, E.P. Baldwin, and B.W. Matthews. 1992. Response of a protein structure to cavity-creating mutations and its relation to the hydrophobic effect. *Science* 255 (1992), 178–183.

[13] N. Fox, F. Jagodzinski, and I. Streinu. 2012. Kinari-lib: a C++ library for pebble game rigidity analysis of mechanical models. *Minisymposium on Publicly Available Geometric/Topological Software* (2012).

[14] S.C. Garman and D.N. Garboczi. 2002. Structural basis of Fabry disease. *Molecular Genetics and Metabolism* 77, 1 (2002), 3–11.

[15] D. Gilis and M. Rooman. 1997. Predicting protein stability changes upon mutation usings database derived potentials : Solvent accessiblity determines the importances of local versus non-local interactions along the sequence. *Journal Molecular Biology* 272 (1997), 276–290. Issue 2.

[16] D.J. Jacobs, A.J. Rader, M.F. Thorpe, and L.A. Kuhn. 2001. Protein Flexibility Predictions Using Graph Theory. *Proteins* 44 (2001), 150–165.

[17] F. Jagodzinski, B. Akbal-Delibas, and N. Haspel. 2013. An Evolutionary Conservation & Rigidity Analysis Machine Learning Approach for Detecting Critical Protein Residues. In *CSBW (Computational Structural Bioinformatics Workshop), in proc. of ACM-BCB (ACM International conference on Bioinformatics and Computational Biology)*. 780–786.

[18] F. Jagodzinski, J. Hardy, and I. Streinu. 2012. Using rigidity analysis to probe mutation-induced structural changes in proteins. *Journal of Bioinformatics and Computational Biology* 10 (2012). Issue 3.

[19] J. Janin and S. Wodak. 1978. Conformation of amino acid side-chains in proteins. *Journal of Molecular Biology* 125 (1978), 357–386. Issue 3.

[20] L. Jia, R. Yarlagadda, and C.C. Reed. 2015. Structure Based Thermostability Prediction Models for Protein Single Point Mutations with Machine Learning Tools. *PloS One* 10, 9 (2015), e0138022.

[21] D. Kingma and J. Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[22] G. Krivov, M. Shapovalov, and R.L. Dunbrack. 2009. Improved prediction of protein side-chain conformations with SCWRL4. *Proteins: Structure, Function, and Bioinformatics* 77, 4 (2009), 778–795.

[23] M.D. Kumar, K.A. Bava, M.M. Gromiha, P. Prabakaran, K. Kitajima, H. Uedaira, and A. Sarai. 2005. Protherm and Pronit : Thermodynamic databases for proteins and protein-nucleic acid interactions. *Nucleic Acids Research* 34 (2005), D204–D206.

[24] C. Lee and M. Levitt. 1991. Accurate prediction of the stability and activity effects of site-directed mutagenesis on a protein core. *Nature* 352 (1991), 448–451.

[25] Y. Li and J. Fang. 2012. PROTS-RF: a robust model for predicting mutation-induced protein stability changes. *PloS one* 7, 10 (2012), e47247.

[26] M. Matsumura, W.J. Becktel, and B.W. Matthews. 1988. Hydrophobic stabilization in T4 lysozyme determined directly by multiple substitutions of Ile 3. *Nature* 334 (1988), 406–410.

[27] B. Mooers, W.A. Baase, J.W. Wray, and B.W. Matthews. 2009. Contributions of all 20 amino acids at site 96 to the stability and structure of T4 lysozyme. *Protein Science* 18 (2009), 871–880. Issue 5.

[28] H. Nicholson, E. Soderlind, D.E. Tronrud, and B.W. Matthews. 1989. Contributions of left-handed helical residues to the structure and stability of bacteriophage T4 lysozyme. *Journal of Molecular Biology* 210 (1989), 181–193. Issue 1.

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[30] J.C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R.D. Skeel, L. Kale, and K. Schulten. 2005. Scalable molecular dynamics with NAMD. *Journal of Computational Chemistry* 26, 16 (2005), 1781–1802.

[31] J.W. Ponder and F.M. Richards. 1987. Tertiary templates for proteins: Use of packing criteria in the enumeration of allowed sequences for different structural classes. *Journal of Molecular Biology* 193 (1987), 775–791. Issue 4.

[32] M. Prevost, S.J. Wodak, B. Tidor, and M. Karplus. 1991. Contribution of the hydrophobic effect to protein stability: analysis based on simulations of the Ile-96-Ala mutation in barnase. *Proceedings of the National Academy of Sciences* 88 (1991), 10880–10884. Issue 23.

[33] J. Snoek, H. Larochelle, and R.P. Adams. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in neural information processing systems*. 2951–2959.

[34] M. Song, C.M. Breneman, J. Bi, N. Sukumar, K.P. Bennett, S. Cramer, and N. Tugcu. 2002. Prediction of protein retention times in anion-exchange chromatography systems using support vector regression. *Journal of Chemical Information and computer sciences* 42, 6 (2002), 1347–1357.

[35] C.M. Topham, N. Srinivasan, and T. Blundell. 1997. Prediction of the stability of protein mutants based on structural environment-dependent amino acid substitutions and propensity tables. *Protein Engineering* 10, 1 (1997), 7–21.

[36] C.L. Worth, R. Preissner, and L. Blundell. 2011. SDM-a server for predicting effects of mutations on protein stability and malfunction. *Nucleic Acids Research* 39 (2011), W215–W222. Issue Web Server Issue.