Predicting Protein Melting Temperature Based On Amino Acid Sequence

Ruben Krueger
Department of Computer Science
Stanford University
ruben1@stanford.edu

Marco Mora-Mendoza
Department of Computer Science
Stanford University
marcom3@stanford.edu

Josh Wolff
Department of Computer Science
Department of Bioengineering
Stanford University
jw1@stanford.edu

Abstract

The structure and function of a protein is based on its sequence of amino acids. At a high temperature, a protein's structure may change such that the protein does not have the function it had at a lower temperature ("unfolded"). These temperatures, referred to as melting temperatures, vary from protein to protein. In this paper, we present three machine learning models for the prediction of protein melting temperature based on their amino acid sequence. The code and data can be found at https://github.com/Ruben-Krueger/cs_221_final_project.

1 Introduction

1.1 Background

Proteins are one of the four fundamental classes of macromolecules, with a variety of biological functions. These functions arise from their structure, and this structure is based on the protein's sequence of amino acids. A protein is functional when it is "folded" and non-functional when "unfolded." Proteins can become unfolded when the temperature rises, and protein thermostability is based on sequence (Ku et.al, 2009). The melting temperature of a protein is the temperature at which half of the protein molecules are unfolded. Deriving the melting temperature of proteins is of interest, particularly in drug development (Gorania et. al 2010). Current techniques, including differential scanning calorimetry, circular dichroism, and Fourier transform infrared spectroscopy, are expensive (Gorania et. al 2010). A computational way to estimate this value would be instrumental for numerous applications.

1.2 Literature Review

Most machine learning research on proteins has been applied to predicting the 3D structure of the protein. AlphaFold (Evans et al. 2018) in particular has accelerated the field's progress in this task.

Many research groups have approached the prediction of protein melting temperature as a classification problem. Wu et al. (2008) used a decision tree to classify proteins into three thermostability classes, achieving an accuracy of 84%. Ku et al. (2009) built a protein melting temperature classifier that differentiates between proteins made from hyperthermophilic microorganisms and those made from mesophilic microorganisms. Additionally, Lin et al. (2010) used a support vector machine to classify thermophillic proteins.

Pucci and Rooman (2010) used a dataset of 45 proteins to create a predictor of protein stability curves using as input the three-dimensional protein structure and the host organism.

Gorania et al. (2010) compiled 230 proteins from existing studies to make their dataset and used an Artificial Neural Network and an Adaptive Network-Fuzzy Inference System to predict a protein's melting temperature.

Our paper deviates from the available literature as we seek to predict a value for the melting temperature, as opposed to a temperature class. Furthermore, we have not found literature in which the authors used a Latent Dirichlet Allocation model in protein characteristic prediction or protein engineering. We find our application to be novel, and potentially useful for uncovering explicit motifs in the protein structure from sequence.

1.3 Task Definition

Given a protein's sequence of amino acids, what is its melting temperature?

2 Infrastructure

2.1 Data Collection

Protein PDB code	$T_m^{\exp}$ (°C)	pH	Protein Name	Res. ($\mathring{A}$)
1aqh	43.7	7.2	α-amylase	2.00
1ppi	65.6	7.2	α-amylase	2.20
1jae	65.9	7.2	α-amylase	1.65
1smd	70.3	7.2	α-amylase	1.60
1am7	52.3	7.0	Lysozyme	2.30
2lzm	64.8	6.5	Lysozyme	1.70
1lz1	64.9	2.7	Lysozyme	1.50

Figure 1. A table of protein melting temperatures used in our dataset. We present this as an example of the data we are scraping. This exact table can be found in the paper by Pucci, et. al. (2015) cited below.

While there have been previous groups that have experimentally characterized melting temperatures of proteins for their research, there is not one single, central database that matches a protein with its melting temperature. For example, the Protein Data Bank (PDB) does not include this value with their information about proteins. The Prokaryotic Growth Temperature database (PGTdb) was an attempt from a previous research group to establish such a database, but it is currently offline (Huang et al. 2004). Attempts to contact the researchers of this project were unsuccessful. Thus, for our data set, we collected a list of 235 proteins with their name, PDB code, and melting temperature from various academic papers on the subject.

2.2 Data Synthesis

From our dataset of real proteins, we derive artificial "mutants": proteins with a single amino acid substitution (e.g., "A" $\rightarrow$ "T") to augment our collected data of real proteins. We aim to improve the robustness of our models. We generated 34,633 mutants.

The melting temperature of the artificial mutants is concordant with the melting temperature of the real protein from which it was derived. We assume that one amino acid substitution does not cause a significant change in protein structure. This assumption is flawed; notably, addition of amino acids such as cysteine or proline can have a large impact on protein folding, and thus the actual melting temperature may be quite different. However, we accept this error as a consequence of the difficulty in obtaining training data.

Montanucci, et. al. predicted thermostability changes to the protein resulting from multiple mutations in the *nucleic acid* sequence. From their experimental dataset description, one notes that even up to four mutations in amino acid sequence changes the temperature by less than 10 degrees Celsius. For example, five amino acid changes to the protein FAOX changes the melting temperature from 37 degrees Celsius to 45 degress Celsius. Nevertheless, the flaw in our assumption is also noted, where, for example, one mutation to CbADH changes the melting temperature by 11.5 degrees Celsius.

3 Approach

3.1 Baseline

Our first models are baseline models. These are simplistic models which give lower bounds on the performance of our other models.

We implemented three baselines. The first baseline simply considered the length of the amino acid. Using this as the only feature for linear regression, we report a value of $r^2 = -0.091$. Indeed, the value reported by r^2 can be negative when the squared distances from a horizontal line through the data exceed that of the squared distances from the given model to the actual data (GraphPad, FAQ). We then tested a single feature that considers the counts of polar, nonpolar, and "special" amino acids, and we report a value of $r^2 = 0.005$. Finally, we had the most success by considering groupings of amino acids. First, we considered the frequency of individual letters in the sequence (e.g. an amino acid such as 'Q') and observed $r^2 = -0.272$. Then, we considered the frequency of groupings of two (e.g. 'QR', a bigram) and report $r^2 = 0.055$. Finally, groupings of three letters results in $r^2 = 0.017$. Our baseline demonstrated that the length of a protein does not correlate well to the melting temperature, which is consistent with previous research (Franzosa et al, 2010). Other simple features also did not show a strong correlation as measured by the r^2 loss. The strongest correlation was polar, nonpolar, and "special" amino acid frequencies. We have chosen to pursue three models that can extract features we otherwise would not have predicted or that can extrapolate from the features we describe.

3.2 Oracle

An oracle is a model which provides an upperbound to the performance of any model on the task. The performance of the oracle will be a benchmark for the other models to achieve. Often, an oracle will "cheat" in some fashion (solve the task with additional information) to achieve performance better than the baseline. For our task, the problem is intractable for a human observer without laboratory equipment. Thus, implementing an oracle would be solving the original task.

3.3 Neural Network

3.3.1 Description

A neural network is a model consisting of three layers: an input layer, hidden layer(s), and an output layer. The input layer is where the features of the data are inserted. The hidden layer(s) are used to extract features. Each layer consists of "nodes" which emulate the functioning of biological neural neurons.

3.3.2 Implementation

Overview. We used the Keras Sequential model for our neural network. Our neural network consisted of an input layer, three hidden layers, and one output layer. Each hidden layer used the Relu activation function and the Adam optimizer.

The features of this model were the counts of the amino acids; the number of hydrophilic amino acids; the number of hydrophobic amino acids; the number of glutamic acid and lysine amino acids divided by the number of glutamine and histine amino acids $(E + K)/(Q + H)$. The last ratio in particular has been found by Wu et al. to be a ratio that is very indicative of a protein's melting temperature.

Each column in the data set was transformed to a Normal distribution with mean of 0 and standard deviation of 1. Additionally, each hidden layer had the $L2$ regularization constant added. This was done to prevent node weights from getting too large and preventing overfitting.

3.3.3 Evaluation Metrics

We evaluate the loss of our neural network with mean squared error. That is, given a prediction $\hat{y}$ and actual value y, we compute the error for our model over the testing data as

$$Loss(y_{test}, y_{pred}) = \sum_{y_i \in T_{test}} (\hat{y}_i - y_i)^2.$$

We evaluate the error of the model, given a protein p with predicted melting temperature y and true melting temperature $\hat{y}$, as

$$Error(p, y, \hat{y}) = \frac{y - \hat{y}}{y}.$$

Over 40 epochs, we trained the neural network with a batch size of 2000 proteins. Figure 2 summarizes the loss of the model over the training set.

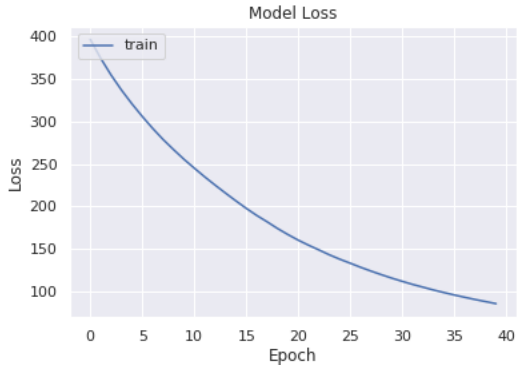

Figure 2. Neural network loss over the training set.

Now with a trained model, we tested the model with our testing set. Figure 3, for each protein in the testing set, plots the actual melting temperature and the corresponding predicted value. The model, using the above metric, had a mean squared error loss of 5334.7 over the testing set.

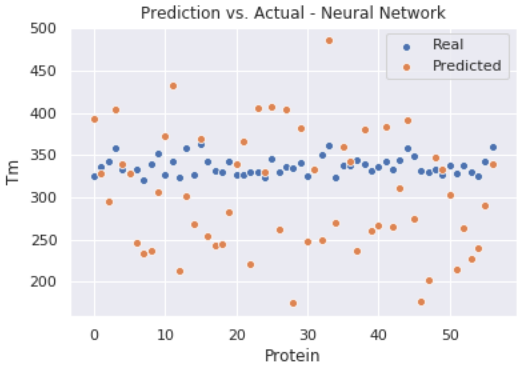

Figure 3. Predicted melting temperature values versus actual.

From this, we can discern that the model has very little predictive power.

3.3.4 Error Analysis

Figure 4. The amino acid composition of proteins for which the model had an error less than 5%.

Figure 5. The amino acid composition of proteins for which the model had an error greater than 5%.

Figures 4 and 5 plot the amino acid compositions of proteins which the model had more than 5% and less than 5% error. Visually and quantitatively, these two graphs are very similar. We can infer that the model predicts poorly regardless of the amino acid composition of the protein. From this, we can discern that it does not seem that the model performs poorly on proteins with some particular subset of amino acid composition.

3.4 Case Study

We now consider two proteins from the testing set. The first, 1CSE, had less than 5% error from our model. The second, 9RNT, had more than 5% error.

1CSE has a sequence which contains 37 polar and 33 nonpolar amino acids. 9RNT has a sequence which contains 64 polar and 40 nonpolar amino acids. Thus, the model performed worse on a protein that had more polar amino acids. This is most likely due to the fact that additional polar amino acids can cause large changes in protein structure, thus affecting melting temperature in a nonlinear fashion (Wu et al. 2009).

3.5 Regression Tree

3.5.1 Description

Regression trees are a type of decision tree where the output can take on continuous values. Each level of the tree divides the data according to a specific feature until a threshold is reached or

the data cannot be divided any further. The order of the features used for each level depends on the minimization of an error function, prioritizing the features that provide the best performance in the determination of the output.

3.5.2 Implementation

Overview. To train our tree, we used python's scikit-learn decision tree regressor library. The features used are extracted from our training set. At each level of the tree, the algorithm splits the data according to a particular feature. We then use this tree to make predictions on our test set. The return value for each input represents the predicted melting temperature of the protein sequence.

The features for this model were the same as the ones used in our neural network approach. This includes counts of the amino acids; n-grams of different lengths; the number of hydrophilic and hydrophobic amino acids; and the number of glutamic acid and lysine divided by the number of glutamine and histine aminoacids (E+K)/(Q+H).

Training. We constructed two versions of our regressive tree. For the first version, we used only 235 real proteins with their actual melting temperature. For the second, we used the 34,633 synthetic datapoints that we generated according to the procedure described above (See "Data Synthesis").

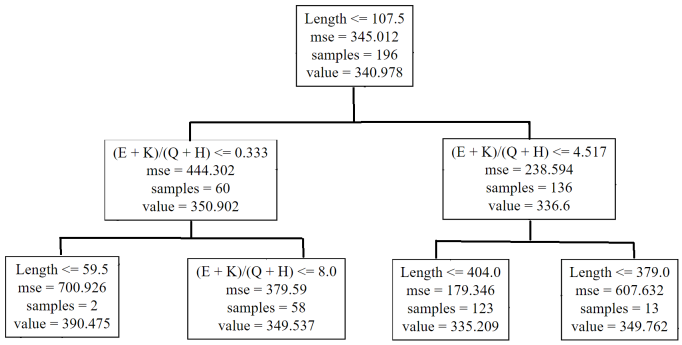

Figure 6. A subtree of our trained regressive tree.

3.5.3 Evaluation Metrics

We evaluate the accuracy of our regressive tree algorithm by comparing the mean squared error of the output over our testing set and the actual values. We use the `accuracy_score` function of python's scikit-learn library. If the entire set of predicted outputs matches exactly the true set of labels, the value for the accuracy is 1.0. Our algorithm trained on real data returns an accuracy of 0.0612 over our testing set, whereas the one trained in synthetic data returns a value of 0.035 over its testing set. The following two graphs show the result of applying the algorithm to each dataset.

3.5.4 Error Analysis

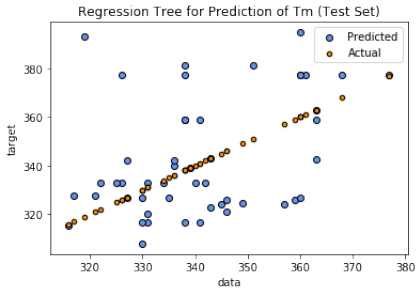

Figure 7. Results of regression tree trained with real data.

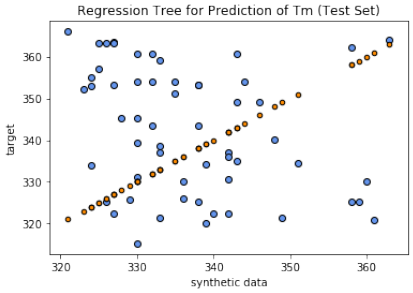

Figure 8. Results of regression tree trained with synthetic data.

Surprisingly, the tree trained on synthetic data performed at a much lower level than the one based on real datapoints. We suspect that this might be due to the algorithm overfitting to the synthetic data, since the synthetic datapoints are made to be very similar to their original counterparts.

3.5.5 Case Study

- Input: See synthetic protein #12 in test_synthetic_data_mutated.csv
- True Output: 324.15
- Predicted Output: 343.15

In this instance, the predicted value was grossly overestimated by the regression tree. In terms of relevant features, protein #12 has a length of 414 aminoacids, its ratio (E+K)/(Q+H) is 1.25, 195 of its aminoacids are polar, 165 nonpolar, 65 are labeled as special, and it had the largest amount of 'E' aminoacids, with a count of 33. As mentioned previously, a large amount of polar amino acids can cause large changes in protein structure and was most likely the factor of such a large discrepancy in predicted and actual melting temperature (Wu et al. 2009).

3.6 Latent Dirichlet Allocation Model

3.6.1 Description

Latent Dirichlet allocation (LDA) is a method to extract topics from text. It returns groups of words, or topics, to explain similar data (Ng, et. al. 2003). We first note that our input is fundamentally text - a string of letters. We propose that there are certain topics, or patterns of amino acid sequences, that appear frequently and characterize a certain range of melting temperatures. We consider that, for example, a particular motif in the protein structure might not even have a name, but might influence the melting temperature of the protein to some extent. Thus, we have implemented LDA to extract these topics from our protein dataset.

3.6.2 Implementation

Overview. We extract topics from a training set. We then compare those topics to those extracted from the test set. Then, we compare the topics from a test datapoint to those from our training set. We use this comparison to determine the range of melting temperature. We output this range as a final result.

Preparation. We separate our training data and our test data. We define the interval at which to divide the entire range of our training data (e.g. every 15K over the range of 300K to 400K). This determines the intervals over which we will produce topics (e.g. 300-344K). We combine all the proteins in a single interval into one long string with spaces in between the individual amino acids.

Output. We output a range as described in the subsection that follows, and we set an upper limit for the length of the range. This is necessary because if, for example, the range we output was 300-400K, then we could achieve 100% accuracy, although this would be entirely useless.

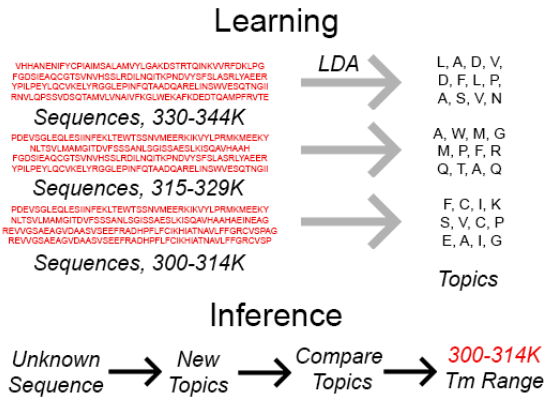

Figure 9. A graphic that helps explain how an LDA applies to our topic.

We use the LDA implementation from the `gensim` Python package.

3.6.3 Evaluation Metrics

Definitions. We define a "`WIN`" as a correct prediction such that the true T_m is within the range predicted. A "`LOSS`" is an incorrect prediction such that the true T_m is not within the range predicted. A "`NONE LOSS`" is where the model outputs nothing because none of the topics extracted from the test datapoint match any of those extracted from training. This is beneficial because we can directly point to our limited data as a reason for our inability to classify certain proteins.

Case Study. Here we report the results of one run of the LDA algorithm. Our pruned dataset was 229 proteins. We removed 6 outliers because these can result in inaccurate ranges when there is not enough data for the given range (e.g. one protein in the 375-389K range). We trained on 184 proteins and tested the algorithm on 45 proteins ($\approx \frac{1}{5}$ of all of the data). The intervals for which we produced topics were the following: 300-314, 315-329, 330-344, 345-359, 360-374, and 375-389. Our proteins vary in range from 312.95 to 368.05. Our ranges were of length 15 (e.g. 330K to 344K). We extracted 15 topics per interval with 4 "words" per topic. Our algorithm correctly predicted the range for 11 of the proteins, incorrectly predicted the range for 12 of the proteins, and returned a "`NONE LOSS`" for 22 of the proteins. Thus, our **overall accuracy** is 24.4% for this particular run; however, if we do not include those for which there was not enough data (i.e. the `NONE LOSS`), then we had a **prediction accuracy** of 47.8%. Because the LDA is a probabilistic algorithm, our results vary by run, even with the same split of test and training data. This was a particularly good run.

3.6.4 Error Analysis

For our error analysis, we pruned a few more proteins and proceeded to classify our data into the following intervals: 315-329, 330-344, 345-359, and 360-374. Here, we analyze our model's accuracy when predicting protein melting temperature for these intervals.

Synthetic Data versus Real Data Performance Because the LDA is a probabilistic algorithm, we ran 20 trials of our LDA using the same test set and training set. We did this twice, once for the real data and once for the synthetic data. We now report two confusion matrices for our classification results.

		Predicted				
		315-329	330-344	345-359	360-374	None
Actual	315-329	0.37	0.22	0.23	0.18	0.52
	330-344	0.42	0.29	0.19	0.10	0.59
	345-359	0.33	0.39	0.16	0.13	0.50
	360-374	0.52	0.32	0.16	0.00	0.42

Figure 10. Confusion matrix resulting from 20 trials of the LDA trained on real data and tested on unseen real data.

Range	Frequency in Test Set
315-329	0.26
330-344	0.56
345-359	0.12
360-374	0.05

Table 1: Frequency of proteins in ranges of selected test set

		Predicted				
		315-329	330-344	345-359	360-374	None
Actual	315-329	0.35	0.25	0.20	0.20	0.83
	330-344	0.26	0.25	0.23	0.26	0.78
	345-359	0.21	0.27	0.24	0.27	0.76
	360-374	0.07	0.40	0.33	0.20	0.75

Figure 11. Confusion matrix resulting from 20 trials of the LDA trained on synthetic data and tested on unseen real data.

Training on synthetic data resulted in a prediction accuracy of 0.272 whereas the training on the real data resulted in a prediction accuracy of 0.282, a negligible difference given the standard deviations.

We note that training on the synthetic data offered a smoothing effect on our results. For synthetic data, the minimum prediction accuracy of one of the runs was 0.00 and the highest was 0.60, with a standard deviation of 0.16. However, when training on real data, the minimum prediction accuracy for one of the runs was 0.17 and the maximum was 0.41, with a standard deviation of 0.07. Thus, the synthetic data likely exacerbated some of our issues pertaining to over-fitting for the training data.

Lastly, the `NONE LOSS`, where our algorithm cannot predict a range with any accuracy because it has never see the input topics before, was much larger for the synthetic data than it was for the real data (see the rightmost columns of Figure 10 and Figure 11). It is possible that incorrect topics were learned for these ranges as a result of the mutations, and thus when presented with real topics in the test set, the algorithm was unable to compare these to the mutated topics. Furthermore, it is also possible that the algorithm was so overfit to our synthetic training data that it generalized poorly compared to training on real data.

Bias Towards Smaller Ranges From Table 1, we see that our data has a large bias towards smaller ranges. Of the four ranges, more than half of the data is in one particular range while only 5% is within the highest range. This indicates that out of the 169 proteins we trained on, less than 10 were in the "360-374" range. Thus, it is obvious why our model performed so poorly for the "360-374" range.

Word Ordering The order of the letters in the array for the topic is important because the most probable word for the given topic is first in the array. However, if we were only to assert that two topics matched if they presented with the same ordering, then we would dramatically increase our `NONE LOSS` because several proteins would likely not have matching topics. This could make our model more accurate, but it would only make the model more useful if we had more data to compensate for the stricter requirement.

Case Study on a Misclassification

1. True Output: 315-329
2. Predicted Output: 360-374
3. Matching Topic: ['V', 'L', 'A', 'G']

The LDA misclassified this protein as the highest range when, in fact, it came from the lowest range. First, it matched this topic only because of our relaxation on word ordering. The learned topic from the "360-374" range that it matched with was ['A', 'G', 'V', 'L']. Second, this particular topic was only the third most probable topic out of 10 learned. Finally, it came quite close to matching with the true range, which contained the following two similar topics: ['L', 'A', 'V', 'T'] and ['L', 'D', 'A', 'V'], which were first and third most probable for the true range (315-329), respectively.

4 Discussions and Future Work

Here, we consider what future explorations would improve our findings.

More Data Primarily, we need more data, but as computer scientists and not experimentalists, we will consider the approaches of our algorithms.

Better Methods for Nuances We will likely need better methods for more nuanced problems. A simple neural network, even with a large amount of data, will likely incorrectly predict a protein's melting temperature if the protein has an obscure length, composition, and resulting structure.

Learning the Ranges The ranges we denoted for our Latent Dirichlet Allocation (LDA) model were largely arbitrary. However, in protein structure, there perhaps are motifs, which are patterns of protein structure, that result in a similar melting temperature. Perhaps demarcating the intervals around these landmark motifs will improve the accuracy of the LDA. On this note, learning the intervals might prove to be the best approach.

Improved Synthetic Data Our current synthetic dataset was likely too uniform and too large. With the synthetic dataset, the LDA achieved approximately the same accuracy, but had a much larger standard deviation. Our models would likely benefit if we applied a Gaussian distribution to the lengths of our mutated proteins rather than maintaining the same lengths for all proteins; randomly selected mutants to remove from the dataset; and had a heuristic that estimates the temperature change resulting from a mutation.

5 Conclusion

Predicting protein melting temperature is, put simply, a difficult problem. Predicting a protein's T_m explicitly from sequence requires the prediction of a number of variables that grows exponentially with protein length. Our models simplify the problem to a text-input and number-output task, and achieve some results in doing so.

External Factors There are several external factors that influence the denaturation of a protein. Salinity and pH are two examples. Each of the melting temperatures of the proteins in our dataset were measured at a given pH, some at 5.5 and some at 7.0, for example. However, the melting temperature of a protein is different at 5.5 than 7.0. Therefore, in order to build an accurate model, it is likely necessary to normalize for these experimental conditions.

Regression versus Classification We first attempted to tackle the problem as a regression problem, using a regression tree and a neural network. However, after running into a data roadblock, we decided to relax the problem slightly by classifying proteins into temperature ranges rather than report specific temperatures. For this, we used a Latent Dirichlet Allocation Model.

A New Approach Lack of data forced us to get creative. Our novel use of an LDA to classify protein melting temperature is a unique way of approaching this difficult problem. By extracting explicit topics from text, we hope that further research into using an LDA for this problem can reveal if these topics correspond to structural motifs. We have not found literature on attempts to do this.

Issues Our models suffered from the typical struggles faced with little data. They do not generalize nearly as well as they could, and they as a result, often offer similar predictions for inputs that have wildly different outputs. The limiting factor is definitely data, and it is likely more productive if more data collection is completed prior to development of the underlying algorithms.

Acknowledgments

We would like to thank the course staff of CS 221, and in particular Shervine Amidi, our project mentor, for his support and guidance.

References

[1] Evans, et. al. *De novo structure prediction with deep-learning based scoring.* In Thirteenth Critical Assessment of Techniques for Protein Structure Prediction (Abstracts) 1-4 December 2018.

[2] Franzosa, Lyangh, and Yu Xia. *Structural Correlates of Protein Melting Temperature.* Experimental Standard Conditions of Enzyme Characterizations, 2010.

[3] Gorania, M., H. Seker, and P. I. Haris. *Predicting a Proteins Melting Temperature from Its Amino Acid Sequence* 2010 Annual International Conference of the IEEE Engineering in Medicine and Biology, 2010. doi:10.1109/iembs.2010.5626421.3.

[4] GraphPad FAQ. Authors Unknown. https://www.graphpad.com/support/faqid/711/

[5] Ku, Tienhsiung, Peiyu Lu, Chenhsiung Chan, Tsusheng Wang, Szuming Lai, Pingchiang Lyu, andNaiwan Hsiao. *Predicting Melting Temperature Directly from Protein Sequences.* Computational Biology and Chemistry 33, no. 6 (2009): 445-50. doi:10.1016/j.compbiolchem.2009.10.002.4.

[6] Lin, Hao, and Wei Chen. *Prediction of Thermophilic Proteins Using Feature Selection Technique.* Journal of Microbiological Methods 84, no. 1 (2011): 67-70. doi:10.1016/j.mimet.2010.10.013.

[7] Montanucci, Fariselli, Martelli, and Rita Casadio. *Predicting protein thermostability changes from sequence upon multiple mutations.* Bioinformatics 24, no. 13 (2008): i190-i195. doi.org/10.1093/bioinformatics/btn166.

[8] Ng, Blei, and Michael Jordan. *Latent Dirichlet Allocation.* Journal of Machine Learning Researchvol 3,4-5. 993-1022. (2003): 993-1022. doi:10.1162/jmlr.2003.3.4-5.9935.

[9] Pucci, Kwasigroch, and Marianne Rooman. *SCooP: an accurate and fast predictor of protein stability curves as a function of temperature.* Bioinformatics, 33. No. 21 (2017): 3415-3422. https://doi.org/10.1093/bioinformatics/btx417

[10] Pucci, Fabrizio, and Marianne Rooman. *Towards an accurate prediction of the thermal stability ofhomologous proteins.* Journal of Biomolecular Structure and Dynamics. vol 33,5. (2015): 1132-1142.doi:10.1080/07391102.2015.10736316.

[11] Rees, D C, and A D Robertson. *Some thermodynamic implications for the thermostability of pro-teins.* Protein science : a publication of the Protein Society vol. 10,6 (2001): 1187-94. doi:10.1110/ps.1801017.

[12] Wu, Li-Cheng, Jian-Xin, Lee, Hsien-Da, Huang, Baw-Juine, Liu, and Jorng-Tzong Horng. *An Expert System to Predict Protein Thermostability Using Decision Tree.* Expert Systems with Applications 36, no. 5 (2009): 9007-014. doi:10.1016/j.eswa.2008.12.020.