Latent space Encoding of proteins and genomes for binding Affinity Prediction (LEAP)

Andrew Wu, Noah Faro, Andrew Hennes

May 2021

1 Abstract

DNA-protein cross interactions are at the heart of regulation in genomics, biological machines such as the ribosome and spliceosome, and the origin of phase separated cellular structures. In these Protein-DNA interactions, binding affinity is often the key thermodynamic regulator. To estimate these affinities, empirical methods, while most accurate, are also highly costly both in time and material. A fast, universal, and accurate approach for computational protein DNA affinity prediction could greatly accelerate protein biochemistry. Previous works have developed models using hand-designed features (e.g., hydrophobicity and molar mass) to predict binding affinities with moderate accuracy, taking on the order of 5 minutes per prediction[Yang and Deng 2020]. We have shown that by learning a low dimensional latent embedding of an amino acid and DNA sequence, we can train a promising regression model to predict protein DNA affinity in milliseconds.

2 Background

Protein nucleic acid interactions are ubiquitous throughout biological systems as living things must read, regulate, and compute on DNA, form functional ribo-protein complexes such as the ribosome and splicosome, and organize the genome at the intra and inter chromosomal level. While much work has been devoted towards empirically and computationally deducing DNA-binding proteins the process remains effort and material intensive [Hu, Ma, and Wang 2019]. Thus, a model robust to protein variation, thermodynamically accurate, and capable of de novo generating protein sequences would complement preexisting computational and empirical techniques for binding affinity determination. Learned relations between paired protein and nucleic acid sequences to binding free energies could accelerate transcription factor binding site identification and permit rapid, high throughput thermodynamic characterization of DNA protein complexes.

Representation learning is the ability to learn low-dimensional representations that encapsulate the same information as the original input. The ability to learn representations on data by training networks to reconstruct the input data from a latent space is the basis of auto encoders, which have been well explored in a variety of tasks [Naderi, Soleimani, and Matwin 2020]. Variational Autoencoders additionally require the distribution of input samples mapped to the latent space to be Gaussian, so that the latent space follows a regularized, easy to work with distribution. Previous efforts with VAE have allowed, among other applications, the generation of realistic faces and other objects with some representation dimensions human interpret-able to allow for rational creation of examples [Hou et al. 2019].

In analogy to previous efforts with small drug molecules, we aim to extend the VAE architecture to encode both proteins and DNA [Jin, Barzilay, and Jaakkola 2018]. This task is aided by the chemical redundancy of amino acids, which enables a concise encoding of complex chemical information at the level of amino acid identity. Thus, by training a VAE on sequence-level information of diverse, biologically representative protein and DNA sequences, we aim to learn a low dimensional latent space for proteins and DNA, which we then use for machine learning tasks such as predicting the free energy of binding. In combination with these learned latent representation of genomic and protein sequences, we propose training a model to characterize a genomic site's propensity to bind an arbitrary protein of interest through a learned relationship between sequence and binding free energies.

3 Results and Discussion

3.1 Data

In training our models, we used three datasets. The first two were simply compilations of sequences: one for proteins and one for DNA, used to train the input Variational AutoEncoders. The protein sequence data consisted of 90,000 protein sequences downloaded from Ensembl. These proteins were chosen from chickens, humans, mice, and zebrafish. The DNA sequence dataset was chosen similarly, consisting of 110,000 DNA sequences corresponding to the 500 nucleotides upstream of gene coding genomic regions of humans, chickens, mice, and zebrafish. Taking these large protein and

DNA sequences as input, the VAEs in the architecture were trained to reconstruct these sequences from a lower dimensional latent space.

The third dataset consisted of compilations of Protein-DNA complex binding affinities. This dataset was sourced from ProNIT, a previously-compiled dataset containing the Gibbs Free Energy of binding for various Protein-DNA complexes [Sarai et al. 2002]. In total, this dataset consisted of 11,600 Protein-DNA complex pairs, complete with temperature information, binding affinity, stoichiometry, and p_h. This dataset was used as training and validation data for the neural network for predicting binding affinity from the VAE's encoded latent space. Then, the PreDBA dataset, which compiled 200 more Protein-DNA complexes, was used as an additional test dataset for comparison of accuracy against contemporary models.

3.2 Methods

Our architecture takes advantage of variational autoencoders to produce latent representations of Proteins and DNA. In this way, we aim to reduce the impact of the smaller amount of data available, and potentially in future work allow for data generation: producing proteins with desired binding affinities for certain DNA strands.

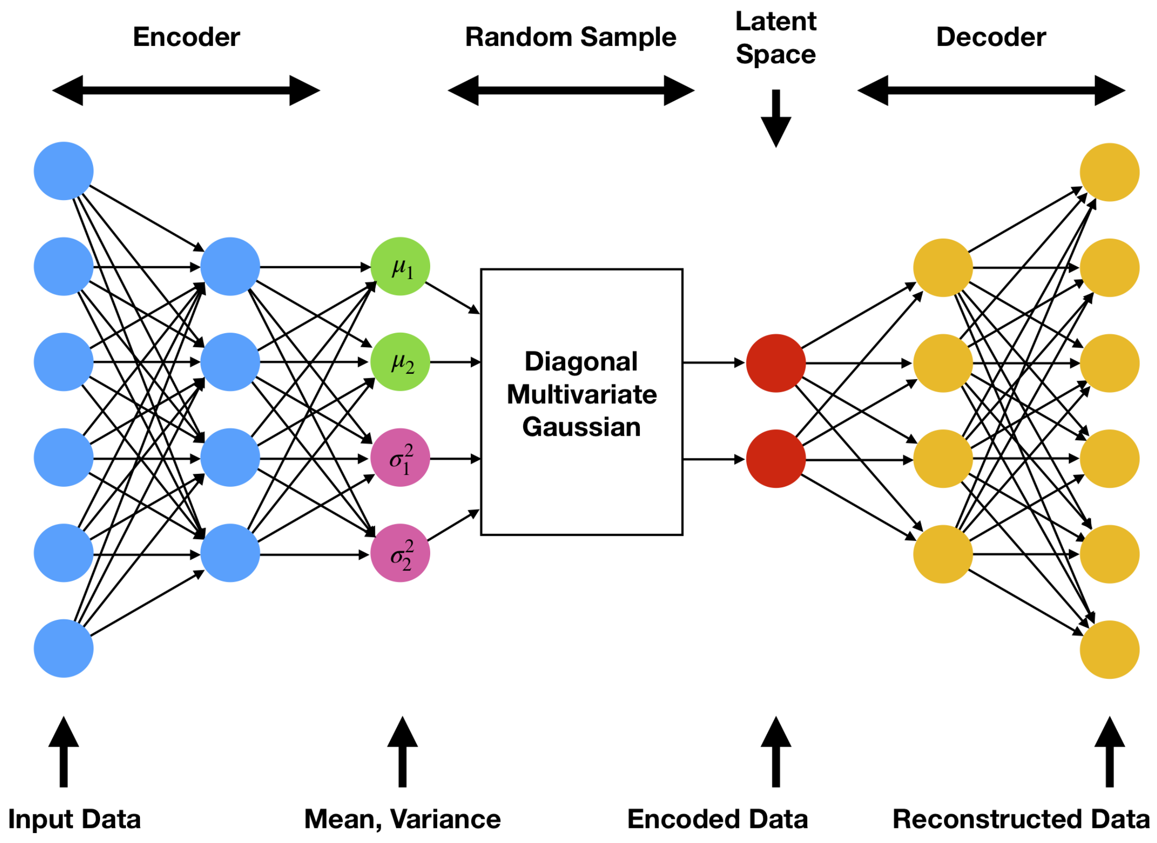


Figure 1: Variational AutoEncoder Architecture by Flores n.d.

3.2.1 Variational AutoEncoders

VAEs learn a mapping from (or "encodes") a high dimensional space such as an image, sequence, or voxel field into a lower dimensional latent space that captures key aspects of variation within a class of images. The reconstruction of the original object from the latent space representation is trained to penalize object reconstruction loss such that the encoding captures the structure and variability of the training set. In the space of polymers such as proteins and DNA, the VAE may learn attributes of sequences that reflect their structure, properties, or key chemical information. As examples, latent encodings in amino acid sequences have been shown to place positive and negative charged amino acids near each other [Hawkins-Hooker et al. 2021] and latent encodings in base pair sequences have been able to reproduce sequences that have organ-specific properties [Way and Greene 2017]. The learned chemical information and patterns from these encodings is what drives the purpose of their use in this project, as their generalization allows for important motifs in both protein and DNA sequences to be the driving force behind strictly computational protein-DNA binding affinity prediction.

Using the typical multivariate Gaussian VAE architecture as shown in Figure 1, we constructed, trained, and optimized two separate variational autoencoders: one to learn a protein latent space, and one to learn a DNA latent space. In designing the best encoder and decoder implementation for both, we first started with constructing a convolutional VAE architecture due to our previous experience with one-hot encoded biological sequences as inputs for convolutional neural networks.

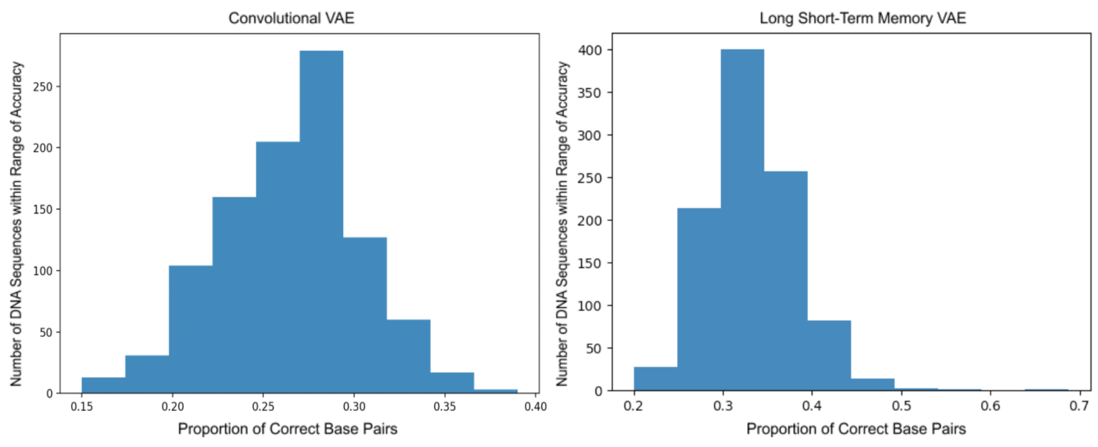


Figure 2: Accuracy between the Convolutional and LSTM VAE Architectures on a Random Sampling of 1000 Sequences from the DNA Test Dataset

Long short-term memory (LSTM) autoencoders are specifically well equiped to handle sequence data, so we also investigated an LSTM architecture as another possibility.

After the training of both of these potential architectures, we determined that both the LSTM and convolutional autoencoders generated marginally different accuracies, further discussed in Section 3.3. Due to our experience with optimizing CNNs, the convolutional autoencoder was chosen to be the main encoder that would be used for both latent space generations, as there was initially no seen benefit between the two implementations.

As a result, the architectures of the VAEs for both the protein and DNA latent space creation are almost identical: each takes in a one-hot encoded input of either amino acid sequences or DNA base pair sequences, and each is sent through in its encoder two convolutional and max pooling layers followed by a fully connected layer, dropout layer, and another fully connected layer. The resulting latent space is sent through the decoder for reconstruction purposes, which utilizes equal amounts of fully connected layers with transpose-convolutional layers and upscaling layers as substitutes for the convolutional and pooling layers respectively. Following determining this architecture, we performed a hyperparameter grid search for each model to determine the optimal regularization coefficient for our L2 Norm regularizer and dropout rate. Assuming that the different inputs for each model may require different utilization of the latent space, the size of the encoding was also incorporated into the this grid search prior to final training of the models. Although the model architectures are similar, the hyperparameters were individualized to each, and were the main differences between the two. Following this, we trained each model on significant chunks of each of their respective respective datasets.

3.2.2 Regression Neural Network

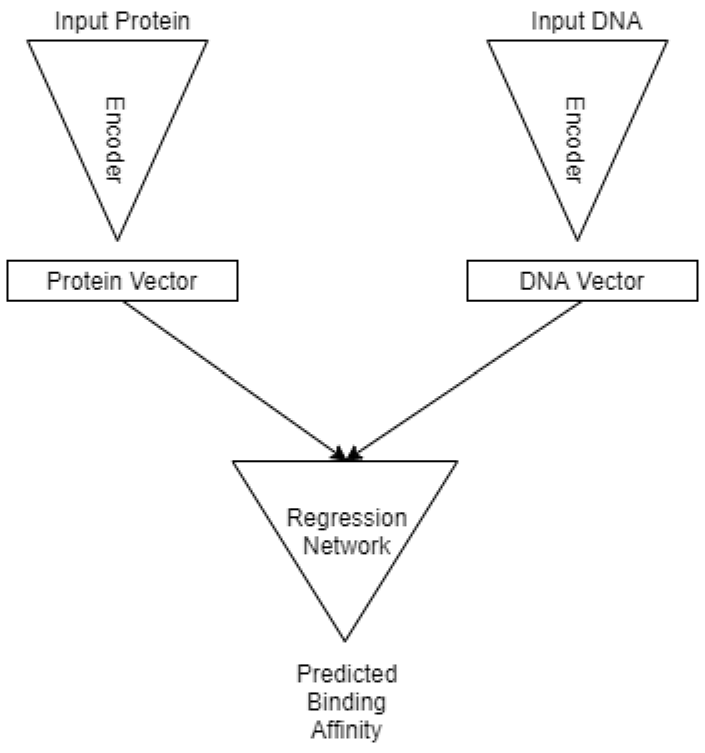


Figure 3: Summary of full predictive network architecture

We then trained a neural network to take as input a concatenation of a single protein latent vector with a single DNA latent vector and output a prediction of the binding affinity energy. This network was trained using the aforementioned ProNIT dataset consisting of Protein and DNA complex pairs and their binding affinity energy. This network design is intentionally simple, with the goal being that the dimensionality reduction of the VAE reduces any need for this network to perform any complex functions. Rather, this network only needs to learn some functional relationship from the low-dimension encoding to the predicted binding affinity data. In total, the combined architecture is depicted in Figure 3.

3.2.3 Network Training

In this project, we considered four different regression architectures. The first was the simple linear regression, computing the least squares estimator over the training data. The second was a simple neural network, consisting of two fully connected ReLU layers with 64 units each and a dropout layer in between, with a tunable dropout rate. The third was a 3-layer network, with a dropout layer in between, and the fourth was the same 3-layer network but using Leaky ReLU activations instead of ReLUs. In essence, our search over networks was to determine whether or not the network had advantages over simple regression, and also whether increasing model complexity led to improved performance.

Unfortunately, a key issue that arose early in Neural Network testing was the poor data distribution, as well as a lack of data availability. While it was initially anticipated that the lack of data would be a problem in training a model like this, the original intention was that dimensionality reduction would reduce the challenge of this issue. Namely, by reducing the input dimension down to a vector of size about 200, we would require much less data to properly train in comparison to an input space of $4^{500} * 21^{500}$ - the number of possible DNA inputs times the number of possible protein inputs (we cap input lengths at 500). However, after processing ProNIT, we were left with only 3000 unique Protein-DNA pairs with which to train, and even then some of the data is redundant. More importantly, the data was heavily biased: the vast majority of Protein-DNA binding affinities lie in the range -8 to -12 kcal/mol.

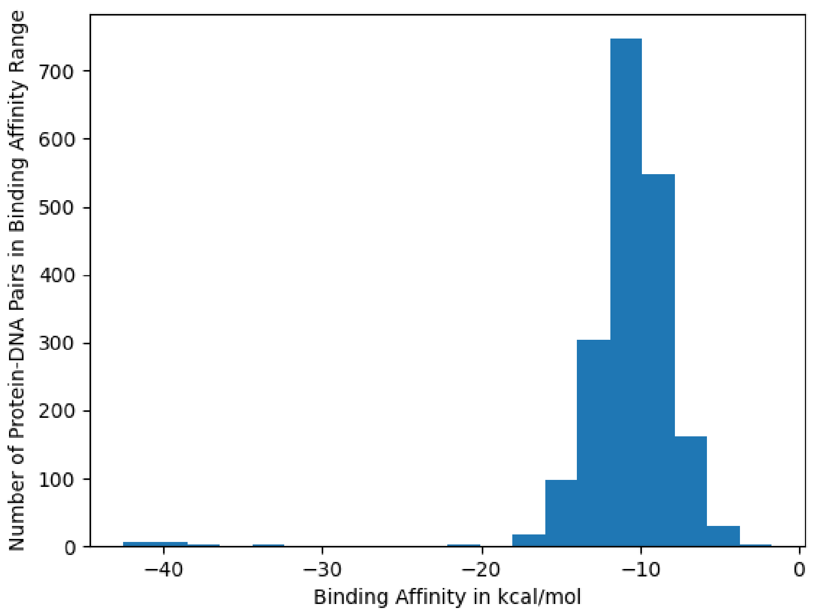


Figure 4: Post-Processing ProNIT Dataset Histogram

As such, trained models learn to quickly minimize error, both squared and absolute, by simply outputting in some stochastic region around the mean of the data. To attempt to resolve this issue, we aimed to lessen the bias of the dataset without inducing any unintended selection bias through removal of samples. To that end, we also implemented stochastic data selection, whereby data originating from the biased region of the dataset described above have only a 15% (a tunable hyperparameter) chance of being actually selected for use as training data. After implementing this trick, we noticed a much better qualitative result, with the model much more willing to attempt to predict outliers, i.e. complexes with large binding affinities.

3.3 Results

3.3.1 VAE Architecture

The VAEs for both protein and DNA sequences were trained on significant parts of their respective datasets: the former was trained on $\tilde{9}0,000$ maximum of 500 amino acid length protein sequences and the latter was trained on $\tilde{2}0,000$ 500 base pair DNA sequences. The encoding process, tuned

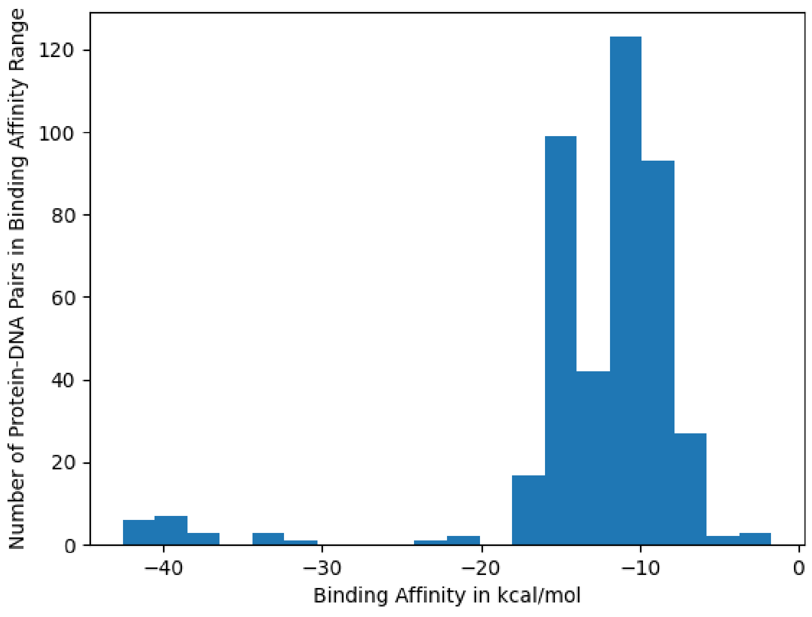
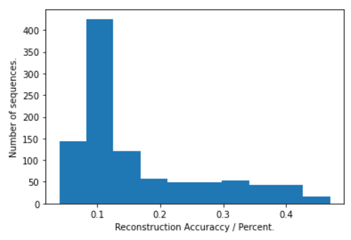


Figure 5: Post-Processing ProNIT Dataset with Stochastic Data Selection

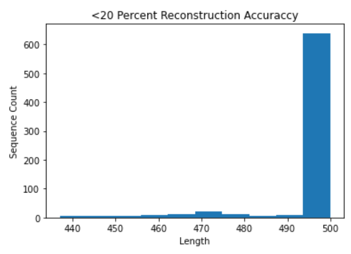
through a hyperparameter grid search, resulted in roughly a 10:1 maximum compression for each type of sequences, the DNA being compacted into length 64 latent space representations and the protein into length 40. Specifically for the protein representation, this corresponds to a single latent space dimension per alpha helix or beta sheet, since each dimension contains information for an average of 10 amino acids [Kabsch and Sander 1983].

While designing each of the protein and DNA convolutional autoencoders, single layer architectures showed poor performance compared with a two layer alternative. This motivated exploration of increasing model capacity as a way of improving reconstruction performance. However, deeper three and four convolutional layer models failed to show improved performance on both test sets, and thus the final architecture displayed a 32 filter layer followed by a 64 filter layer for both. We also found that dropout changes influenced model optimization in both of the VAE architectures, but the use of l2 normalization did not improve reconstruction performance within the protein VAE. While multiple sizes of convolution were tested as a part of the model optimization process, a convolutional filter of shape (1, 10) showed greatest performance for proteins and a convolutional filter of (4,1) showed the same for the DNA. Research and empirical results also pointed towards the use of Evidence Lower Bound (ELBO) as the loss functions for these VAEs.

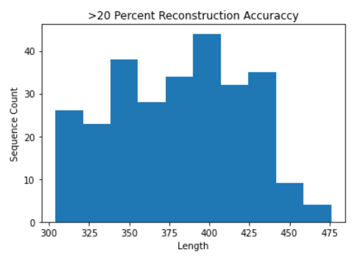
3.3.2 Protein VAE Results



(a) Distribution of Protein VAE Reconstruction Accuracies



(b) Length of Poorly Reconstructed Protein Sequences



(c) Length of Well Reconstructed Protein Sequences

Figure 6: Reconstruction Accuracy and Length of 1000 Randomly Sampled Test Set Protein Sequences

Performance in reconstructing the protein input sequence was highly varied, with a mode around 10 percent reconstruction accuracy, and with a long tail ending at 40 percent reconstruction accuracy, see figure 6. This suggested that some factor was present that allowed some sequences to be well represented by the VAE. Sampling of well reconstructed sequences (accuracy greater than 20%) showed a strong negative relationship between protein length and reconstruction accuracy. The average length of well reconstructed proteins was 381 amino acids which is 23 percent greater than for poorly reconstructed sequences. Despite these potential sources for future VAE improvement the VAE still managed a level of reconstruction accuracy (16.9 percent) over 3 fold greater than randomized peptide sequence generation (5 percent).

Early attempts at model training were improved by removing short protein with lengths shorter than 300. Learning to reconstruct these peptides placed a great burden on the model to learn to recognize identifying features of the short peptides, leading to overly conservative estimates of protein length. It was decided early on that said sequences were likely sufficiently structurally byzantine from DNA binding proteins so that excluding them from further VAE data sets would be unlikely to hinder representation of proteins highly represented in the pronit dataset. Further it was decided that by pruning longer proteins in the dataset to 500 amino acids, we would likely allow the model to focus on learning chemical structure, rather than simply length.

3.3.3 DNA VAE Results

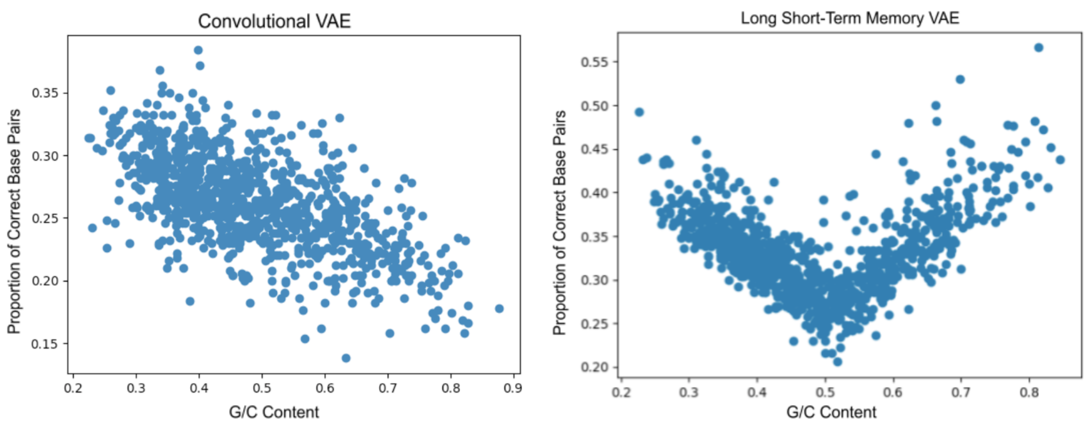


Figure 7: How G/C Content affected Reconstruction Accuracy between the Convolutional and LSTM VAE Architectures on a Random Sample of 1000 Sequences from the DNA Test Dataset

In comparison to the latent space generated from the encoding of the protein VAE, the DNA VAE was highly fitting to a Gaussian distribution, as seen in Figure 7. Centered around a 28% accuracy, this suggests that the encoder is properly utilizing and conforming to its prior to create a fitting latent space for each input. Thus, unable to draw any interpretable conclusions about the latent space from accuracy alone, we next investigated how reconstruction accuracy changed for DNA sequences with different G/C content. This produced striking results, showing that there is a clear negative correlation between G/C content and reconstruction accuracy for this specific convolutional autoencoder. To improve this specific model in the future, we would like to investigate how the autoencoder handles learning strictly DNA sequences with near equal G/C and A/T contents, as this would restrict these potential influences on the encoding of the latent space.

To understand whether the learning of G/C content was the result of the dataset or the model, we investigated how G/C content and accuracy of reconstruction were related in the LSTM autoencoder model that was discussed in Section 3.2.1. The results of this are shown in Figure 7. It is clear that in the LSTM system, the reconstruction accuracy rose for more homogeneous DNA sequences such as ones with very high or very low G/C content. This matches intuition, as more complex DNA sequences would be more difficult to represent in a compressed latent space. Given the time constraint of the class, we were unable to pivot our use of the DNA VAE from the convolutional architecture to the LSTM architecture to investigate how this would affect the prediction of binding affinity. However, in future work we plan to determine how the LSTM model changes binding affinity prediction accuracy compared to the current convolutional implementation.

3.3.4 Neural Network and Binding Affinity Results

After tuning the hyperparameters for the variational autoencoders, as previously mentioned it was found that a latent space of 64 dimensions maximized the ELBO for the DNA VAE, and a latent space of 40 dimensions maximized the ELBO for the Protein VAE. Since both the latent mean and variance were passed together as inputs, each input to the neural network was a 1-D vector of length 208 values.

To decide upon the neural network architecture, an ”architecture search” was performed over the four candidate models described above. This was a key benefit over the low-dimensionality encoding: because the input vector is extremely small, we could train over many epochs with thousands of datapoints within seconds. As a result, it became feasible to perform wide hyperparameter optimizations. After training all four models over 100 epochs using both Mean Absolute Error and Mean Squared Error loss functions, the resulting errors over the test set are tabulated below. Recall that the reduced dataset was the dataset obtained by selecting over-represented binding affinities with probability 0.15.

Train Environment	Linear Baseline	2 Layer NN	3 Layer NN	Leaky 3 Layer NN
Base, MSE	11.913	6.9515	7.126	6.859
Base, MAE	1.934	1.882	2.380	1.760
Reduced, MSE	15.735	14.174	17.242	12.23
Reduced, MAE	2.238	2.359	2.274	2.268

Looking at the performance between the architectures, it becomes quite clear that all the neural network architectures seem to have reasonable performance increases over the linear regression methods (at least on the base dataset). Moreover, the 2 - layer Neural Network seems to already be robust and sufficient for learning a functional relationship between the latent space and the binding affinity values, with the 3 layer network performing worse on the Base Dataset. We attribute this decrease in performance to the classical problem of overfitting; the increased model complexity in the 3 layer network decreases its ability to generalize, and this problem is especially exacerbated due to the simplicity of the input. However, there is a noticeable benefit to using the Leaky 3 Layer network over the 2 Layer, with improvements in all 4 computed metrics.

The original motivation for implementing the Leaky network was the idea that the binding affinity outputs lie entirely in the negative value space; if the binding affinity is positive, then the complex cannot form in the first place. Thus, by allowing negative values to propagate through the network via the Leaky channels, we make it easier for the network to learn and compute negative values throughout training. One can think of this as adding a prior belief to the network, which in turn helps the network reach a more optimal solution. In the reduced dataset, the number of training samples is relatively small, 400-500 pairs; thus, there is a more pronounced improvement in MSE. In contrast, the raw dataset has thousands of pairs, and thus the improvement of a prior is relatively minor.

As was mentioned above, the motivation behind implementing the reduced dataset was to force the network not to simply learn to output the mean of the data. From the above table, this reducing technique seems to lead to much worse performance: the resulting models seem to be quantitatively worse, suffering with higher MSE and MAE for all four architectures. However, we observed a qualitative improvement: a reducing in clustering. As one can see from the plot in Figure 8, training on the base dataset heavily encourages simply outputting values in the most represented range. By training on the reduced dataset switching to a MSE loss function, we observe a direct improvement in this qualitative distribution, where the model is immediately more willing to predict outliers, as shown in Figure 9.

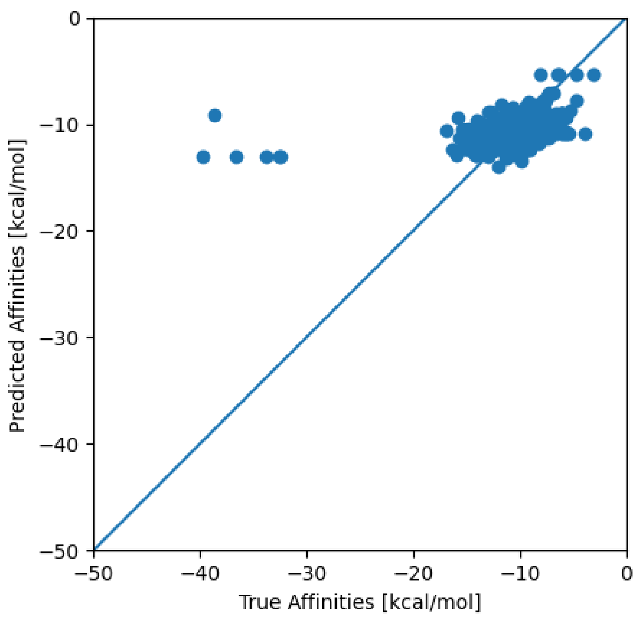


Figure 8: Predicted vs True Binding Affinities for the Leaky 3 Layer Network using the Base Dataset and trained on MSE loss for 100 epochs.

After looking at the data distribution, one can also notice the curious behavior of "horizontal" predictions, where it seems many points have the same predicted binding affinities. Unfortunately, this is due to discrepancies between the autoencoder and final neural network design. While the true input sequences are distinct and have distinct affinities, the one-hot encodings used for autoencoder training did not account for RNA sequences (by encoding Uracil) or "X" nucleotides, where "X" is allowed to vary with the same binding affinity result, while the DNA data used to train the final neural network did. Thus, distinct sequences with these characters get encoded into the same latent space, and the neural network predicts the same value. Regardless, the implementation of these techniques still leads to much more varied results, with the network willing to predict a wider range of values, and with fewer obvious outliers.

3.3.5 Rank Correlation Plots

Following this analysis of the data distribution, it seemed prudent to reconsider the objective of the model. It is difficult to train a perfect model for predicting binding affinity from the latent space due to the limited data available, and moreover, the prediction is altogether not too meaningful. When

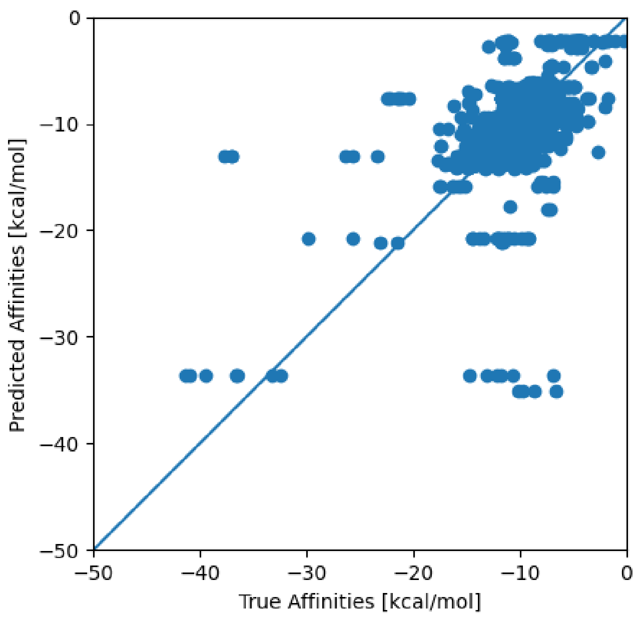


Figure 9: Predicted vs True Binding Affinities for the Leaky 3 Layer Network using the Reduced Dataset and training on MSE loss for 100 epochs.

the majority of binding affinities lie in the range of -8 to -14 kcal/mol, being off by 1-2 kcal/mol leads to an equilibrium constant being off by a factor of about 4, which, while significant if considering concentrations analytically, is not too important when considering protein-DNA binding potency.

Thus, the more interesting application was in considering whether or not we could rank relative protein-DNA binding affinities. If the model could predict whether or not a protein is a stronger DNA-binder than another, then this has biological implications in considering cell dynamics, where we can better model how proteins interact with DNA. With this application in mind, we explored rank correlation plots, where for each test data point, we plotted its rank among true binding affinities against its rank among predicted binding affinities. Qualitatively, a good predictor would lead to a strongly linear relationship in the rank correlation plot. For the leaky 3 layer model, the resulting rank correlation plots for the two datasets are depicted in Figures 10 and 11.

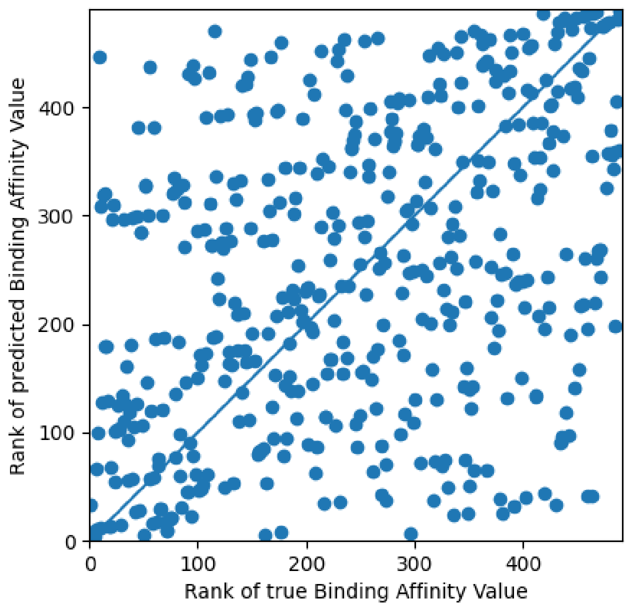


Figure 10: Rank Correlation Plot for the Leaky 3 Layer Network trained with MSE loss function on Raw Data

Unfortunately, the rank correlation plots paint a very discouraging picture. The distribution of points is almost entirely random, and it clearly indicates the inability of the model to be used to rank relative binding affinities for proteins. Rather, it seems an altogether distinct architecture would be much better suited for attempting this task. Moreover, it seems also that such a task is made extraordinarily difficult by the data: empirical measurements of binding affinity are often subject to numerous inaccuracies and high uncertainty, which would significantly impede the training of any

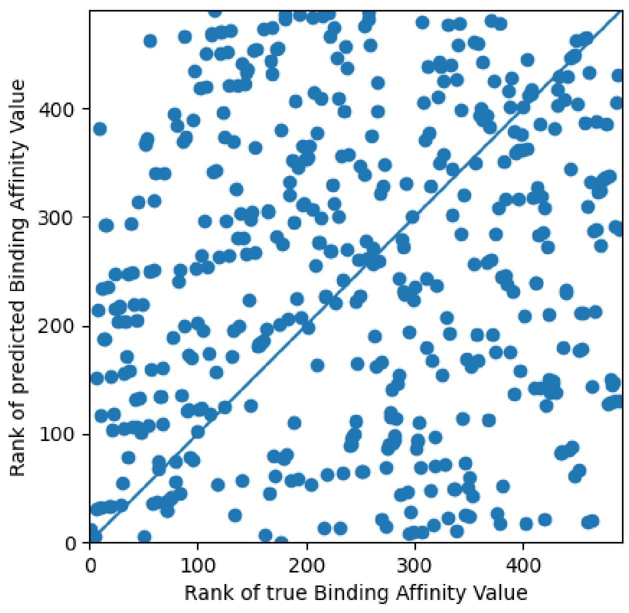


Figure 11: Rank Correlation Plot for the Leaky 3 Layer network trained with MSE loss function on Reduced Data

potential regressive model.

Additionally, the rank correlation plots for the reduced and raw data look exactly the same. This suggests that while reducing the data concentration succeeds in allowing the model to take risks and predict outliers, it also ends up simply allowing the network to increase its variance. Rather than learning any new structure in the data, the network is more "unsure" in the distribution of its outputs, since the total quantity of data is smaller; there is no true increase in learning. While unfortunate, this does not mean the results are meaningless: rather, it suggests a need for a much more representative latent space, as well as a much better curated dataset. Since the results are qualitatively better and easier to interpret, we still consider the reduced dataset in the remainder of this project.

3.3.6 Outlier Accuracy

Environment	Precision	Recall
3-Layer Leaky, MSE on Normalized Data	62.5%	83.3%
3-Layer Leaky, MSE on Raw Data	71.4%	83.3 %

Finally, still considering the importance of "outlier complexes," i.e., protein-dna pairs with extremely high binding affinities, we compute the relative ability of the model to detect these outliers against the true dataset. To that end, we compute the precision, the fraction of predicted outliers that are indeed outliers, and the recall, the fraction of outliers that were successfully detected.

As mentioned above, our architecture suffers some issues within its dimensionality reduction structure. However, the results are still promising: Using the Leaky 3 layer network, defining outliers as those with binding affinity less than $-30 kcal/mol$, our network can achieve a precision rate of 62.5% and a recall rate of 83%. For a network learning entirely from VAE-defined features, this rate is relatively promising. Thus, this seems to be the most promising application of the network, since there is not enough data available to fine-tune some true predictor.

3.3.7 Conclusion

Overall, we believe our architecture holds promise. While the rank correlation plots are discouraging, and there are myriad issues with encoding and data quantity, the model is still able to provide insight into outlier prediction, and it seems that many of these issues could have been fixed over a longer-term project. Moreover, the results themselves are remarkable: after all, the features learned by the VAE are totally unrelated to the eventual binding affinity prediction problem. To be able to have reasonable results is theoretically interesting, as they indicate that there indeed might be some exploitable underlying structure within DNA-binding proteins and protein-binding regions of DNA. Moreover, this architecture is remarkable fast: it can be trained in seconds, returning predictions instantly. In comparison to the base PreDBA model with hand-designed features, our model is hundreds of times faster. It is our hope that future works may be able to expand upon this introductory architecture to develop novel systems to rapid binding affinity prediction.

4 Future Goals

While the model's performance is somewhat promising on its own, one of the most important next steps would clearly be to incorporate hand-designed features into the model, and then be able to compare the relative importance of hand-designed and latent space features. Taking insight from other works in neural network prediction architectures, it is often much more effective to assist machine-learned features with chemically-important manually computed features, so that the network may learn to gain insight into some new, previously undiscovered space of the input [DW; n.d.]. Moreover, it would be greatly interesting to explore the relative importance of the two features: if there are performance improvements, how much can be attributed to the robustness of the latent space?

Beyond structural improvements, there is simply much more work to be done on fine tuning the architecture as-is. The current state of the VAE provides a relatively poor reconstruction of the data, whether for a lack of latent expressivity or otherwise. As such, extended work on potential VAE architectures, as discussed in Section 3.3.3, would be a good next step to improve this aspect. Moreover, the datasets are insufficiently robust and contain numerous duplicate samples; moreover, the datasets contain other interesting information, like base and mutation DNA strand, stoichiometry, pH, and other values that could be significant in training a binding affinity predictor.

5 Comparison with Original Proposal

Our final project remained mostly faithful to the original proposal. The architecture is the same, and the training procedures and datasets were precisely what we envisioned. However, the "stretch" goal that we laid out, i.e., predicting optimal DNA-binding proteins (or vice versa) was not feasible to achieve with the time available. Moreover, the performance of the model was certainly not sufficient to achieve any meaningful results in this space. In large part, this was due to our underestimation of the time required to properly fine tune and train the necessary architectures. The VAEs proved much harder to train than initially anticipated, and the lack of attention paid towards properly curating the input datasets between components meant that their eventual performance was relatively subpar.

Certainly, our proposal was overly ambitious: designing and implementing a model with three different subnetworks was extremely challenging, especially because interpreting the progress of VAE training was rather difficult. Moreover, we originally hoped to obtain our own specially created datasets for the task, but this proved much, much more challenging than initially anticipated, and instead we resorted to processing and adapting existing datasets for our purposes, which certainly significantly hampered our eventual results. However, the initial proposal was not vague: we had clearly laid out goals, and their implementation was clear and succinct for all team members, making the combination of the VAEs into the final model relatively easy.

6 Commentary on Experience

The two biggest issues with this project were clearly in the VAE and in properly processing the datasets. In our original estimation of effort required for the VAEs, we thought each VAE could be trained within 12 hours, but this was a severe underestimate. Moreover, we failed to comprehend the idea that the VAEs were much more important than the final neural network: the low dimensional encoding would already allow the Neural Network to be trained in seconds, so tuning it would be relatively easy, whereas the VAEs would require hours to train each time. As a result, our misallocation of time cost us in being able to fully tune our model to our needs.

The second biggest issue was in properly understanding the data we were given. We had access to essentially all data necessary for our purpose, but we failed to realize that it was crucial that we tailored the rest of our models to ProNIT's design. For example, DNA sequences were inputted in lengths ranging from 5-500 from ProNIT, but our general dataset for training the DNA VAE consisted only of sequences of length exactly 500. Similarly, the ProNIT dataset has peculiar formatting, offering base and mutated DNA strands, as well as base and mutated protein sequences for each protein-DNA pair, and we failed to properly comprehend and take advantage of this data. If we were to redo this project, it is extremely clear that we should have spent much more time properly understanding and evaluating the given data, rather than focusing on procuring data or fine-tuning the model.

Andrew Wu's Commentary: Specifically for me, I predominantly worked on producing the final Neural Network to do the binding affinity prediction. I greatly enjoyed this part of the project, because it was an extremely open-ended problem. I could implement whatever network made sense, and it was great to see that the Leaky network was able to outperform some generic models. Even more importantly, I sincerely enjoyed being able to explore the intepretability of the resulting model. I had the chance to analyze the distribution of predictions, the relative rank correlation plots, and the precision recall values of outliers; basically any metric that I thought was interesting for the resulting model. If I were to start over, I think I would recommend paying much more attention towards analyzing the dataset, specifically ProNIT. When I originally processed ProNIT for our model, I failed to properly analyze all the data available, and ended up using misrepresentative data, and this ended up being a major waste of time and resources.

Noah Faro's Commentary: Prior to this class, I did not have experience with VAEs and thus was very excited to take on the role of designing the DNA VAE. This entire experience was certainly my favorite part of the project, as researching the functionality and implementation of VAEs allowed me to develop the skill of learning completely new deep learning model types without clear guidance, exposing me to reading scientific papers, scraping documentations, and understanding generic implementation patterns within the the scope of machine learning. Although I do appreciate the skills that I received from this process, it was certainly difficult to successfully optimize the VAE given the amount of time necessary to train it. Most of my time was spent waiting for the training to finish before iterating on my architecture or hyperparameters, and this made the overall time spent on the project drastically more than anticipated. Had I had the opportunity to restart this project, I would have done more interpretability on the potential VAE implementation types prior to taking the time to fully optimize. Had I done this, I may have used a different architecture than the convolutional VAE, potentially giving better results for overall project.

Andrew Hennes's Commentary: Working on the VAE for our project was my first time exploring implementing and training an autoencoder model. I was especially intersested in working with a VAE architecture as the mathematics and probabilistic justifications behind the model seem interesting from both a theoretical and practical generative standpoint. I also really enjoyed getting to work with biological datasets from the ensembl database and learn by myslef how to process and prepare the datasets and deal with practical limitations and problems that arose during the process. Looking back I do wish that I had considered the structure of my training and test datasets and considered discrepencies between them which could have compromised the generalizability of the VAE architecture. I think that learning a chemical latent space is a promising idea as it suggests the potential for unbiasedly learning the underlying structure of proteins only from their primary sequence. However I think I should have tried multiple types of architectures, for example LSTM's and CNN architectures with diffuse layers, earlier on instead of spending so long trying to optimize a single model architecture. Overall however I really enjoyed the experience and look forward to contributing to more such projects going forward.

7 Division of Labor

Our division of labor tracked very similarly with our Ghantt chart designed in the original proposal. Noah gathered and processed the ProNIT dataset for binding affinities, and both Andrew Hennes and Andrew Wu brought together the generic protein and DNA sequence datasets. Noah designed, trained, and optimized the DNA convolutional variational autoencoder and exclusively designed, trained and investigated the LSTM VAE on the DNA dataset. Andrew Hennes worked tangentially on a similar design, training, and optimization for the Protein convolutional VAE. Andrew Wu took the lead on the design for the final neural network to predict the binding affinities, incorporating the ProNIT dataset, training the model, and optimizing its hyper parameters. All members of the group met weekly to update on progress, and also frequently met to work in pairs and help solve technical issues or discuss new ideas. The entirety of the team worked on each milestone assignment, and on this final report.

References

DW; Kucer M;Loui AC;Messinger (n.d.). *Leveraging expert feature knowledge for predicting image aesthetics.* URL: https://pubmed.ncbi.nlm.nih.gov/29994210/.

Flores, Steven (n.d.). *Variational Autoencoders are Beautiful.* URL: https://www.compthree.com/blog/autoencoder/.

Hawkins-Hooker, Alex et al. (2021). "Generating functional protein variants with variational autoencoders". In: *PLOS Computational Biology* 17.2. DOI: 10.1371/journal.pcbi.1008736.

Hou, Xianxu et al. (2019). "Improving Variational Autoencoder with Deep Feature Consistent and Generative Adversarial Training". In: *CoRR* abs/1906.01984. arXiv: 1906.01984. URL: http://arxiv.org/abs/1906.01984.

Hu, Siquan, Ruixiong Ma, and Haiou Wang (Nov. 2019). "An improved deep learning method for predicting DNA-binding proteins based on contextual features in amino acid sequences". In: *PLoS ONE* 14.11. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0225317. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6855455/ (visited on 03/25/2021).

Jin, Wengong, Regina Barzilay, and Tommi Jaakkola (Feb. 2018). "Junction Tree Variational Autoencoder for Molecular Graph Generation". en. In: URL: https://www.arxiv-vanity.com/papers/1802.04364/ (visited on 03/25/2021).

Kabsch, W and C Sander (1983). *How good are predictions of protein secondary structure?* URL: https://pubmed.ncbi.nlm.nih.gov/6852232/.

Naderi, Habibeh, Behrouz Haji Soleimani, and Stan Matwin (2020). "Generating High-Fidelity Images with Disentangled Adversarial VAEs and Structure-Aware Loss". In: *2020 International Joint Conference on Neural Networks (IJCNN).* DOI: 10.1109/ijcnn48605.2020.9207056.

Sarai, Akinori et al. (2002). "ProNIT: Thermodynamic Database for Protein-Nucleic Acid Interactions." In: *Seibutsu Butsuri* 42.6, pp. 279–281. DOI: 10.2142/biophys.42.279.

Way, Gregory P. and Casey S. Greene (2017). “Extracting a Biologically Relevant Latent Space from Cancer Transcriptomes with Variational Autoencoders”. In: DOI: 10.1101/174474.

Yang, Wenyi and Lei Deng (Jan. 2020). “PreDBA: A heterogeneous ensemble approach for predicting protein-DNA binding affinity”. In: *Scientific Reports* 10. ISSN: 2045-2322. DOI: 10.1038/s41598-020-57778-1. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6987227/ (visited on 03/25/2021).