Predicting Changes in Protein Thermostability From Single-Site Residue Mutations Using Structural Information

Manjot Sangha
Massachusetts Institute of Technology
manjot@mit.edu

John Clarke
Massachusetts Institute of Technology
jaclarke@mit.edu

I. Introduction

Proteins are quite literally the building blocks of life. They play a role in virtually all biological processes ranging from catalyzing the reactions that allow plants to produce energy from the sun to triggering an immune response when you get sick. The ubiquitous nature of proteins has attracted a lot of attention from researchers in fields such as agriculture, medicine and chemical engineering, among others. However, despite the intense research efforts that have been exerted, many of the complexities of proteins and how they function remain a mystery. For these reasons they remain an exciting target for further investigation.

Proteins are large macromolecules composed of a chain of amino acids folded in a unique three-dimensional configuration. A proteins function is largely dictated by its structure, which is in turn largely dictated by the sequence of its constituent amino acids. Whereas protein sequence data is relatively easy to obtain, the experiments required to determine the chemical properties of proteins are more expensive in terms of time and cost. For this reason the ability to predict the chemical properties of proteins would be very useful.

Protein thermostability is an important factor to consider for protein structure prediction, protein function prediction and protein design. In many of these applications what is most interesting is how the proteins thermostability changes in response to mutations at certain residues, known as point mutations. For example the ability to optimize a proteins thermostability would allow for the creation of more commercially viable protein variants that have a longer half-life.

Protein thermostability is generally measured as the change in Gibb's Free Energy (ΔG) associated with the folding of a protein. A general rule of thumb is that a protein's three-dimensional conformation is located at an energy minima in order to maximize stability thus giving a negative ΔG value. When predicting changes in thermostability associated with mutations we are interested in the change in the change in Gibb's Free Energy ($\Delta\Delta G$). By convention a positive value for $\Delta\Delta G$ is stabilizing and a negative value is destabilizing.

In most applications the sign of $\Delta\Delta G$ (increase or decrease in stability) is more important than the exact value so we framed this problem as a classification task. Despite the abundance of previous approaches to predict the mutation-triggered change in the thermostability of a protein, we found that hardly any take advantage of structural information. Studies that leveraged this information did so in a simplistic manner, likely losing understanding of the structure's underlying distribution. We hypothesize that structural information would be highly relevant to this task since a large portion of overall stability is derived from spatial interactions between amino acids. For this reason, in our investigation we set out to incorporate three-dimensional information into a predictor for the change in protein thermostability given a mutation.

In terms of the overall organization of this paper, we first discuss past approaches and their reported classification accuracies in Section II. Then in Section III we discuss our proposed approach and the

intuition guiding its formulation. Next in Section IV we discuss our methods in more detail including the data we used, our feature representation, network architecture and how we trained our network. Then in Section V we discuss the results of our experimentation, and finally in Section VI we present our conclusions and potential future work.

II. Related Works

There has been a number of approaches to predicting the direction and magnitude of protein thermostability changes in response to point mutations. All have used some sort of structural and sequence-based data. In this context, structural data describe the orientation of atoms in the relevant space (generally 2D or 3D), while sequence data are the chains of amino acids (1D). We have chosen to focus on predicting the direction of stability changes in response to point mutations. Capriotti et al. described the first successful neural method in this space. They applied their method to the Protherm Database–a collection of numerical data of thermodynamic parameters such as Gibbs free energy change, enthalpy change, heat capacity change, transition temperature etc. for wild type and mutant proteins, that are important for understanding the structure and stability of proteins [1]. Using amino acid sequence data, they predict the direction of stability change with (1) a feed-forward neural network and (2) a neural net augmented with another energy-based method [2]. The same group developed I-Mutant2.0, a support vector machine that performs the same classification task, but is capable of using either structural or sequence-based data. Their model can be access and used on this web interface: http://gpcr.biocomp.unibo.it/cgi/predictors/I-Mutant2.0/I-Mutant2.0.cgi [3]. Cheng et al. created MuPRO: a support vector-based method, achieving better results in the prediction of direction [4]. Masso Vaisman developed AUTO-MUTE: a random forest model and support vector machine to perform the same classification, focusing on 3D structural information to make their predictions [5]. Lastly, Chen et al. proposed iStable: an ensemble method using a support vector machine to integrate various weaker predictors. Their model can be accessed here: http://predictor.nchu.edu.tw/iStable [6].

TABLE I. The prediction accuracies of iStable and earlier models on the M1311 dataset. Ensemble methods are starred. [6]

Predictors	Acc
iStable*	0.969
I-Mutant_PDB	0.800
I-Mutant_SEQ	0.883
AUTO-MUTE_RF*	0.958
AUTO-MUTE_SVM	0.907
MUPRO_SVM	0.896
PopMuSiC2.0	0.724
CUPSAT	0.742
Majority Voting*	0.902

The accuracies of these and other related models are summarized in **Table 1**.

III. Approach

We propose a new approach to predicting changes in protein thermostability that better incorporates structural information. Currently most approaches that incorporate structural information do so in a simplistic way, generally involving summary statistics that result in the loss of information. For example, one approach is to generate a vector of length 20 where each element represents an amino acid and its value is the number of times that amino acid appears within a bounding region around the mutation site. The loss of sequence based and structural information is clear here. Other approaches use values computed from the 3D structure such as solvent accessibility.

Although these approaches use structural information, they are not good representations and disregard a lot of the information contained in structural data. For example they no longer contain information on the spatial correlation of atoms within the structure.

In order to address this problem we propose to use a convolutional neural network (CNN) as a method of feature extraction. Protein structures lend themselves well for use with CNNs because they are composed of many layers of structure of increasing complexity (primary, secondary, tertiary) with many locally relevant features. This draws a parallel to why CNNs work so well with images. It has been shown that filters in earlier convolutional layers learn simple features (lines, edges) and then later

convolutional layers learn more complex compound structures (shapes, objects). In a similar manner we predict the CNN will be able to learn the various levels of complexity in protein structure with early layers learning things like chemical bonds and later layers learning more complex structures like alpha helices and 3D structural motifs.

There has been some previous work related to the use of CNNs with protein structure that gives our theory more credence. AtomNet is a CNN used to predict small molecule activity given the structure of the protein-small molecule complex and has reported impressive results. They have also shown that their convoltional filters were able to learn chemical constructs such as functional groups and how they interact [7].

Figure 1 shows a high level overview of our method. We will crop the input structure to only contain the region in the neighbourhood of the mutation based on some distance threshold. This is to reduce noise and the dimensionality of the representation since local interactions should be much more relevant than long distance interactions. We looked at a couple different representations that maintain the three-dimensional information and these are discussed further in Sections IV and V. This 3D representation is then input to a 3D CNN where the filters are three-dimensional instead of the more common two-dimensional filters. The output of these convolutional layers will then be a learned representation of the information in the 3D structure. This feature embedding is then concatenated with a vector encoding which amino acids were mutated (we discuss this encoding in Section IV) and then passed through a feed-forward neural network. The output is a single node whose value is then passed through a sigmoid function. If the value is ≥ 0.5, then we predict an increase, otherwise a decrease in protein thermostability.

IV. Methods

In this section we describe in detail the experimental methods we used including the acquisition and processing of our data, the architecture of our network and the methods we used to train it.

A. Dataset

For our dataset we selected a set of mutations in proteins whose structures were available in the Protein Data Bank (PDB) and where the experimentally determined $\Delta\Delta G$ values for the mutation were available [2]. Given the 3D structure, we sought to predict the change in thermostability.

We were originally going to use data directly derived from ProTherm, one of the largest publicly available datasets containing experimentally determined thermodynamic data for proteins and their mutants. However, when we were working on this project the ProTherm site was down and we were unable to access it. Instead we used a previously published dataset which was extracted from ProTherm by Capriotti et. al and last updated in 2006 [1]. This dataset contained all entries in ProTherm that (1) had experimentally determined values for $\Delta\Delta G$, (2) had a protein structure available in the PDB and (3) whose data regarded single point mutations. Since these data were compiled in 2006, it is likely that more data meeting these criteria are available today and eventual access to these data will most likely result in an improved model.

The data comprised of 1948 mutations spanning over 58 proteins. Of these mutations 562 increased in stability (28.9%) and 1386 decreased (71.1%). This data was randomly split into 70% training data, 20% validation data and 10% test data.

B. Data Processing

Structure information was retrieved for each protein as a .cif file from the PDB. We converted the representation in the .cif file to that which we passed through our CNN. We treated the PDB file as essentially describing a point cloud representation of atoms. For each mutation, we cropped out the region near the mutation site. First we set the origin as the center of mass of the original amino acid. We then computed the distance of all atoms to this origin and only kept those that were within a threshold which we set to be 10Å.

Next we needed to convert these filtered atoms into a matrix representation that could be passed into a CNN. We tried a couple different representations. First, we tried thinking of a 3D lattice at 1Åunit

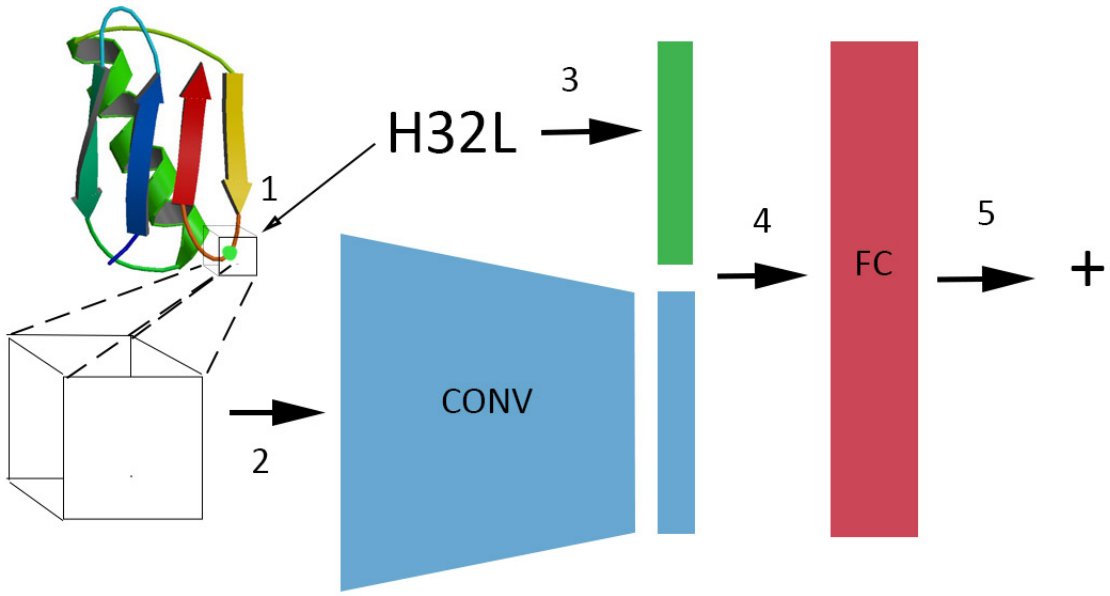

Fig. 1. An overview of our prediction workflow for a given protein (pictured) and mutation (H32L). **(1)** The location in the structure of the mutated residue is found (green dot) and the structure is cropped to focus on a bounding box centered about the mutated residue, **(2)** The 3D representation of the structure in the mutation neighbourhood is passed through a 3D convolutional neural network, **(3)** a vector encoding the mutation is created, **(4)** the mutation vector and the structure feature representation vector are concatenated and passed through a fully connected layer, **(5)** the output is passed through a sigmoid function and thresholded to determine a final prediction.

spacing (so a 21^3 array). Each atom would occupy the closest lattice point and the value at this point would represent the type of atom. Five atoms were present in these proteins: H, C, O, N and S and they were numbered from 1-5. All points without atoms had a value of zero. Another representation we tried also used this lattice approach but different atom types were represented by different channels so the input was five 21^3 arrays. For example, the lattice for the oxygen channel would have a one in locations with an oxygen atom and zeros in all other positions. Finally, we also looked at a smaller 2D representation. Taking the first lattice representation where atoms are numbered from 1-5, we take the 3 orthographic projections, where if two atoms are present along the same axis of projection, only the "front" one is projected. This results in a representation of three 21^2 arrays. We felt this projection method would work because the original lattice was very sparse so this projection would not result in a huge loss of information.

We also encoded a mutation as a vector where every element corresponds to an amino acid. The value corresponding to the old amino acid is -1 and the value corresponding to the new amino acid is 1.

C. Network Architecture

When using input representations as three-dimensional arrays we used a relatively small network to allow for training in a reasonable amount of time. We did not try to optimize the network architecture but instead found a set of hyperparameters that gave reasonable results and allowed us to explore other aspects of the problem.

We used a CNN with 2 convolutional layers. The first layer used a 5^3 filter with a stride of 2 and a depth of 10. The second layer used a 3^3 filter with a stride of 1 and a depth of 20. The fully connected layer had 10 hidden nodes. We used a cross-entropy objective function.

When using the projected 2D input we also tried a larger network because the decreased dimensionality of the input allowed us to try a more computationally demanding model. We added a third convolutional layer with a 3×3 filter with stride 1 and a depth of 40.

D. Training

We explored different learning rates and methods of optimization but ended up settling on simple SGD optimization using a learning rate of 0.01. We used L2 regularization with an L2 constant of 0.01. We had a relatively small batch size of 10. We used an AWS p2.xlarge instance for training which generally took 4 minutes per epoch.

V. Results

In this section we discuss the results of our experiments. Since we used a relatively small batch size, our accuracy measurements throughout training were very noisy. In order to make the plots more readable we passed the data through a Savitzky-Golay filter with window size 51 and order 3 to get smoother data.

A. Resampling

As previously mentioned, the initial dataset's composition is skewed towards destabilizing or decreasing mutations. This reflects a biological reality where random changes will generally act to destabilize a protein.

We found that this skewed data resulted in a classifier that was biased towards predicting a mutation will be destabilizing. In order to address this issue we tried resampling our data by duplicating all entries that were stabilizing. This new resampled dataset contained 2510 mutations with 44.8% being increasing and 55.2% being decreasing.

Figure 2 highlights the bias in the initial data and how resampling addressed this issue. The top plot shows training and validation accuracy for models trained with the initial skewed data and resampled data respectively. Both exhibit similar training and validation accuracies. However, the bottom plot shows the classifier trained with the biased data has a much worse accuracy when the mutation it is classifying is stabilizing. This is because a model trained with data biased towards a class benefits from biasing its guesses towards that class. This also means the comparable performance the biased classifier had on overall accuracy would be a product of the validation set it was tested against being biased as well.

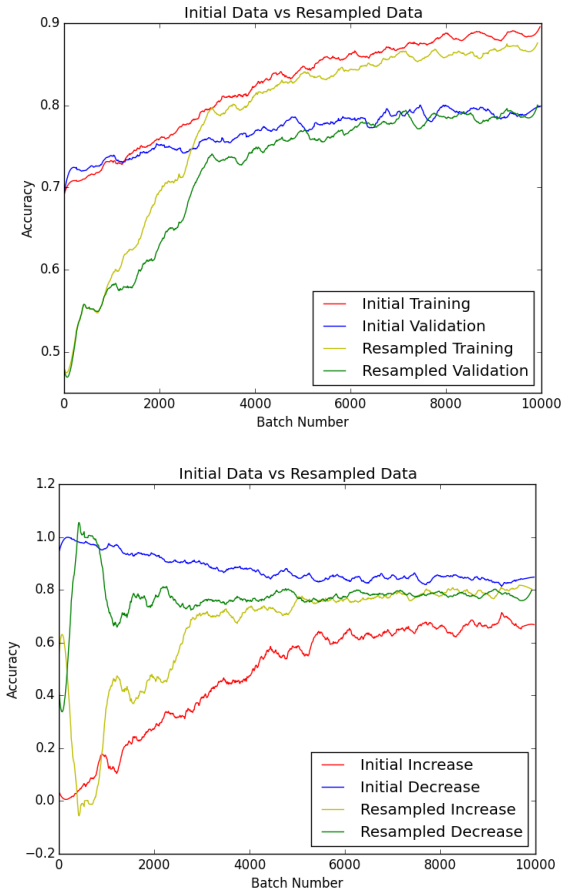

Fig. 2. *This figure shows the effects of using resampled data. The top plot shows accuracy on the training and validation sets for the initial and resampled datasets. The bottom plot shows accuracy for just increasing mutations and decreasing mutations, respectively, for models trained with the initial data and the resampled data.*

B. Learning Rate and Optimization Methods

We explored the use of different optimization methods with different learning rates to see how they impacted performance. **Figure 3** shows the results for the different optimization methods. The different methods did not show any remarkable differences. One of the methods we tried was using SGD with a learning rate schedule. **Figure 3** only shows the first half of the schedule for better comparison with the other plots in **Figure 3**, however, the results were not of note. We also tried using an Adam optimizer but this also did not result in any significant differences [8]. Not shown is our trial using the Adam optimizer with lr=0.01 which resulted in significantly worse performance with accuracies lower by 10 points. Since regular SGD with lr=0.01

marginally performed the best in terms of speed of improvement, we decided to use it for the remainder of the experiments.

C. 2D Orthographic Projection Representation

Our initial representation used 3-dimensional positional data describing the orientation of atoms within amino acids constituting the proteins. Since the CNN receives feedback from a cross-entropy loss function on what features to focus its extraction, it ultimately decides what information is important to extract. We used 3D representations at first to leverage the ability of CNNs to extract and compose proximate structural features into less sparse, more descriptive embeddings. Through inspection of the data, however, we found that majority of the 3D structures were very sparse. That is, the Euclidean representations of the proteins did not contain many atoms, and, as such, the lattice cube representation contained many zero entries relative to the number of lattice points (21^3).

Due to the sparsity of our 3D representations, we chose to project our data into 2D space. This was accomplished by observing the xy plane parallel to the z axis, and, for each 21x1 vector (21^2 vectors), projecting the first non-zero element on to a 21^2 plane. This was repeated for the yz and xz axes. The projection transforms our initial representation dimensions (x,y,z) = (21, 21, 21) to (i, j, channels) = (21, 21, 3), where i,j are arbitrary coordinates and channels are the three views of a particular i,j coordinate. The logic is that within these 21x1 vectors, there are not many collisions. So therefore, we can reshape the data representation without significant information loss. This transformation improves the runtime from $\mathcal{O}(n^3)$ to $\mathcal{O}(n^2)$, where n is the number of dimensions in our representation. Keeping the neural architecture and hyperparameters the same, this translated to an epoch run time of $\approx$ 4 minutes and $\approx$ 2.5 minutes for the 3D and 2D features, respectively. **Figure 1** We report on the 2D neural model's accuracy for different hyperparameters in **Table 2**. We also illustrate the model's convergence in Figure 4.

We achieved our best 2D results using three convolutional layers of depth 20, 20, and 40, respectively. We initially trained models with two layers, varying their depth. As expected, the models with greater depths converged on higher validation accuracies when trained with optimal hyperparameters. This is a result of the model having more free parameters to fit the data. This effect is further observed in examples 4 and 5, where we added a third convolutional layer. The highest validation accuracy in the 2D model was $\approx$ 81 %.

TABLE II. THE TABLE BELOW REPRESENTS THE NEURAL NETWORK'S ACCURACY IN CLASSIFYING THE 2D PROTEIN DATA. EXAMPLES 1-3 USED THE SAME NEURAL ARCHITECTURE AS DESCRIBED IN FIGURE 1 (2 CONVOLUTIONAL LAYERS CONNECTED TO A FULLY CONNECTED LAYER). WE ADDED A THIRD FILTER OF DEPTH 40 IN EXAMPLES 4-5.

BS	L1 Depth	L2 Depth	Train Acc.	Val. Acc.
100	10	20	0.86	0.79
30	20	20	0.88	0.80
30	10	10	0.66	0.65
30	10	20	0.89	0.78
30	20	20	0.87	0.81

D. 3D Multichannel Representation

In this section we test a different 3D representation of our structural data. Instead of combining all atom representations into a single lattice, we assign each type of atom to a channel so the input is now five lattices. This representation is described in more detail in Section IV-B.

We found that this multichannel representation outperformed our combined representation in terms of validation accuracy. **Figure 4** shows a comparison of the performance of the two representations.

This representation could be favourable because it avoids some numerical and semantic issues with the initial representation. Splitting the types of atoms across multiple lattices allows each atom to be represented as a 1. This avoids the problem of having certain atoms inherently weighted higher. For example in the initial representation a sulfur atom would contribute five times as much to the input of a node when compared to a hydrogen. This creates a bias towards hydrogen atoms. Furthermore this representation avoids the problem of implicitly defining nonsensical relationships between atom types like the sum of two carbons is a nitrogen.

Overall the 3D multichannel representation gave us our best results. **Table 3** shows a comparison of the best observed validation performance for each

TABLE III. CLASSIFICATION ACCURACY OF DIFFERENT STRUCTURE REPRESENTATIONS

Representation	Training Accuracy	Validation Accuracy
Orthographic Projection	0.88	0.80
Single Lattice	0.89	0.82
Multi-Channel Lattice	0.90	0.86

of the representation types we tested.

It is also worth making a direct comparison to a model previously reported in the literature to gain a sense for the improvements this structure representation makes. In the I-Mutant 1.0 paper [2], Capriotti et al. report a simple fully connected neural network approach that uses the same mutation encoding vector that we used as the input. They also used the exact mutation dataset we used for this work. They reported a classification accuracy of 0.74 showing that the addition of our structure representation gave up to a 12 point increase in accuracy. This provides further evidence for the effectiveness of this structure-based approach.

VI. CONCLUSION

Here we presented, to our knowledge, the first use of convolutional neural networks to generate a protein structure feature vector to be used for the prediction of thermostability changes in response to single-site residue mutations. Furthermore, to our knowledge, it is one of but a few cases where convolutional neural networks have been applied to protein structure representations.

The primary intent of this work was to explore the use of CNNs as feature extractors for protein structure representations and to show the usefulness of these features with respect to protein-related prediction problems. We explored a variety of representations of protein structure as input and found a multi-channel lattice approach worked the best. With this model we were able to classify the direction of thermostability change resulting from a given single-site residue mutation with 86% accuracy. This is comparable to the state-of-the-art accuracies for this problem, beating many values previously reported in the literature. This also makes this model one of the best non-ensemble based classifiers for this task.

We also think it is important to note that we did not put any significant effort into optimizing the hyperparameters of our model and due to technical difficulties we were using a presumably smaller and dated dataset. This means there is still a lot of room for improvement with our model, especially if ensembled with other predictors, and that it is definitely something worth exploring.

We think this work makes a strong case for the efficacy of using CNNs for protein structure related tasks. We foresee CNNs becoming a staple of future methods for the design and optimization of proteins.

REFERENCES

[1] Gromiha et al., ProTherm: Thermodynamic Database for Proteins and Mutants. *Nucleic Acids Res.*, **27**, 286-288 (1999).

[2] Capriotti et al., A neural-network-based method for predicting protein stability changes upon single point mutations. *Bioinformatics*, **20**, 63-68 (2004).

[3] Capriotti et al., I-Mutant2.0: predicting stability changes upon mutation from the protein sequence or structure. *Nucleic Acids Res.*, **33**, 306-310 (2005)

[4] Cheng et al., Prediction of protein stability changes for single-site mutations using support vector machines. *Proteins.*, **62**, 1125-1132 (2006).

[5] Masso & Vaisman, Accurate prediction of enzyme mutant activity based on a multibody statistical potential. *Bioinformatics.*, **23**, 3155-3161 (2007).

[6] Chen et al., iStable: off-the-shelf predictor integration for predicting protein stability changes. *BMC Bioinformatics*, **14**, S5 (2013).

[7] Wallach et al., AtomNet: A Deep Convolutional Neural Network for Bioactivity Prediction in Structure-based Drug Discovery (2015).

[8] Kingma Ba, ADAM: A Method for Stochastic Optimization. *ICLR* (2015)

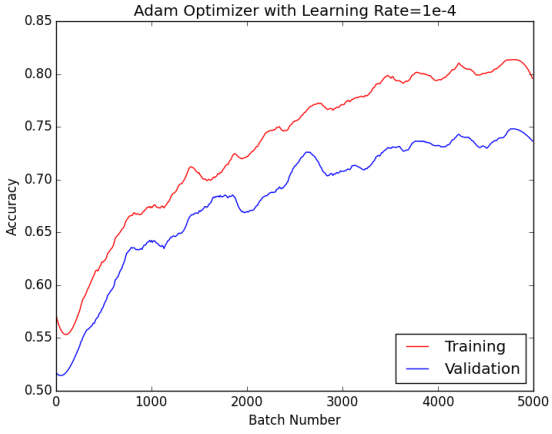

Fig. 3. *This figure shows the effects of using different optimization methods on the training and validation accuracies. The top plot used SGD with a 0.01 learning rate. The middle plot used SGD with a learning rate schedule where the first 3000 batches had lr = 0.01, the next 2000 had 0.001 and the remainder is not shown. The bottom plot used the Adam optimizer with lr = 0.0001.*

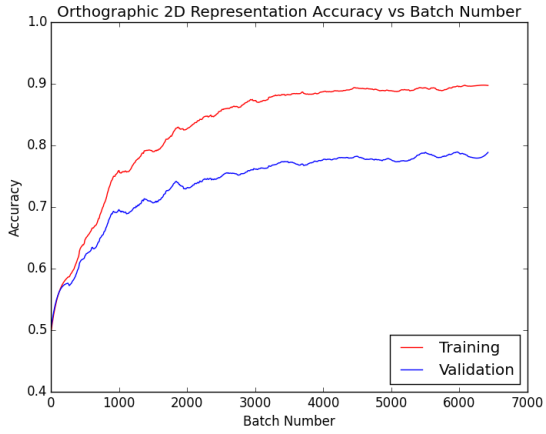

Fig. 4. *This figure illustrates the convergence of the 2-dimensional neural model on the training and validation data for example four. After approximately 3000 batches, the model converges. The neural network was optimized using Adam.*

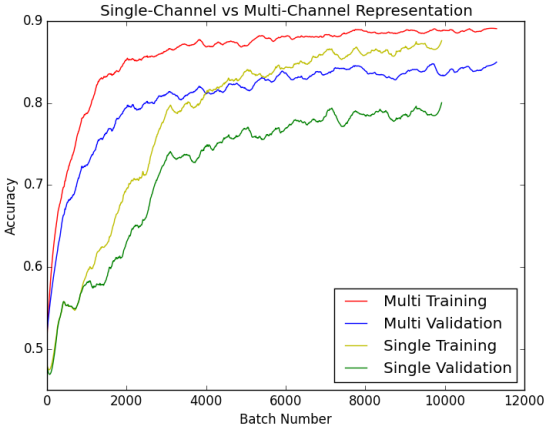

Fig. 5. *This figure shows the training and validation accuracies for the single-channel representation vs the multi-channel representation. The multi-channel representation trains faster and achieves a higher accuracy.*