Detecting 3-D Protein Architectures in 2-D Contact Maps

Examining and recognizing protein architectures takes an experienced eye and knowledge of many different folds. Because of this, many well-established protein classification groups still require manual evaluation to construct databases. Towards the automation of this process, I present a fully convolutional neural network that performs residue-wise semantic segmentation on protein contact maps. The model performs with 90.3% position-wise accuracy, 95.2% average within-class accuracy, and 87% average within-structure accuracy.

Raphael R. Eguchi

1. Introduction

Proteins are biomolecules that regulate nearly all chemical processes in living organisms. Proteins are a linked sequence of amino acids ("**residues**"), of which there are 20 different types. Proteins can be represented using **contact maps** which are pair-wise distance matricies between the α-carbons (Fig.1). Most proteins are comprised of multiple **domains** (Fig. 2), which are structural units that can each be assigned to different architecture classes.

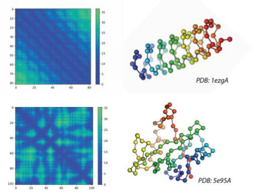

Figure 1. Protein and Corresponding Contact Maps.
Proteins colored as chainbows. Contact map in units of Å.

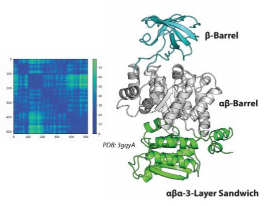

Figure 2. Example of a 3-Domain Protein.
A single-chain, 3 domain protein. Each domain is colored and annotated with architecture name.

2. Dataset

- CATH Database v4.2
- 132380 non-redundant annotated protein chains. (42 classes, 202506 domains, 3.3x10^7 residues).
- Effectively the set of all protein structures known to humans.

- Selected Dataset
- 126069 non-redundant chains (38 classes, 81753 domains, 2.9x10^7 residues.
- Length < 520 Residues, no domains belonging to classes with < 10 members.
- Spans 95% of all non-redundant chain data.
- 8000 reserved for each the development and test sets.
- Selection did not drastically alter the overall structure of the data (Fig. 3).

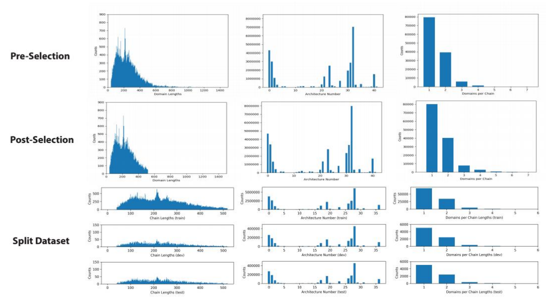

Figure 3. Comparison of Dataset Structure.
Overall the structure of the data is preserved throughout selection and splitting. The dataset is highly unbalanced, with the with the largest class containing nearly 7 million residues and the smallest containing less than 3500. Stratified splits were adjusted so all sets had at least 650 residues from each class present

3. Model Architecture and Training

- Encoding: 4x4 Convolutions, 6 Layers
- Decoding: Pixel Shuffle to 512x512 feature map → Rectangular convolutions are then used to reshape the map to a 1x512@38 → Softmax.
- Encoding layers are followed by a BatchNorm and a LeakyReLu. The 4x1 convolution is followed only by a LeakyReLu.
Dropout regularization was used throughout the encoding layers with p = 0.1. All weights were initialized using Xavier Initialization.

$$Loss(z,c) = -\frac{1}{r}\sum_{r} w_c \, log \frac{e^{z_{c,r}}}{\sum_j e^{z_{j,r}}}$$

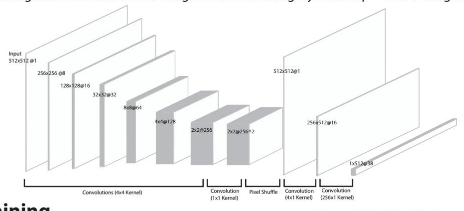

Figure 4. Model Architecture

4. Training

- Algorithm: Adam (β-values: 0.9, 0.999)
- Duration: 90 Epochs.
- Mini-batch Size: 64.
- Learning Rate: 55 epochs @ 0.001, 15 epochs @ 0.0001. 20 epochs @ 0.00001.
- Loss Function: Class-weighted cross entropy loss averaged across all residues in the input chain.
- Development Set Performance: 90% position-wise accuracy, 95% within-class accuracy.

5. Test Performance and Example Outputs

- Position-wise Accuracy: 90.3%
- Average within-class Accuracy: 95.2%
- Average within-structure Accuracy: 87.0%

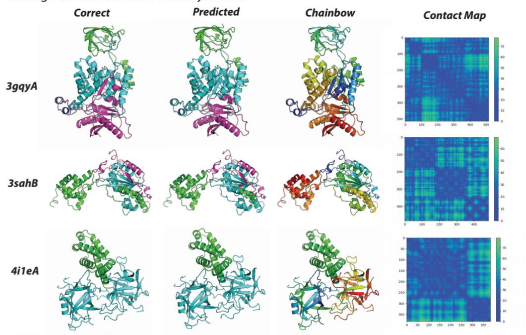

Figure 8. Example Outputs.
3gqyA. 86.9% position-wise accuracy: beta-barrel (green), alpha-beta barrel (cyan), loop (dark blue), 3-layer-αβα(magenta).
3sahB. 89.0% position-wise accuracy: orthogonal-bundle (green), 2-layer sandwich (cyan), loop (magenta).
4i1eA. 99.2% position-wise accuracy: alpha-horseshoe (green). trefoil (cyan).

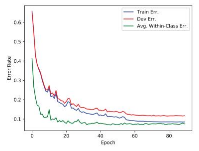

Figure 5. Error Rates During Training.
After 90 epochs of training, the model achieves a peak accuracy of 90% and an average within-class accuracy of 95% on the development set.

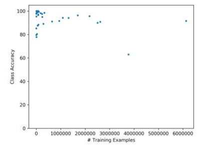

Figure 6. Class Accuracy vs Training Counts.
Each point represents a single class. No obvious correlation between the number of examples in the training set and class accuracy on the test set is observed.

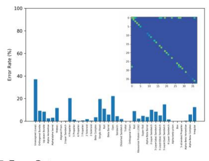

Figure 7. Error Rates.
The error rates for each classes. The confusion matrix is shown as an insert and indexed by architecture number. Green indicates a higher count. The matrix is clipped for visibility.