Predicting protein functional sites through deep graph convolutional neural networks on atomic point-clouds

David Toomer
Stanford University
djtoomer@stanford.edu

Abstract

One of the most important processes in *de novo* drug design is to determine functional sites on the protein of interest. Recent research in computer-aided drug design (CADD) has developed several algorithms that have made headway in automating this process to be completely structure-based, i.e. without any information about the physical or chemical properties of the binding ligand. The salient way these algorithms represent proteins is through discrete voxelization, but this data structure is sparse and requires significant preprocessing. This paper proposes a novel protein functional site predictor that avoids this expensive representation and instead leverages point-clouds as protein embeddings. Each protein is treated as a graph where physicochemical properties are signals on the graph. The network is a state-of-the-art graph convolutional neural network (GCNN), which has been shown to be effective when analyzing spatial information. The proposed network reaches high accuracy and precision, showing competitive results with previous methods, and it provides key insights into the structure of functional sites on enzymes.

1 Motivation

Computer-aided drug design (CADD) has become one of the most important components in targeting disease. One of the fundamental goals of CADD is to predict if, where, and how a candidate drug molecule will bind to a target protein. This project aims to approach the problem in reverse: to find suitable binding sites on proteins without prior knowledge of the ligand's molecular conformation in order to allow for more flexibility in designing novel therapeutics and to support *de novo* drug discovery.

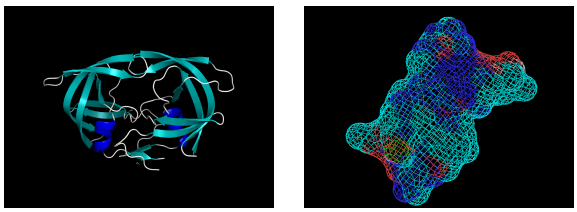
While many researchers have proposed computational methods for structure-based drug-discovery, many of these methods are either computationally expensive, or the underlying data structure is not optimal. This project aims to approach the challenge of *de novo* drug discovery from an embedding standpoint, trying to find a more natural representation for 3D molecular structure.

2 Data and Methods

2.1 Dataset

The sc-PDB protein database was used for training and testing, which is an annotated set of druggable binding sites available at `http://bioinfo-pharma.u-strasbg.fr/scPDB/` [1]. The database contains nearly 5,000 proteins and their attachments to almost 7,000 different ligands. Each protein-ligand complex contains information about every atom in the interaction and its location in 3D-space, as well as chemical features such as charge, hydrogen bonding capacity, and its associated amino acid residue. A detailed description of the .mol2 file format and what it contains can be found at `https://reference.wolfram.com/language/ref/format/MOL2.html` [2].

With a total of 16,034 protein-ligand interaction entries, the dataset was split into 12,000 entries for training and the remaining 4,034 for development/testing. However, since each protein is divided into subregions before being fed into the network, the dataset is effectively augmented to over nine million trainable examples.



(a) (b)

Figure 1: Example of a protein contained in the sc-PDB dataset (PDB ID: iodw). (a) The protein structure. (b) The mesh surface plot of the binding site cavity.

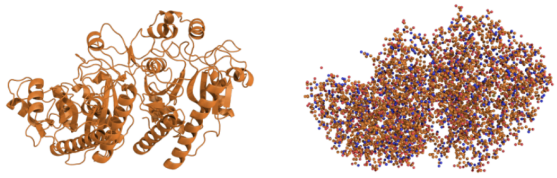
2.2 Data Representation

Several researchers have offered compelling methods for representing proteins molecules for various applications.

Amino acid encoding. One of the most popular ways of representing proteins is based off of its amino acid chain. The protein is represented as a chain of residues, each having physicochemical properties, such as charge, acidity/basicity, size, and hydrophobicity. While this has been shown to be useful in applications to determine the secondary/tertiary structure of a protein [3, 4], it neglects the positioning of the individual atoms in a residue, which is crucial for understanding how a ligand will interact with a functional site. The problem of drug discovery requires atom-scale resolution.

Voxelization. The current state-of-the-art protein representation for functional site prediction without explicit molecular simulation is voxelization. Jiang *et al*. and Altman both use a discrete atomic voxelization approach, performing calculations on both empty and protein-occupied voxels to determine subregions that are likely to be binding pockets [5, 6]. While this has been shown to be effective, it requires large amounts of pre-processing, and the data structure is very sparse. A better format for the data would capture spatial and chemical features in a more succinct way.

I propose a shift from the voxelization approach to a point-cloud representation of the protein. The point-cloud intrinsically contains spatial features of the atoms, and additional atomic properties (such as charge and polarity) can be included as features of each node. The point-cloud can be interpreted as a graph, where each feature is a signal on the graph. These features would then be fed into a deep graph convolutional neural network (GCNN) that predicts the likelihood that a ligand could bind there.



(a) (b)

Figure 2: (a) A 3D protein structure. (b) The 3D point-cloud representation of the atoms in the protein. [7].

The promise of using point-clouds for proteins is not novel, as Benhabiles *et al.* have exploited in their transfer-learning approach to protein representation for a different application [8]. However, in their model, they still default to voxelizing the point-cloud before feeding it into a network, whereas this model will learn using the raw point-cloud data.

2.3 Atomic Feature Extraction

Each atom will hold several features corresponding to its own chemical properties and the properties of its surroundings, such as its charge, nearby Van der Waals interactions, and electrostatic attraction/repulsion. A summary of the chosen features for the network are given below.

Feature	Formula (if relevant)
(x, y, z)-coords	
charge	
hydrogen bonding	
occupancy	$1 - \exp(-(r_{vdw}/r)^{12})$
Van der Waals	$\sum_{j \in \mathcal{N}_k(i)} \frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}}$
electrostatic potential	$k_e \cdot \sum_{j \in \mathcal{N}_k(i)} \frac{q_i q_j}{r_{ij}}$

Table 1: The physicochemical features used for each atom.

3 Network Architecture

The basis of DeepPCSite is a graph convolutional neural network, which leverages the principle ideas behind signal processing and applies them to graph structures. A modified version of the PointGCN architecture with global pooling was used for the classification task [9, 10, 11, 12].

A protein is treated as a general, undirected, k-nearest neighbors graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X})$, where $\mathcal{V}$ is the set of $N = |\mathcal{V}|$ vertices, $\mathcal{E}$ is the set of edges, and $\mathcal{X} \in \mathbb{R}^{N \times d}$ is the feature matrix, where d is the number of features per node. The features of each atom are its (x, y, z)-coordinates and its chemoinformatic properties, shown in Table 1. Given the coordinates of the atoms in the protein, the k-nearest neighbor graph is weighted with a Gaussian kernel,

$$\mathcal{W}_{i,j} = \begin{cases} \exp(-||x_i - x_j||^2/\sigma^2 & \text{if } j \in \mathcal{N}_k(i) \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where $\mathcal{N}_k(i)$ is the set of k-nearest neighbors of vertex i. The choice to use k-nearest neighbors for edge weighting instead of the bond affinities of the covalent and weak bonding interactions is to deliberately extract the spatial relationships between proximal non-bonded atoms.

The spectral properties of the graph, which are crucial to convolution, are computed through the graph Laplacian, which is the difference operator $L = D - \mathcal{W}$, where D is the degree matrix of the graph. For propagation, the normalized version of the Laplacian $\mathcal{L}$ is used:

$$\mathcal{L} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I_N - D^{-\frac{1}{2}} \mathcal{W} D^{-\frac{1}{2}} \quad (2)$$

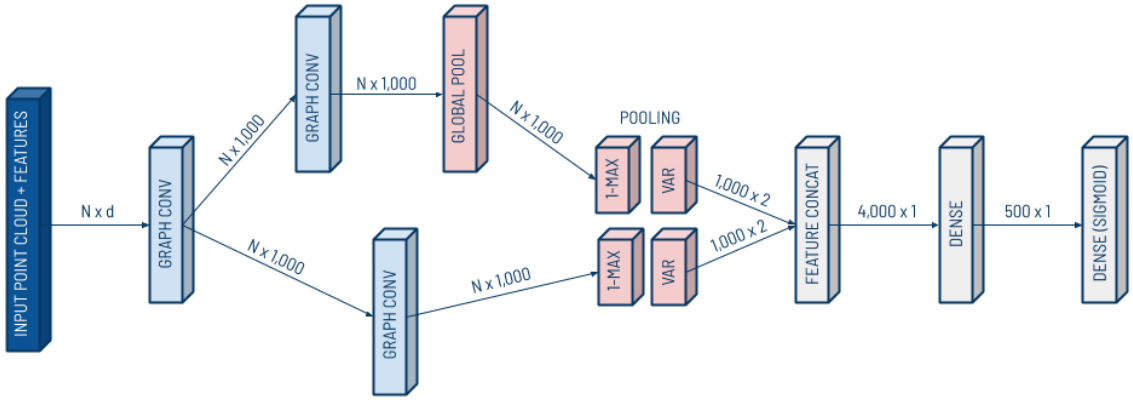


Figure 3: The PointGCN global pooling architecture, adapted to the protein site classification task. Layers in light blue are graph convolutional layers, layers in light red are pooling layers (global, 1-max, and variance), and layers in grey are either fully connected or otherwise noted. The dimensions feeding out of one layer and into another are given between the two layers.

3.1 Convolution

The convolution and pooling layers in GCNN follow a similar metaphor to traditional convolutional neural networks. For convolution, linear shift-invariant filters in the vertex domain transform a graph signal $\mathbf{x}$ to another graph signal $\mathbf{y}$ via

$$\mathbf{y} = g_\alpha(L)\mathbf{x} = \sum_{k=0}^{K} \theta_k T_k(L)\mathbf{x} \tag{3}$$

where θ_k is learned from data, and Equation 3 is the Chebyshev polynomial, which can be computed recursively as $T_0(L) = I_N$, $T_1(L) = L$, and for $k \geq 2$,

$$T_k(L) = 2LT_{k-1}(L) - T_{k-2}(L). \tag{4}$$

Each convolution layer uses these polynomials with order $k = 3$ for graph convolution, followed by ReLU activation.

3.2 Hyperparameters

Many of the network's hyperparameters aligned well with the original PointGCN global pooling architecture. Details on the hyperparameters are explored below.

Graph properties. Each protein point-cloud was divided into subregions before being fed into the network. Given the interaction data contained in sc-PDB, the average binding site radius was calculated to be between 10-15Å. Likewise, each subregion comprises a 20Å×20Å×20Å cubic volume, which aligns well with previous research [6, 9].

For each of these potential functional sites, it was determined that an insignificant number of binding sites had less than 300 atoms, but a significant number of them had more than 300 atoms, so this was considered the threshold for a valid functional site candidate. Using a threshold eliminates the need to train on a myriad of sparse regions on the protein that are likely not functional sites. Each atom in the graph was weighed with its $k = 30$ nearest neighbors.

Network hyperparameters. The hyperparameters for the network were mostly taken from the PointGCN network or from common practice. The network was trained using an Adam optimizer ($\beta_1 = 0.9$, $\beta_2 = 0.999$) using mini-batch gradient descent (mini-batch size = 28). Adjustment of the mini-batch size (28, 32, 64) led to no significant differences in training.

3.3 Regularization

Because functional sites can comprise anywhere from 10-20% of an enzyme, there were more negative examples than positive examples in the training set. This unbalanced data led to overfitting on the training set, and lower performance on positive examples (low recall). In order to offset this, inverted dropout was added after each convolution layer and each dense layer, and ℓ_2-regularization was added to the network. Additionally, the sigmoid cross-entropy terms in the loss function were weighted to account for the relative difference:

$$\mathscr{L} = -\sum_i \alpha \cdot y_i \log(\hat{y}_i) + \beta \cdot (1 - y_i) \log(1 - \hat{y}_i) \tag{5}$$

where $\alpha > \beta$ to add more weight to the positive examples.

4 Results

The protein functional site predictor was evaluated on two scales: (1) a site-level scale, to characterize accuracy in

predicting whether a subregion is a functional site, and (2) a protein-level scale, to determine if the predictor can choose the most probable region on a protein to be the functional site.

4.1 Subregion Classification

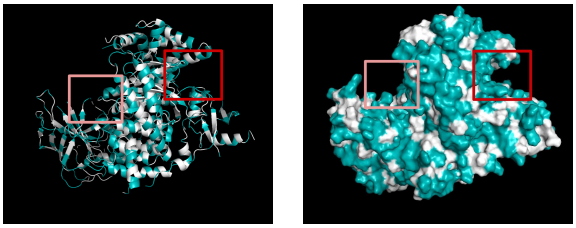
The first task, to classify subregions as functional sites or not, is used to determine which examples the network is classifying as positive or negative before evaluating it on a whole protein. The confusion matrix for the network is shown below:

$n = 2{,}278{,}142$	Predicted: True	Predicted: False
Actual: True	340,887	23,616
Actual: False	10,627	1,903,012

Table 2: Confusion matrix for the protein functional site classifier.

From the confusion matrix, the accuracy is 0.985, the precision is 0.970, and the recall is 0.935. This level of accuracy and precision is highly competitive with other protein site classifiers, such as fPocket and DeepSite, yet this is largely in part to the overwhelming number of negative examples when performing subregion classification. The recall is a more important measurement, as false negatives show the inactive regions on a protein that the classifier thinks are functional sites.

Hydrophilic surface cavities return false negatives. Analyzing many of the false negatives reveals that cavities containing polar and charged residues on the surface of a protein are frequently classified as functional sites. This is consistent with biochemistry because hydrophilic and charged residues are most common for hydrogen bonding, which is crucial for a ligand to bind. As shown in Figure 4, the falsely predicted region is similar in chemical and physical composure to an actual functional site. This result suggests that the network is learning the correct physicochemical properties of functional sites.



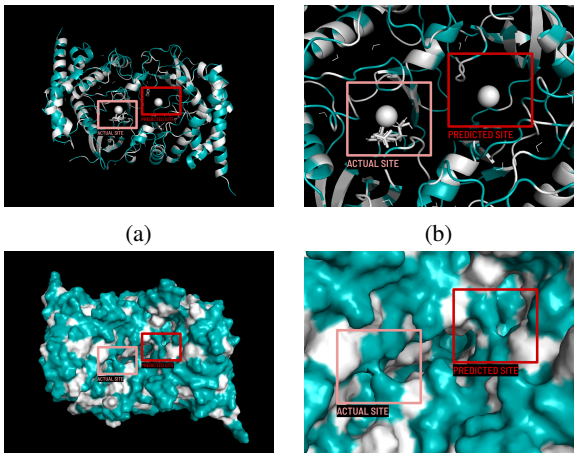
(a) (b)

Figure 4: An example of a misclassified subregion (highlighted in red), with the active site as a reference (highlighted in coral) on protein kinase 1e7v. Hydrophilic residues are shown in teal. (a) Cartoon depiction of the misclassification. (b) Surface contour plot of (a).

4.2 Functional Site Prediction

Knowing that the network identifies the proper functional regions, a larger-scope task can be performed. The more important use of the site predictor is to be able to determine the most likely functional site(s) on a given enzyme. For this task, a successful example is defined as when the center of the predicted functional site is within 10Å of the center of the actual binding site. The network achieved a 96.75% accuracy in this task, with an average error of 12.7Å from the binding site.

Errors in functional site prediction reveal putative allosteric inhibitory regions. Analysis of the predictor's mistakes shows that the error described in Section 4.1 was often between 0-10Å or much larger than that, with few in-betweens. This is to be expected, and it suggests that the network was finding the centers of similar sites on the protein rather than the outskirts of the true functional site. Many of these distal regions contained regulatory elements, such as allosteric inhibition sites, or cofactor binding sites.



(a) (b)

(c) (d)

Figure 5: Example of an incorrect prediction made by the network. The actual site is shown in coral, and the predicted site is shown in red. Hydrophilic residues are colored in teal. (a) Cartoon depiction of protein 3akk, its binding site, and the predicted site. (b) Close-up view of (a). (c) Surface contour plot of (a). (d) Surface contour plot of (b).

As shown in Figure 5, the region that was predicted is a region containing a Mg^{2+} ion cofactor, which is appropriately shrouded in a hydrophilic cavity. The actual active site has a similar chemical composure, including another Mg^{2+} ion.

Regions such as this display a greater trend of the network—incorrectly predicted regions may be susceptible to functionality as allosteric sites. These could be used in *de novo* drug design as regions for non-competitive inhibitors to bind. Further research into these sites, such as

molecular dynamics simulation, could characterize their ability to be used as functional sites for novel ligands.

5 Conclusion

Using point-clouds as protein embeddings has shown to be effective for the task of protein functional site prediction. The protein site classifier performs similarly well to other classifiers, maintaining a high accuracy and precision.

Future research would focus on improving recall. These efforts would aim to prevent the network from misclassifying non-functional hydrophilic regions, such as incorporating more spatial features. Research into the latent space of the GCNN could also reveal more information about what the network considers the most important, which would help further reduce the size of the input by lessening the number of chemical features on the graph. Additionally, because the kNN graph structure is rotation- and shift-invariant, researching methods of data augmentation in order to equilibrate positive and negative data would be useful to improve recall.

6 Contributions

All portions of the project were contributed by the author.

References

[1] J. Desaphy, G. Bret, D. Rognan, and E. Kellenberger. sc-pdb: a 3d-database of ligandable binding sites—10 years on. *Oxford Academic*, 2014.

[2] Wolfram Language and System Documentation Center. Mol2.

[3] K. Yang, Z. Wu, and F. Arnold. Machine-learning-guided directed evolution for protein engineering. *Nature Methods*, 16:687–694, 2019.

[4] F. Noé, G. De Fabritis, and C. Clementi. Machine learning for protein folding and dynamics. *Elsevier, Current Opinion in Structural Biology*, 60:77–84, 2020.

[5] M. Jiang, Z. Li, Y. Bian, and Z. Wei. A novel protein descriptor for the prediction of drug binding sites. *BMC Bioinformatics*, 20(478):1–13, 2019.

[6] W. Torng and R. Altman. High precision protein functional site detection using 3d convolutional neural networks. *International Society for Computational Biology*, 35(9):1503–1512, 2019.

[7] R. Townshend, R. Bedi, P. Suriana, and R. Dror. End-to-end learning on protein structure for interface prediction. *arXiv*, pages 1–8, 2019.

[8] H. Benhabiles, K. Hammoudi, F. Windal, M. Melkemi, and A. Cabani. A transfer learning exploited for indexing protein structures from 3d point clouds. *Processing and Analysis of Biomedical Information*, 11379, 2019.

[9] Y. Zhang and M. Rabbat. A graph-cnn for 3d point cloud classification. *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2018.

[10] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

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

[12] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.