PRODYN0: INFERRING CALPONIN DOMAIN STRETCHING BEHAVIOR USING GRAPH NEURAL NETWORKS

Ali Madani, Cyna Shirazinejad, Hengameh Shams, Mohammad Mofrad
Molecular Cell Biomechanics Laboratory
University of California, Berkeley
Berkeley, CA 94720, USA
{madani,mofrad}@berkeley.edu

ABSTRACT

Graph neural networks are a quickly emerging field for non-Euclidean data that leverage the inherent graphical structure to predict node, edge, and global-level properties of a system. Protein properties can not easily be understood as a simple sum of their parts (i.e. amino acids), therefore, understanding their dynamical properties in the context of graphs is attractive for revealing how perturbations to their structure can affect their global function. To tackle this problem, we generate a database of 2020 mutated calponin homology (CH) domains undergoing large-scale separation in molecular dynamics. To predict the mechanosensitive force response, we develop neural message passing networks and residual gated graph convnets which predict the protein dependent force separation at 86.63 percent, 81.59 $kJmol^{-1}nm^{-1}$ MAE, 76.99 $psec$ MAE for force mode classification, max force magnitude, max force time respectively– significantly better than non-graph-based deep learning techniques. Towards uniting geometric learning techniques and biophysical observables, we premiere our simulation database as a benchmark dataset for further development/evaluation of graph neural network architectures.[1]

1 INTRODUCTION

The primary sequence of a well-ordered protein encodes its structural information as well as its functional properties. Proteins undergo dynamic shifts in conformation to perform their designated tasks. The growth of computational resources and simulation algorithms allow for studying molecular trajectories at atomic resolution commonplace. However, because of the heterogeneity of biomolecules and their chemical environments, computational models can suffer in performance when tasked with predicting molecular conformations, physical and chemical properties, and protein-protein interaction networks. Along with the growing excitement in machine learning in tangent fields, there is emerging interest in data-driven approaches to studying biological function at scales ranging from biochemical properties to tissues mechanics (Glazer et al. (2008); Botu & Ramprasad (2015); Schütt et al. (2017); Gastegger et al. (2017); Madani et al. (2018); Wu et al. (2018); Ramakrishnan et al. (2015)).

Mechanosensitive proteins are central to a plethora of biological phenomena (Galkin et al. (2010)). Our goal in this paper is to bridge the primary sequence of proteins to their intrinsic force observables while undergoing large-scale conformational changes. To accomplish this, we generated a molecular dynamics database of trajectories of two calponin homology domain mutants being pulling apart [Figure 2] by Steered Molecular Dynamics (SMD) (Israelewitz et al. (2001)). SMD is a technique that allows for biasing the interaction potential between explicit regions in a simulation; in our case, two globular domains, CH1 and CH2, are gradually pulled apart during the simulation. While these simulations are computationally expensive and require careful consideration for proper physical dynamics, they provide information at the atomic resolutions about the sensitivity of the roles mutations play in response to mechanical loads in proteins. Our aim to cast protein structures as graphical models allows us to take the first imperative steps towards building relationships between sequences and physical observables that can be computed with more expensive techniques such as molecular dynamics. We show that the force between the two CH domains, a global property over a long simulated trajectory, can accurately be predicted using graph neural networks.

[1] https://github.com/a-mad/prodyn

2 METHODS

2.1 GENERATING A DATABASE OF POINT-MUTATED PROTEIN DYNAMICS

To generate models to predict protein dynamics, we established the first ProDyn dataset with high-throughput *in silico* mutagenesis experiments of proteins undergoing large-scale conformational changes. The generated trajectories span 2020 unique point-mutations along two calponin homology domains, CH1 and CH2, connected by a tether that comprise a total of 224 residues. A full description of the molecular dynamics methods can be found in Supplementary Materials. The data per simulation was processed to yield a graphical input representation of residue interactions in terms of node and edge features, in addition to an output representation for force profile classification and regression. We then construct and compare conventional and graph neural network architectures to predict the force characteristics of our biophysical system.

2.2 DATA PRE-PROCESSING

A molecular dynamics (MD) simulation under a given force field outputs a trajectory file that includes type, position, and velocity information for every atom in the system for each time step. There are several prediction problems of interest that can be encapsulated by our data. In this study, our specific objective is to predict the force characteristics for a given protein initial state (i.e. residue information and interactions pre-steering).

The input representation is a directed graph, G, with node features, x_v and edge features, e_{vw}. As shown in Table 1, node features comprise of the residue type, residue properties, and initial secondary structure type as calculated by mdtraj (McGibbon et al. (2015)). Edge features currently consist of pair-wise dihedral angles, Kabsch-Sander (K-S) hydrogen bonds (Kabsch & Sander (1983)), and the initial distance between steered and fixed residues.

The output representation, a graph-level target of protein mechanosensitivity, is of two types: regression and classification. Each SMD simulation exhibits a characteristic force response over time of the simulation. The regression target is a 2 dimensional vector encoding maximum pulling force magnitude and maximum pulling force time-point respectively. The classification target is a one-hot encoded 2-dimensional vector that encodes a force mode category. The force mode categories were ascertained via spectral k-means (k=2) clustering on the force-response graphs in a lower dimensional t-SNE (Maaten & Hinton (2008)) derived space as shown in Figure 2. There emerged two clearly well-separated clusters which we utilized as categories for graph-level classification. Lastly, all continous data was scaled to [0,1], and the entire dataset was split randomly into a 404 sample held-out test set and 1616 sample 80-20 training/validation split. For the classification task, we compute the accuracy and macro-averaged F1-score as performance metrics. For the regression task, we calculate the mean absolute error (MAE) for maximum force magnitude and maximum force time-point.

Table 1 - Input Representation Features

Feature	Node/Edge	Type	Index	Examples
Residue Type	Node	One-hot	$[0:20]$	ALA, GSP, ALY, ...
Residue Secondary Structure	Node	One-hot	$[21:26]$	alpha-helix, turn, ...
Residue Properties	Node	Multi-label	$[27:43]$	acidic, polar, ...
Dihedral-ϕ	Edge	Continuous	$[0]$	$0-2\pi$
Dihedral-ψ	Edge	Continuous	$[1]$	$0-2\pi$
Dihedral-ω	Edge	Continuous	$[2]$	$0-2\pi$
Hydrogen Bond	Edge	Continuous	$[3]$	K-S energy
Steer-to-Fixed Residue Dist	Edge	Continuous	$[4]$	average distance

2.3 BASELINE CONVENTIONAL DEEP LEARNING MODELS

As a comparison to graph-based neural networks described below, conventional multi-layer perceptron (MLP) and gated recurrent unit (GRU) were trained (Cho et al. (2014)). The graph node features x_v were provided as input either by concatenation or sequentially. For the MLP architectures with one vector input of concatenated features, the number of layers [1-5], activation functions [ReLU, eLU], nodes per layer [128-512], dropout regularization [0.0-0.5], and loss functions [L1, L2, log-

cosh, cross-entropy] were experimented with. For the GRU+MLP architecture with inputs features passed sequentially through the GRU, we experimented with different GRU implementations- varying hidden dimensions [16-64], stacking [1-2], bi-directionality, regularization [via dropout], and outputs [last layer depth-wise vs time-wise].

2.4 NEURAL MESSAGE PASSING NETWORK MODEL

In line with Gilmer et al. (2017), we developed various neural message passing network (MPNet) architectures. Our models learn to compute a function of the entire graph by learning features, invariant to graph isomorphisms, through a message passing algorithm. We define the hidden state of each node in the graph by h_v^t with $h_v^0 = x_v$ where v denotes residue/node number and t denotes the cycle number, and the directed edge from node w to v as $e_{v \leftarrow w}$. The message passing scheme runs for a total of T cycles which can be interpreted as the message-passing neighborhood size. During each cycle, messages are passed between a neighboring node and the current node based on the edge direction. There is a unique message function, M, that computes the message given the current node features, neighboring node features, and edge features which is then aggregated into one message update per node, m_v^t. This message is then used to update the hidden node state by the function, U. The final hidden states of all nodes are then passed through a readout function, R, which yields an output layer of the dimension of the sample target. To formalize, for each message passing cycle the node states are updated in the following fashion:

$$m_v^t = \frac{1}{|N(v)|} \sum_{w \in N(v)} M(h_v^t, h_w^t, e_{v \leftarrow w}) \tag{1}$$

$$h_v^{t+1} = U(h_v^t, m_v^t) = h_v^t + m_v^t \tag{2}$$

where $N(v)$ denotes the neighbors of v in graph G. The message function, M, is a 3-layer MLP with 0.4 dropout, 256 hidden layer dimensions, and output layer of the same dimension as node features. After T cycles, the readout function R is applied as follows:

$$\hat{y} = R(\{h_v^T | v \in G\}) \tag{3}$$

We did not see a significant increase in performance for increasing T cycles and present our MP models for T=1 in Results. For the choice of R, we experimented with a max-pooling operation over the feature dimension and also feeding the nodes in primary sequence order to a GRU. The final output time-wise of the GRU is fed to a 2-layer MLP with 0.5 dropout. The final loss function, $L(\hat{y}, y)$ was either a L1-norm for regression or softmax+cross-entropy for classification. Models were trained using stochastic gradient descent with ADAM (Kingma & Ba (2014)) and batch size of 25.

2.5 RESIDUAL GATED GRAPH CONVNET MODEL

Residual Gated Graph ConvNets (RGGCN) is a method introduced by Bresson & Laurent (2017) that combines the vanilla graph ConvNet (Sukhbaatar et al. (2016)) and the edge gating mechanism (Marcheggiani & Titov (2017)) with residual connections for problems involving variable graphs. We extend this method by incorporating edge features, such that each graph convolution layer follows the layer-wise propagation rule:

$$h_i^{\ell+1} = \mathrm{ReLU}\left(U^\ell h_i^\ell + \sum_{j \to i} \eta_{j \to i} \odot (V^\ell h_j^\ell + D^\ell e_{j \to i})\right)$$

where h_i^ℓ denote features of node i at layer l, $e_{j \to i}$ denote features from edge $j \to i$, and edge gates $\eta_{j \to i} = \sigma\left(A^\ell h_i^\ell + B^\ell h_j^\ell + C^\ell e_{j \to i}\right)$. U, V, A, B, C and D are learnable parameters.

In our experiments, we used a 4-layer RGGCN and applied max pooling as our readout function. For classification, our model had 50 hidden dimensions and a single linear layer. For regression, our model had 256 hidden dimensions with a 2-layer MLP. Batch normalization was employed, as with residual connections between successive convolution layers.

Our models were initialized using Glorot initialization (Glorot & Bengio (2010)). We used the Adam SGD optimizer with an initial learning rate of 0.0005, and computed loss using binary cross-entropy for classification, and L1 for regression. Dropout rate was set to 0.4 for both node features and edge gates in classification, and only for the 2-layer MLP in regression.

3 RESULTS AND DISCUSSION

One of our main achievements is creating a database that is uniquely well-positioned to benchmark the rapid development and evaluation of novel graph neural network architectures on both a graph-level classification and regression task. As shown in Table 2, our baseline non-graph-based deep learning models– both (1) variations of concatenating features and processing through an MLP and (2) sequentially feeding primary sequence-ordered nodes through a GRU – were unable to learn effectively and predicted tightly near the majority/mean target. Within our data size scale, the complexity of many biophysical phenomena, through initial states and short/long-range bonded and non-bonded interactions partially captured in this study, is too difficult to train effectively unconstrained via an MLP or recurrently via a GRU.

Table 2 - Prediction Performance of Various Architectures

Model	Accuracy	F1-Score	Force$_{mag}$ MAE	Force$_{time}$ MAE
Conventional	70.54	0.414	89.68	111.39
MPNet + Maxpool	77.09	0.673	85.39	92.44
MPNet + GRU	**86.63**	0.839	**81.59**	**76.99**
RGGCN + Maxpool	**86.39**	**0.904**	89.00	100.52

Our goal was to experiment with 2 types of graphical neural networks: our own designed MPNet and a state-of-the-art graph technique such as the RGGCN. Both graph neural network architectures, MPNet and RGGCN, capture the graphical structure, information communication, and structural invariances to learn an effective node embedding for classification/regression tasks. In end-to-end training after node embedding, a readout function is applied that is agnostic to the number of nodes/residues. Among functions invariant to graph isomorphism, we observed that maxpool outperformed average pool. As proteins have multiple hierarchies of structure, we can view the MPNet with GRU readout as encoding the protein structure hierarchy with the GRU applied after message passing cycle, as to allow message passing interactions to be communicated between nodes before the recurrent primary structure network. The choice of GRU readout is likely more effective for protein-like graph systems; the RGGCN+Maxpool would be more generalizable otherwise. As shown in Table 2, all graph neural networks significantly outperform conventional deep learning techniques. The RGGCN+Maxpool and MPNet+GRU perform well on the classification task at around 86%. For the regression task, the MPNet+GRU is able to learn with a MAE of 81.59 $kJmol^{-1}nm^{-1}$ MAE on max force magnitude prediction and 76.99 $psec$ MAE on max force time prediction. To put into context the performance to the natural stochasticity of the MD system, we ran duplicate SMD simulations and observed the mean standard deviation of the max force mag was 57.94. Also, in Figure 1, we show the well-behaved convergence of the classification model training in addition to the resemblant output statistics of the true and predicted targets for regression.

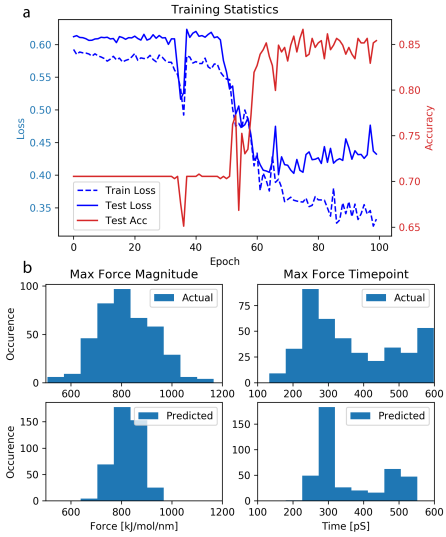

Figure 1: MPNet + GRU Performance. a) Classification training statistics b) Regression histograms of predicted vs actual values

Future work can include learning edge states in addition to node states, set2vec as readout function (Vinyals et al. (2015)) or GRU as an update function, and experimenting with unsupervised graph embedding techniques such as Veličković et al. (2018). From the biophysical perspective, we can expand the choice of node/edge features, try different protein systems, or formulate a plethora of pertinent biophysical prediction tasks.

To conclude, the progress of geometric deep learning techniques is inextricably linked to the quality of benchmark datasets– analagous to the impact of benchmark datasets for vision and language. We view our work as laying the groundwork for the intersection of graph neural network and biophysics researchers for the advancement of both fields.

REFERENCES

V. Botu and R. Ramprasad. Learning scheme to predict atomic forces and accelerate materials simulations. *Phys. Rev. B*, 92:094306, Sep 2015. doi: 10.1103/PhysRevB.92.094306. URL https://link.aps.org/doi/10.1103/PhysRevB.92.094306.

Xavier Bresson and Thomas Laurent. Residual gated graph convnets. *arXiv preprint arXiv:1711.07553*, 2017.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Vitold E Galkin, Albina Orlova, Anita Salmazo, Kristina Djinovic-Carugo, and Edward H Egelman. Opening of tandem calponin homology domains regulates their affinity for f-actin. *Nature structural & molecular biology*, 17(5):614, 2010.

Michael Gastegger, Jörg Behler, and Philipp Marquetand. Machine learning molecular dynamics for the simulation of infrared spectra. *Chemical science*, 8(10):6924–6935, 2017.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1263–1272. JMLR. org, 2017.

Dariya S Glazer, Randall J Radmer, and Russ B Altman. Combining molecular dynamics and machine learning to improve protein function recognition. In *Biocomputing 2008*, pp. 332–343. World Scientific, 2008.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.

Barry Isralewitz, Mu Gao, and Klaus Schulten. Steered molecular dynamics and mechanical functions of proteins. *Current opinion in structural biology*, 11(2):224–230, 2001.

Wolfgang Kabsch and Christian Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers: Original Research on Biomolecules*, 22(12):2577–2637, 1983.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Ali Madani, Jia Rui Ong, Anshul Tibrewal, and Mohammad RK Mofrad. Deep echocardiography: data-efficient supervised and semi-supervised deep learning towards automated diagnosis of cardiac disease. *npj Digital Medicine*, 1(1):59, 2018.

Diego Marcheggiani and Ivan Titov. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv preprint arXiv:1703.04826*, 2017.

Robert T. McGibbon, Kyle A. Beauchamp, Matthew P. Harrigan, Christoph Klein, Jason M. Swails, Carlos X. Hernández, Christian R. Schwantes, Lee-Ping Wang, Thomas J. Lane, and Vijay S. Pande. Mdtraj: A modern open library for the analysis of molecular dynamics trajectories. *Biophysical Journal*, 109(8):1528 – 1532, 2015. doi: 10.1016/j.bpj.2015.08.015.

Raghunathan Ramakrishnan, Mia Hartmann, Enrico Tapavicza, and O Anatole Von Lilienfeld. Electronic spectra from tddft and machine learning in chemical space. *The Journal of chemical physics*, 143(8):084111, 2015.

Kristof T Schütt, Farhad Arbabzadah, Stefan Chmiela, Klaus R Müller, and Alexandre Tkatchenko. Quantum-chemical insights from deep tensor neural networks. *Nature communications*, 8:13890, 2017.

Sainbayar Sukhbaatar, Rob Fergus, et al. Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems*, pp. 2244–2252, 2016.

Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. *arXiv preprint arXiv:1809.10341*, 2018.

Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. Order matters: Sequence to sequence for sets. *arXiv preprint arXiv:1511.06391*, 2015.

Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.