On the Development of a Reward-Conditioned Protein Engineering Framework

Minji Lee

Abstract

Enhancing the properties of a protein, or protein engineering is crucial for pharmaceutical and industrial usages. However, recent literature focus on sequence-based protein representations and are only modeled to single protein family as their proposed method require training with data for a specific protein function. To tackle these problems, we propose and investigate a reinforcement learning-based protein engineering framework that defines the action as a mutation of a amino acid in a protein sequence. We show the preliminary results of a surrogate reward network that model the functionality of proteins using a sequence-based representation of a protein. We also discuss the next steps regarding the implementation of our protein engineering framework.

1. Introduction

Proteins are complex molecules responsible for different functions in the human body. Proteins are also an important part of products in the food, pharmaceutical, and chemical industries. Protein engineering, or improving the functions and properties of proteins, is critical for its effective usage in various applications (Fig. 1). Due to its significance, extensive researches [2–4, 13] are held regarding deep-learning based protein engineering. However, most of these researches rely on models trained with data of a specific protein family, in a way that they are applicable only to a single protein engineering problem. Treating the engineering problems individually, *i.e.* designing models for each protein family independently, is inefficient and is not significantly faster than handcrafted protein engineering.

Reinforcement learning (RL) based protein engineering can be a way to overcome part of these inefficiencies by having a unique framework that is guided by suitable objective functions, dependent on the protein engineering goal. Similar RL frameworks and algorithms have been applied to *games* that have the same properties, so, we hypothesize that a shared RL framework might be able to design different engineering objectives, for different protein families. Until now, not many works have been done regarding RL-based protein engineering. DynaPPO [2] achieved state-of-the-art performance in several proteins and DNA engineering benchmarks using their novel variant of proximal policy optimization [16], but is only applicable to short (less than 50 amino acids) sequences since it defines the problem as generating one amino acid at each time step until the end of the sequence. Knowing that the sequence length of a single-chain antibody chain is around 110 [8] and proteins usually contain hundreds of amino acids, it is hard to apply DynaPPO to real-world protein engineering tasks.

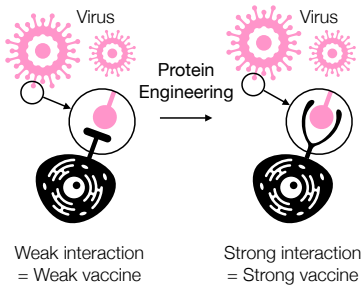

Figure 1. Protein engineering is critical for human life

In this paper, we propose, to the best of our knowledge, the first reward-conditioned reinforcement learning-based protein engineering framework that defines the action at one time step as a mutation, *i.e.* substituting one amino acid, in the protein sequence, thus applicable to most of the existing protein and antibody engineering tasks. Also, we show that a structure-based protein representation, obtained using deep learning-based structure prediction networks, is competitive with existing sequence-based representations for protein functionality prediction and can accurately guide protein engineering. Furthermore, we will explore the possibility of training a single shared protein *designer* agent that can be applied to multiple protein engineering tasks.

2. Related Works

2.1. Protein structure prediction

Recently, breakthrough results were obtained using deep neural network architectures for the protein structure prediction problem. The trRosetta network proposed in [18] considers structure prediction as a classification problem of

Codes are available at github.com/protein-dscig/design.

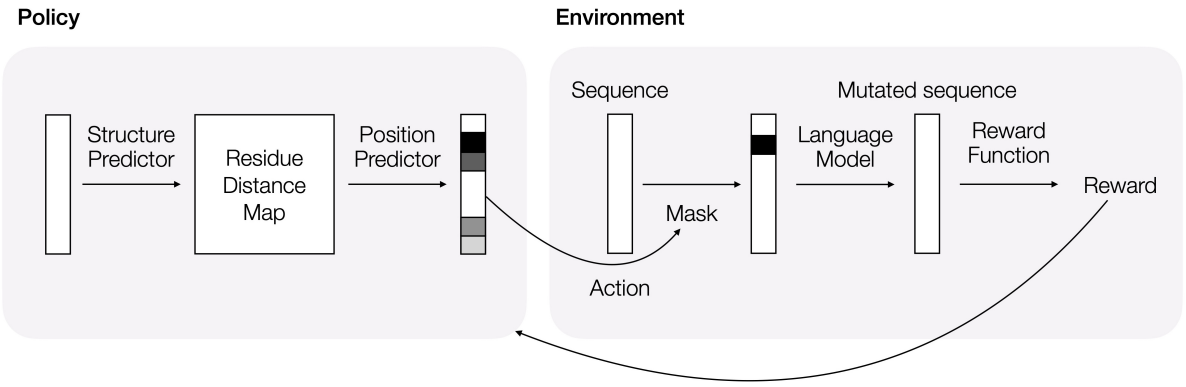

Figure 2. Overview of the proposed protein engineering framework.

four features: $C_\beta - C_\beta$ distance between two amino acids, *i.e.* $C_\beta - C_\beta$ distance, the rotation along the virtual axis connecting the C_β atoms of two amino acids, *i.e.*, ω dihedral, and two angles specifying the direction of the C_β atom of one amino acid in a reference frame centered on another amino acid. As another view on this problem, AlphaFold [9] views structure prediction as a graph inference problem. The evolutionary representation of the protein is processed through a graph attention module named Evoformer that is repeated along the architecture, resulting in the final predicted features, such as pairwise distance representations, that guide the prediction of the final structure by a structure module.

2.2. Protein representation learning

Since proteins have evolutionary related sequences, previous studies investigated the protein space by attempting to decipher their language: the amino acid sequence. Sequence-based methods [1, 4, 13, 19] typically train a feature extractor to generate continuous, fixed-size embeddings from the discrete, variable-size amino acid sequence, where the feature extractor usually consists of complex and state-of-the-art natural language processing models such as transformers. Recently, Yang *et al.* [13], proposed a network architecture composed of a series of dilated convolutions and 1-dimension convolutions based on ByteNet [10] that can compete with transformers [17] while being computationally more efficient.

2.3. Protein functionality prediction

Proteins can have different types of representations. These representations are used as input features to train machine learning algorithms that aim to predict important protein functions. Even though sequence-based methods can extract important features of the protein space, they do not take the protein structure into account. There have been recent studies that investigates the capacities of large protein language models to convey structure properties [13, 19] but, explicitly, structures are not taken into consideration during the training process. The protein structure in 3-dimensional space of its sequence of amino acids is closely related to its function. Recently, the high accuracy obtained by structure prediction networks, such as AlphaFold [9] and RosettaFold [18], opened the possibility of using these methods as tools for different applications, including the investigation of structure-based representations for predicting functionalities important for protein engineering tasks.

2.4. Protein engineering

Protein engineering is accomplished by introducing a series of mutations from a wild-type amino acid sequence, *i.e.*, the protein to be optimized. The engineering process is guided by an objective function that predicts the desired function of the protein. Biswas *et al.* [4] introduced a protein engineering framework that used a sequence-based representation model called eUniRep to guide evolution via a greedy algorithm that is modified to allow exploration. Angermueller *et al.* [2] introduced a model-based reinforcement learning policy optimization method to engineer protein sequences.

3. Methodology

In this section, we present the details of our proposed RL-based protein engineering framework (Fig. 2). The framework consists of a policy which predicts the position to mutate and an environment that predicts the functionality of the mutated sequence, that is then is used as the reward to guide the optimization process.

3.1. Markov decision process

Let $x \in \mathrm{A}^L$ be an amino acid sequence of length L over the set of amino acids A. We formulate the protein engineering problem of a sequence x as a Markov decision process $\mathcal{M} = (S, A, p, r)$ were S is the state space, A is the action space, p is the state transition function, and r is a re-

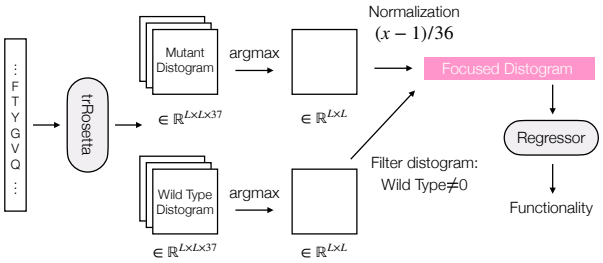

Figure 3. Procedure to obtain focused distogram of a protein.

ward function. The state space S is the set of all possible sequences of length L, i.e. all the possible proteins with length L. The state s_t at time t is defined as the current sequence x. The action space A corresponds to the set of positions $P = \{1, \cdots, L\}$ that can be mutated. The action a_t is defined as choosing one of the positions of s_t. The state transition function $p(s_{t+1}|s_t)$ is deterministic and corresponds to mutating amino acid of s_t at position a_t to the output of a protein language model, modeled by a neural network, defined as $m_{s_t}(i) = q$. $m_{s_t}(i)$ is the output of the language model of sequence s_t masked at position i and q is the amino acid given as output by the language model. The reward function $r(s_t) = f(n(s_t))$ where f is mapping function from sequence to protein functionality modeled by a neural network. In the reward function, $n(x)$ is a structure prediction network, such as AlphaFold, and the details of this formulation will be detailed in Sec. 3.3.

3.2. Policy optimization

The policy network π take as an input the current state s_t, i.e. current amino acid sequence. First, the current state passes to a structure prediction network $n(x)$ and its output is the input to another neural network that predicts the position to be mutated. We train a policy $\pi_\theta(a_t|s_t)$ to optimize the expected sum of rewards:

$$\hat{\mathbb{E}}_t = \sum_{s_t} \sum_{a_t} \pi_\theta(a_t|s_t) r(s_t|a_t) \tag{1}$$

We use proximal policy optimization [16] to optimize the objective.

3.3. Surrogate reward function

The reward function, representing the protein functionality prediction, is modeled using a structure-based representation of the current mutant sequence as its input. The protein structure for a target amino acid sequence is predicted using a deep learning-based structure prediction network. In this paper, we use trRosetta as the main structure prediction network. Our protein representation is focused on the distance probability distribution, or *distogram*, of the mutant sequence when compared to the distance probability distribution of the wild type sequence. As distances are discretized into discrete bins in trRosetta, the argument max of the distogram is taken to obtain the bins with the maximum predicted probability. Protein distograms are symmetric and usually sparse. In order to reduce the sparsity of the final feature vector, the distograms are filtered to the pixels in which the wild-type distogram is non zero, *i.e.*, the amino acid pair is closer than 20Å. The formalization of this process is as follows. For a mutant distogram $M_{i,j}$ and wild-type distogram $W_{i,j}$ where $i, j \in \{1, \cdots, L\}$, the proposed set of normalized features F is defined as $F = \{(M_{i,j} - 1)/36 | W_{i,j} \neq 0\}$. The normalization is performed to transform the discrete bins representing distances from 2Å to 20Å to values from 0 to 1, and distances larger than 20Å to a negative value. For the rest of this paper, the feature vector is called *focused distogram*, and the unfiltered feature vector is called *full distogram*. The procedure to obtain focused distogram is illustrated in Fig. 3. The distogram representation of protein structure is used to train a machine learning algorithm or neural network to predict the functionality as an output.

4. Experimental results

4.1. Surrogate reward function modeling

4.1.1 Experimental settings

We test the proposed framework to model our surrogate reward function for predicting the fluorescence activity of the green fluorescent protein of *Aequorea victoria* (avGFP). A surrogate reward function model is trained on the dataset proposed by Sarkisyan *et al.* [15] which is comprised of 54045 sequences of avGFP mutants paired with their fluorescence value. The dataset is randomly split into train (90%) and test (10%). Smaller splits (10000 and 500 samples) are also sampled from the train set for experiments. Distograms for the wild type and all mutants are collected using trRosetta. Five protein representations, two structure-based and three sequence-based, are used for comparison: focused distogram, full distogram, one-hot encoding, UniRep [1], and UniRep64 [1]. UniRep and Unirep64 differ on the size of the final generated feature vector. Using each representation as input, a ridge regressor is trained to predict fluorescence as an ablation study to confirm the superior performance of the structure-based representation for protein functionality prediction.

4.1.2 Results

As shown in Tab. 1, the distogram representations achieved the smallest mean-squared error (MSE) for all datasets when compared to sequence-based methods. For the larger train sets, the full distogram achieved the best MSE while for the smallest train set, the focused distogram achieved the

Representation	No. features	Train data size		
		Full	10000	500
Focused distogram	1443	0.4722	0.5488	**0.8049**
Full distogram	56169	**0.4556**	**0.5295**	1.0022
One-hot	4740	1.1171	1.1172	1.1190
UniRep [1]	1900	1.32‡	0.6858†	0.8673†
UniRep64 [1]	64	1.0826	1.0889	1.1267

Table 1. Comparison of mean squared error between representations. All representations are tested on same test set except †,‡. For †, 5-fold cross-validation results are shown due to long inference time of UniRep. For ‡, regression method is linear regression and we compared with the result from [12] which used the same dataset but with a different split. Best results on each dataset are **highlighted.**

Regression methods	Data size		
	Full	10000	500
LightGBM [11]	**0.3017**	**0.3511**	**0.6467**
Ridge [7]	0.4722	0.5488	0.8049
ElasticNet [20]	1.1171	1.1171	1.1173

Table 2. Comparison of mean squared error between different regression methods. All representations are tested on same test set.

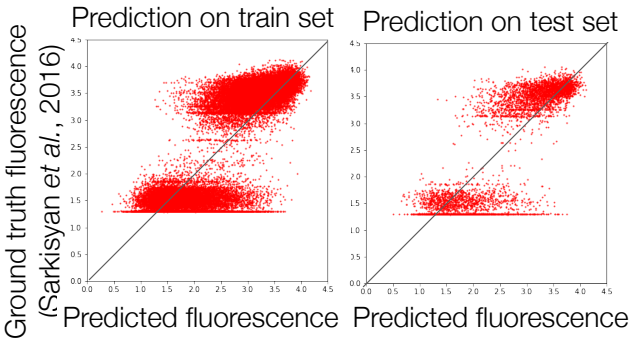

Figure 4. Fluorescence prediction on train and test set. LightGBM is trained on full train set of focused distogram representation.

best MSE. It is important to observe that the focused distogram showed competitive results when compared to the full distogram with only 2.6% of the number of features. By filtering amino acid pairs that interact in the wild-type protein, the surrogate reward network was able to focus on important interactions. This observation is also important to decide structure-based representation alternatives to a distogram when modeling a functionality predictor. Next, as detailed in Tab. 2 and visually shown in Fig. 4, the best results were obtained by using a LightGBM regressor. LightGBM achieved the smallest MSE for all dataset splits. The best value of approximately 0.30 is better when compared to other sequence-based benchmark results [12].

4.2. Reward-conditioned RL-based protein engineering framework

4.2.1 Experimental settings

The state, action, and reward will follow the modeling defined in Sec. 3.1 and Sec. 3.3. The environment starts an episode defining an initial sequence for optimize, *e.g.*, wild type or a random sequence. The episode is finished after N steps of optimization. At each time step, the agent receive a sequence to optimize and returns a position in the sequence that should be mutated. The environment then chooses the amino acid to mutate using a pre-trained protein language model. The policy of amino acid selection is non-deterministic. The exploration is stochastic following the distribution of amino acids to ensure that enough exploration in the state space is performed. The sequence is evaluated at each time step to return a reward to the policy.

4.2.2 Preliminary Results

As preliminary experiments for the ability of the surrogate reward function trained in Sec. 3.3, we experiment the modeling of fluorescent proteins using hallucination as implemented in [3]. Our surrogate reward function is combined via addition with the output of a background network that represents how different the current sequence is from an *average* protein. This combined optimization objective is used to guide protein engineering. We test the engineering starting from two option: from a random sequence, and from the wild type protein (PDB:1EMA). The optimization also follows two options: using an MCMC algorithm in which exploration is enabled, and using greedy optimization only accepting mutations that lower the objective function. Some of the examples obtained during these preliminary examples are shown in Fig. 5. It can be observed that starting from a random sequence, the mutations lead to unstable proteins even parts that resemble the wild type are observed. For the designs that uses greedy optimization, 46 mutations were accepted during the optimization process of 40K steps. These results also highlight that, while creating

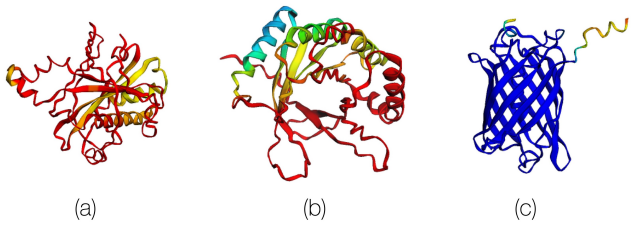

Figure 5. Protein engineered through guided evolution using surrogate reward network using Hallucination [3]. Starting from a random sequence with MCMC optimization (a); starting from a random sequence with greedy optimization (b); starting from the wild type with greedy optimization (c).

a protein engineering framework, it is important to take into consideration the characteristics of the protein function being optimized. The chromophore part of the wild type is usually regarded as being responsible for the fluorescence activity [15], so it is important that during the engineering process this part of the protein is kept active while the policy focus on developing and optimizing parts of the protein that do not affect the functionality being optimized.

5. Discussion

Existing works on protein representation learning [1, 4, 13, 19] mostly focused on sequence-based methods. However, our comparison between representations shows that the structure of the protein encodes more valuable information than the embeddings extracted from the amino acid sequence. This result is quite reasonable since the structure has explicit relation with function, where amino acid sequence does not. Also, the sequence-based methods are trained on whole protein database, without the information about the target function. So the representation may not include the property of the protein which is crucial when predicting certain functionality. This is also the reason why sequence-based method show increased accuracy when fine-tuned to specific target proteins [4]. Also, the proposed representation is able to achieve accurate results even with a train dataset with only 500 samples. Because collecting large amounts of functionality data experimentally is expensive, the ability to predict functionality successfully with limited data will be critical in guiding protein engineering and protein design in the years ahead.

5.1. Interpretability of functionality prediction

The ability of LightGBM to recognize the importance of each feature for the final prediction is valuable. As in the proposed representation each feature corresponds to an interaction between a pair of amino acids, we can qualitatively analyze the importance given by the regressor to each feature. For that, we extracted and plotted (Fig. 6) the importance of interactions given by the LightGBM regressor

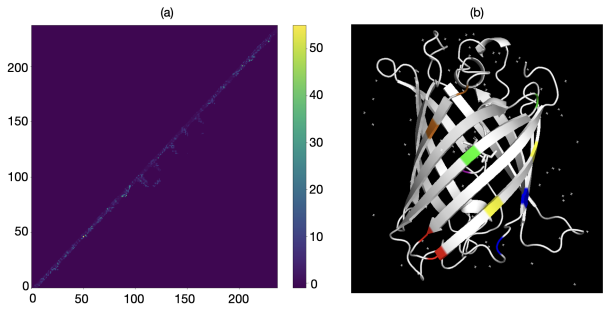

Figure 6. (a) Feature importance of LightGBM regressor. Note that only half of the distogram is used due to its symmetry. (b) Visualization in PyMol [6] in the wild-type (PDB:1EMA). Top 6 interactions are colored from red, most important, to purple, least important.

with best result. It's worth noting that the chromophore area (purple) is one of the six most crucial features for a prediction, which corresponds to the analysis from [15]. The findings suggest that these methods could be useful for identifying critical amino acid interactions that are involved in specific protein functions.

6. Future work

6.1. Protein functionality prediction

We observed that structure-based representation is as powerful as sequence-based representation for predicting the protein functionality. Therefore, we will adapt a self-attention module for protein representation learning and find an efficient way to get the structure-based representation. Then we will test our architecture in avGFP fluorescence dataset [15] and TAPE benchmark [14]. Furthermore, we plan to test our architecture to predict, affinity, specificity and developability of antibodies.

6.2. Protein engineering framework

We plan to implement the protein engineering framework as in Fig. 2 using an OpenAI Gym environment framework and, at first, using the CARP algorithm [19] with a pre-trained model as its language model given its computation efficiency for inference. To model the agent, we plan to test two RL algorithms, one on-policy, PPO [16], and one using a offline framework, Decision Transformers [5], to evaluate the proposed framework in different settings. The implemented protein engineering framework is going to especially be tested to avGFP engineering due to the size of the dataset available for training our functionality prediction network. In addition, in parallel we plan to curate antibody datasets to apply our framework to various antigen-conditioned antibody design problems.

References

[1] Ethan C Alley, Grigory Khimulya, Surojit Biswas, Mohammed AlQuraishi, and George M Church. Unified rational protein engineering with sequence-based deep representation learning. *Nature methods*, 16(12):1315–1322, 2019. 2, 3, 4, 5

[2] Christof Angermueller, David Dohan, David Belanger, Ramya Deshpande, Kevin Murphy, and Lucy Colwell. Model-based reinforcement learning for biological sequence design. In *International conference on learning representations*, 2019. 1, 2

[3] Ivan Anishchenko, Samuel J Pellock, Tamuka M Chidyausiku, Theresa A Ramelot, Sergey Ovchinnikov, Jingzhou Hao, Khushboo Bafna, Christoffer Norn, Alex Kang, Asim K Bera, et al. De novo protein design by deep network hallucination. *Nature*, 600(7889):547–552, 2021. 1, 4, 5

[4] Surojit Biswas, Grigory Khimulya, Ethan C Alley, Kevin M Esvelt, and George M Church. Low-n protein engineering with data-efficient deep learning. *Nature methods*, 18(4):389–396, 2021. 1, 2, 5

[5] Lili Chen and et al Lu, Kevin. Decision transformer: Reinforcement learning via sequence modeling. *arXiv preprint arXiv:2106.01345*, 2021. 5

[6] Warren L DeLano et al. Pymol: An open-source molecular graphics tool. *CCP4 Newsl. Protein Crystallogr*, 40(1):82–92, 2002. 5

[7] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970. 4

[8] Charles A Janeway Jr, Paul Travers, Mark Walport, and Mark J Shlomchik. The structure of a typical antibody molecule. In *Immunobiology: The Immune System in Health and Disease. 5th edition*. Garland Science, 2001. 1

[9] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021. 2

[10] Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aaron van den Oord, Alex Graves, and Koray Kavukcuoglu. Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099*, 2016. 2

[11] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017. 4

[12] Amy X Lu, Haoran Zhang, Marzyeh Ghassemi, and Alan Moses. Self-supervised contrastive learning of protein representations by mutual information maximization. *BioRxiv*, 2020. 4

[13] Soumya Ram and Tristan Bepler. Few shot protein generation. *arXiv preprint arXiv:2204.01168*, 2022. 1, 2, 5

[14] Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Peter Chen, John Canny, Pieter Abbeel, and Yun Song. Evaluating protein transfer learning with tape. *Advances in neural information processing systems*, 32, 2019. 5

[15] Karen S Sarkisyan, Dmitry A Bolotin, Margarita V Meer, Dinara R Usmanova, Alexander S Mishin, George V Sharonov, Dmitry N Ivankov, Nina G Bozhanova, Mikhail S Baranov, Onuralp Soylemez, et al. Local fitness landscape of the green fluorescent protein. *Nature*, 533(7603):397–401, 2016. 3, 5

[16] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 1, 3, 5

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2

[18] Jianyi Yang, Ivan Anishchenko, Hahnbeom Park, Zhenling Peng, Sergey Ovchinnikov, and David Baker. Improved protein structure prediction using predicted interresidue orientations. *Proceedings of the National Academy of Sciences*, 117(3):1496–1503, 2020. 1, 2

[19] Kevin K Yang, Alex X Lu, and Nicolo K Fusi. Convolutions are competitive with transformers for protein sequence pretraining. *bioRxiv*, 2022. 2, 5

[20] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320, 2005. 4