ProGen: Language Modeling for Protein Generation

Ali Madani [1] **Bryan McCann** [1] **Nikhil Naik** [1] **Nitish Shirish Keskar** [1] **Namrata Anand** [2] **Raphael R. Eguchi** [2] **Po-Ssu Huang** [2] **Richard Socher** [1]

Abstract

Generative modeling for protein engineering is key to solving fundamental problems in synthetic biology, medicine, and material science. We pose protein engineering as an unsupervised sequence generation problem in order to leverage the exponentially growing set of proteins that lack costly, structural annotations. We train a 1.2B-parameter language model, ProGen, on ∼280M protein sequences conditioned on taxonomic and keyword tags such as molecular function and cellular component. This provides ProGen with an unprecedented range of evolutionary sequence diversity and allows it to generate with fine-grained control as demonstrated by metrics based on primary sequence similarity, secondary structure accuracy, and conformational energy.

1. Introduction

Generating proteins with desired properties is one of the most complex yet impactful problems in biology. Protein engineering research has grown over the past 50 years and yielded remarkable outcomes including the development of new enzymes, therapies, and sensors. However, leading experimental techniques for protein engineering such as directed evolution (Arnold, 1998) still rely on heuristics and random mutations to select initial sequences for rounds of evolution.

The raw amino acid sequence encodes a protein, and during synthesis, this chain of amino acids folds in ways that exhibit local (secondary) and global (tertiary) structure. These structural properties then directly determine a unique function, which is of ultimate interest to protein engineers. Unfortunately, obtaining three-dimensional structural information for proteins is expensive and time consuming. Consequently, there are three orders of magnitude more raw sequences than there are sequences with structural annotations, and protein sequence data is growing at a near exponential rate.

Recent research (Alley et al., 2019; Rives et al., 2019; Rao et al., 2019) has begun to capitalize on the much larger set of raw protein sequences by adapting state-of-the-art representation learning techniques (Devlin et al., 2018) from natural language processing (NLP) to classification of protein properties. However, there has been no attempt to adapt state-of-the-art methods for artificial text generation (Radford et al., 2019), and in particular the kind of controllable generation (Keskar et al., 2019) that would be most useful for protein engineering.

We introduce ProGen for controllable protein generation. ProGen is a 1.2 billion parameter conditional language model trained on a dataset of 280 million protein sequences together with conditioning tags that encode a variety of annotation such as taxonomic, functional, and locational information. By conditioning on these tags, ProGen provides a new method for protein generation that can be tailored for desired properties (Figure 1).

According to NLP metrics, ProGen is a powerful language model, achieving comparable performance to similarly-sized models for English. This performance improves in settings with larger amino acid contexts and when ProGen is provided a larger number of conditioning tags, which highlights its potential for applications in providing viable, starting sequences for directed evolution or *de novo* protein design (Huang et al., 2016). ProGen also performs well when used to model unseen protein families, but it is even more effective when fine-tuned for those unseen families as an alternative to training from random initialization. These results inspire the use of ProGen to generate candidate sequences in challenging, low-homology applications.

Proteins generated by ProGen satisfy desired structural, and by extension functional, properties when evaluated with metrics for sequence similarity, secondary structure accuracy, and conformational energy– from lower level structure to higher level structure. Generation performance is judged higher quality by higher level metrics, which suggests that ProGen has learned invariances to mutations at the sequence level that conserve structure and inferred function. At the highest level, conformational energy analysis reveals that generated proteins exhibit energy levels near that of native

[1]Salesforce Research, Palo Alto, CA, USA [2]Department of Bioengineering, Stanford University, Stanford, CA, USA. Correspondence to: Ali Madani <amadani@salesforce.com>.

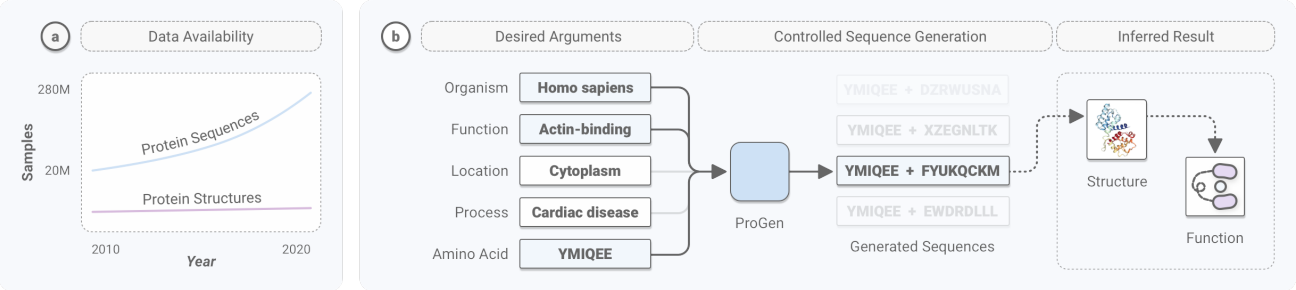

Figure 1. a) Protein sequence data is growing exponentially as compared to structural data. b) We utilize protein sequence data along with taxonomic and keyword tags to develop a conditional language model: ProGen.

proteins, providing our strongest evidence that these proteins satisfy the desired structure and inferred function. In our first case study, we examine completion of a VEGFR2 protein, which is held-out in training. ProGen generates candidate completions that conserve the structural elements most important for determining function and exhibit conformational energy near to native levels across a variety of generation lengths. In our second case study, we observe that ProGen can select high fitness antibody-binding GB1 proteins without supervised training or unsupervised finetuning– indicating that ProGen has learned the underlying distribution of functional proteins.

2. Related Work

Protein representation learning. Recent methods for contextualized representations (McCann et al., 2017; Peters et al., 2018; Devlin et al., 2018) in natural language processing have been demonstrated to work well for contextual protein representation learning. Structural information about a protein can be extracted from such representations using linear methods, and the representations themselves can be adapted to improve performance on other tasks (Rives et al., 2019). Similarly, UniRep (Alley et al., 2019) demonstrated that such representations could be used to predict stability of natural and *de novo* designed proteins as well as quantitative function of molecularly diverse mutants. TAPE (Rao et al., 2019) is a new benchmark consisting of five tasks for assessing such protein embeddings. While this body of prior work focuses on transferable representation learning using bidirectional models, our work demonstrates controllable protein engineering with generative, unidirectional models.

Generative models for protein engineering. Recent generative modeling work such as Ingraham et al. (2019) extends the transformer to condition it on a graph-structured specification of a desired target. Anand & Huang (2018) utilizes generative adversarial networks to produce 2D pairwise distance map for given protein structural fragments, essentially in-painting missing residues. The aforementioned work, along with O'Connell et al. (2018), Boomsma & Frellsen (2017), and Greener et al. (2018), all utilize explicit structural information for generative modeling, thereby are unable to fully capture the number and diversity of sequence-only data available. Meanwhile sequence-only generative modeling have been attempted recently through residual causal dilated convolutional neural networks (Riesselman et al., 2019) and variational autoencoders (Costello & Martin, 2019). Unlike these prior works, our work on generative modeling focuses on a high-capacity language models that scale well with sequence data and can be used for controllable generation.

Language Models and Controllable Generation. Large Transformer architectures (Vaswani et al., 2017) like GPT-2 (Radford et al., 2019) represent the state-of-the-art in unconditional language modeling and demonstrate impressive text generation capabilities (Zellers et al., 2019) after training on vast amounts of unsupervised English text. CTRL (Keskar et al., 2019) trained a similarly large Transformer architecture for language generation by conditioning on properties of the text easily extracted at scale, e.g. domain, style, and even associated URL. We adapt this perspective to protein engineering by training a conditional transformer language model on amino acid sequences conditioned on a set of protein properties referred to as conditioning tags. Notably different from Keskar et al. (2019), protein engineering requires a finer-grained, much larger, and more complex set of conditioning tags. Additionally, a single protein can be paired with dozens of conditioning tags.

3. Methods

Let $a = (a_1, \ldots, a_{n_a})$ be a sequence of amino acids that constitutes a protein. In the context of protein engineering, there is typically also a set of desired protein properties such as function or affiliation with a particular organism. Following recent work on controllable, conditional language modeling (Keskar et al., 2019), we refer to these properties

generally as 'conditioning tags' through which we would like to control generation of amino acid sequences. Let $c = (c_1, \ldots, c_{n_c})$ be a sequence of such conditioning tags, and let $x = [c; a]$ the sequence formed by prepending a conditioning tag sequence to an amino acid sequence. $p(x)$ is then the probability over such combined sequences of length $n = n_a + n_c$. We can factorize this distribution using the chain rule of probability (Bengio et al., 2003):

$$p(x) = \prod_{i=1}^{n} p(x_i | x_{<i})$$

This decomposes the problem of conditional protein generation into next-token prediction, where a token x_i can either be an amino acid or a conditioning tag. A neural network with parameters θ can then be trained to minimize the negative log-likelihood over a dataset $D = \{x^1, \ldots, x^{|D|}\}$:

$$\mathcal{L}(D) = -\sum_{k=1}^{|D|} \log p_\theta(x_i^k | x_{<i}^k)$$

Note that $p(a|c)$, the distribution over proteins conditioned on their corresponding conditioning tags, is just one of the many conditional distributions that can be recovered from a model that learns $p(x)$. A new protein $\tilde{a}$ of length m_a with desired properties encoded by a conditioning tag sequence $\tilde{c}$ of length m_c can then be generated by sequentially sampling its constituent symbols: $p_\theta(a_0|\tilde{c}), p_\theta(a_1|\tilde{a}_0, \tilde{c}), \ldots, p_\theta(a_p|\tilde{a}_{<p}, \tilde{c})$.

We train a variant of the Transformer (Vaswani et al., 2017) to learn these conditional distributions over amino acids and conditioning tags. A sequence containing n tokens is embedded as a sequence of n corresponding vectors in $\mathbb{R}^d$. Each vector is the sum of a learned token embedding and a sinusoidal positional embedding as in the original Transformer architecture. This sequence of vectors is stacked into a matrix $X_0 \in \mathbb{R}^{n \times d}$ so that it can be processed by l attention layers. The ith layer consists of two blocks, each of which preserves the model dimension d.

The core of the first block is multi-head attention with k heads that uses a causal mask to preclude attending to future tokens:

$$\text{Attention}(X, Y, Z) = \text{softmax}\left(\frac{\text{mask}(XY^\top)}{\sqrt{d}}\right) Z$$
$$\text{MultiHead}(X, k) = [h_1; \cdots; h_k] W_o$$
$$\text{where } h_j = \text{Attention}(XW_j^1, XW_j^2, XW_j^3)$$

The core of the second block is a feedforward network with ReLU activation that projects inputs to an inner dimension f, with parameters $U \in \mathbb{R}^{d \times f}$ and $V \in \mathbb{R}^{f \times d}$:

$$FF(X) = \max(0, XU)V$$

Each block precedes core functionality with layer normalization (Ba et al., 2016; Child et al., 2019) and follows it with a residual connection (He et al., 2016). Together, they yield X_{i+1}:

Block 1	Block 2
$\bar{X}_i = \text{LayerNorm}(X_i)$	$\bar{H}_i = \text{LayerNorm}(H_i)$
$H_i = \text{MultiHead}(\bar{X}_i) + \bar{X}_i$	$X_{i+1} = \text{FF}(\bar{H}_i) + \bar{H}_i$

Scores are then computed from the output of the last layer:

$$\text{Scores}(X_0) = \text{LayerNorm}(X_l) W_{vocab}$$

During training, these scores are the inputs of a cross-entropy loss function. During generation, the scores corresponding to the final token are normalized with a softmax, yielding a distribution for sampling a new token.

3.1. Data

We utilize all protein sequences and associated tags available in Uniparc (Leinonen et al., 2004), UniprotKB (Bairoch et al., 2005), SWISS-PROT (Bairoch et al., 2004), TrEMBL (Boeckmann et al., 2003), Pfam (Bateman et al., 2004), and NCBI taxonomic information (Federhen, 2012). The aggregated dataset contains over 281M proteins—the most comprehensive, non-redundant, annotated database of proteins used to train a machine learning model. For the amino acid vocabulary, we use the standard 25 amino acids designations in IUPAC (Pettit & Powell, 2006). The conditioning tags are divided into 2 categories: (1) keyword tags and (2) taxonomic tags. Following the definitions laid out in the UniprotKB controlled, hierarchical vocabulary of keywords (many of which are derived from Gene Ontology (GO) terms) (Ashburner et al., 2000), the conditioning keyword tags included 1100 terms ranging from cellular component, biological process, and molecular function terms. The taxonomic tags include 100k terms from the NCBI taxonomy across the eight standard taxonomic ranks. The aggregated dataset was split into a training set of size 280M, a held-out protein family[1] test set (OOD-test) of size 100k, and a randomly sampled test set (ID-test) of size 1M. OOD-test comprises of 20 protein families, as defined in Pfam, that were excluded from the training data. Performance on OOD-test measures ability to model samples from unseen protein families, whereas performance on ID-test measures ability to model samples from a wider range of protein families that more closely match the distribution of the training set as described in section A.1

[1]Protein families are groups of evolutionarily-related proteins that have similar structure, function, and sequence similarity as defined by Pfam (Bateman et al., 2004)

3.2. Training Details

For training, we include each sequence and its reverse, as proteins are invariant to the temporal notion of sequence generation. We then prepend each sequence (and its reverse) with a corresponding subset of conditioning tags. For a given sequence, there can be multiple versions across databases, each with their own associated conditioning tags. In training, we randomly sample which set of conditioning tags to utilize but bias toward SWISSPROT tags as they are manually verified. We apply dropout to the conditioning tags themselves at a rate of 0.4. We additionally always include a sample with the sequence alone without conditioning tags so that ProGen can be used to complete proteins using only sequence data even when no protein properties are known. We then truncate all sequences to a maximum length of 512. Sequences of length less than 512 were padded, but no loss was backpropagated through the network for padding tokens. The model has dimension $d = 1028$, inner dimension $f = 512$, 36 layers, and 8 heads per layer. Dropout with probability 0.1 follows the residual connections in each layer. Token embeddings were tied with the embeddings of the final output layer (Inan et al., 2016; Press & Wolf, 2016).

Our model was implemented in TensorFlow (Abadi et al., 2016) and trained with a global batch size of 64 distributed across 256 cores of a Cloud TPU v3 Pod for 1M iterations. Training took approximately two weeks using Adagrad (Duchi et al., 2011) with linear warmup from 0 to $1e^{-2}$ over 40k steps. Gradient norms were clipped to 0.25. Training in early stages was improved and stabilized by initializing with the pretrained weights of Keskar et al. (2019).

3.3. Generation Details

ProGen generates proteins one amino acid at a time. For one step of generation, ProGen takes a context sequence of amino acids as input and outputs a probability distribution over amino acids. We sample from that distribution and then update the context sequence with the sampled amino acid. This process repeats until a protein of desired length has been generated. We compare different combinations of top-k sampling (Radford et al., 2019) with a repetition penalty designed for amino acid sequence generation. The repetition penalty reduces the probability of amino acids that have been generated within 4 tokens prior to the token to be predicted. Top-k sampling draws the next token from the k most probable tokens in the distribution output by ProGen. We report results for top-k values of $k = 1$ and $k = 3$ with repetition penalties of 0 and 1.2.

3.4. Evaluation Details

To assess how well ProGen models the training and test distributions, we rely on perplexity as the standard metric for language models, a mean hard accuracy over each token to strictly assess each amino acid error, and a mean soft accuracy defined by incorporating BLOSUM62 (Henikoff & Henikoff, 1992), a standard amino acid substitution matrix.

Perplexity is the exponentiated cross-entropy loss computed over each token in a dataset. Thus, high quality language models are expected to have low perplexities. Mean per-token hard accuracy over the tokens in a sequence judges a prediction incorrect for any amino acid that is not the ground truth. Mean per-token soft accuracy relies on BLOSUM62, a block substitution matrix that specifies which amino acid substitutions are more or less acceptable according to their frequency in known well-formed proteins. BLOSUM62 is widely used across adopted alignment software (e.g., BLAST[2]). Our mean per-token soft accuracy uses BLOSUM62 to penalize incorrect amino acid predictions according to the frequency of that substitution in the matrix. In this way, if the substitution is likely in nature, soft accuracy penalizes the model less.

To assess the quality of generation, we evaluate across three levels of structure: (1) primary sequence similarity, (2) secondary structure accuracy, and (3) conformational energy analysis.

Primary sequence similarity is defined by a global, pairwise sequence alignment score computed with the Biopython package[3]. This score is based on the Needleman-Wunsch algorithm (Needleman & Wunsch, 1970) informed by the BLOSUM62 substitution matrix. We use a gap open penalty of -0.5 and gap continue penalty of -0.1. The resulting score is then normalized by the length of the protein. Experiments reporting sequence similarity are limited to test samples with a form of experimental evidence of X-ray/NMR crystallography, mass spectrometry, or existence in cDNA or RT-PCR to indicate transcript existence. We refer the reader to UniprotKB existence scores with experimental evidence[4] for further details.

Secondary structure accuracy was computed per-residue for predicted secondary structures by PSIPRED[5] with greater than 0.5 confidence. PSI-BLAST was performed on each generated sample to extract the Multiple Sequence Alignments (MSAs) with respect to the UniRef90 database (Suzek et al., 2015). These MSAs were provided to PSIPRED for higher quality secondary structure prediction. Experiments reporting secondary structure accuracy were limited to test

[2]https://blast.ncbi.nlm.nih.gov/Blast.cgi
[3]https://biopython.org/
[4]https://www.uniprot.org/help/protein_existence
[5]http://bioinf.cs.ucl.ac.uk/psipred/

Table 1. ProGen outperforms uniform random and empirical baselines on the full test set, which includes ID- and OOD-test. OOD-test results reveal that ProGen also performs well on protein families unseen during training. Fine-tuning ProGen dramatically improves performance over training from random initialization.

MODEL	PPL	HARD ACC.
UNIFORM BASELINE	25	4
EMPIRICAL BASELINE	18.14	6
PROGEN	8.56	45
ID-TEST	8.17	45
OOD-TEST	13.34	22
OOD-TEST-20 (RAND. INIT.)	17.78	9
OOD-TEST-20 (FINE-TUNED)	7.45	50

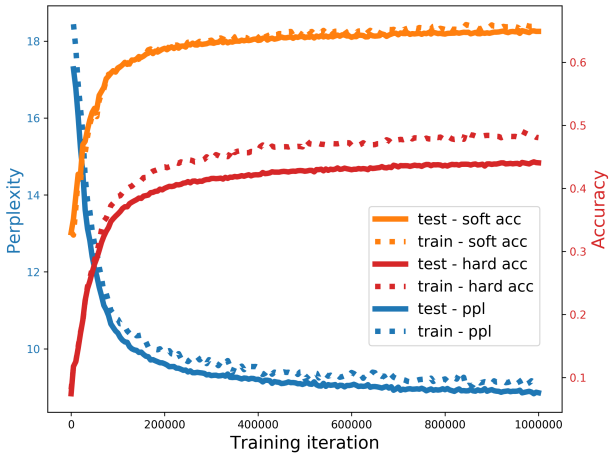

Figure 2. Large model capacity is warranted as ProGen has yet to overfit. BLOSUM62-informed soft accuracy shows no gap between train and test performance, suggesting hard accuracy hides the possibility that ProGen errors often correspond to amino acid substitutions found in nature. For metrics details see Section 3.4.

samples with high UniprotKB existence scores as described in the previous paragraph.

Conformational energy uses the Rosetta-RelaxBB protocol[6]. Rosetta-RelaxBB performs a Monte Carlo optimization of the Rosetta energy function over the space of amino acid types and rotamers. The Rosetta energy is based on biophysical laws and constraints. Between each design round, amino acid side-chains are replaced, while the carbon backbone torsions are kept fixed. Energy minimization/relaxation is performed after threading the amino acid sequence through the known structure. This allows the backbone to move, possibly into a lower energy state. A lower resulting Rosetta energy correlates to a more relaxed-state and viable conformation for a given protein sequence. Before applying the procedure above, we relax the native template first. Experiments that report conformational energy are limited to test samples from SWISSPROT with associated 3D structures in RCSB PDB [7].

To assess generative quality, we provide baselines for different levels of random mutation. For a given sequence, a proportion $(25 - 100\%)$ of amino acids in the sequence is randomly substituted within one of the 20 standard amino acids other than itself. For conformational energy, we also include an all-alanine baseline (i.e. a sequence with only the amino acid alanine), as it is a non-bulky, chemically inert amino acid that mimics the existing secondary structure well when substituted. These baselines provide a scale across each of the above metrics. A particular random mutation may or may not have constructive or destructive effects on protein structure or function. But viewed in aggregate, the performance of the 100% mutation baseline for any metric indicates failed generation. As performance approaches 0%, generation statistically indicates a closer reflection to desired structural and functional properties.

[6]https://www.rosettacommons.org/

[7]https://www.rcsb.org/

4. Results and Analysis

4.1. Evaluating ProGen as a language model

In this section, we demonstrate that ProGen is a high-quality language model according to per-token metrics on the training and test sets.

ProGen generalizes to the full test set and achieves perplexities representative of a high-quality language model. Perplexities reported in Table 1 demonstrate that ProGen dramatically improves over a Uniform Baseline, in which amino acids are sampled according to a uniform distribution, and an Empirical Baseline, in which amino acids are sampled according to the empirical frequencies in the training set. As a point of reference, state-of-the-art unidirectional language models for English Wikipedia achieve perplexities that range from 10 to 17 depending on model size (between 257M and 8.3B parameters) and whether training data was constrained to English Wikipedia (Rae et al., 2019) or not (Shoeybi et al., 2019).

ProGen generalizes to unseen protein families. The second section of Table 1 breaks this result into perplexities over the ID-test and OOD-test sets separately. Results on ID-test confirm that ProGen generalizes well to sequences that belonged to protein families randomly sampled. As expected, performance is worse on the sequences in the OOD-test set, but the model still outperforms the Empirical Baseline for those held out protein families.

Fine-tuning ProGen on unseen protein families improves over training from random initialization. We fur-

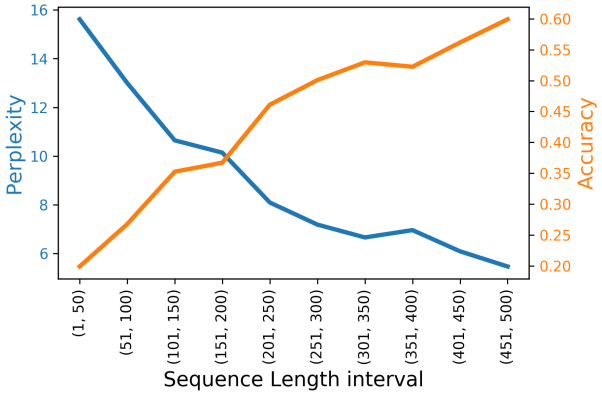

Figure 3. Full test set performance is better for later segments of sequences in keeping with intuition that additional context supports better predictions. We examined intervals up to 500 tokens to ensure a minimum of 30k samples per interval.

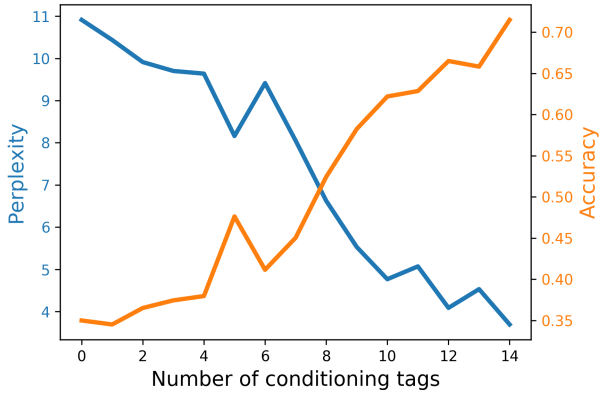

Figure 4. Full test set performance also improves as the number of conditioning tags associated with proteins increases. We examined proteins with up to 14 conditioning tags to ensure a minimum of 3k samples per category.

ther split OOD-test into OOD-test-80 and OOD-test-20, fine-tuned ProGen on OOD-test-80 until convergence (5 epochs; Adam; linear learning rate warmup to 1k iterations), and retested on OOD-test-20. The third section of Table 1 shows that fine-tuning from ProGen improves over training the same architecture with randomly initialized weights.

ProGen performance improves with increased amino acid and conditioning tag context. In Figure 3, we examine the mean perplexity and per-token hard accuracy over different portions of proteins. Perplexity decreases and hard accuracy increases for later portions of a protein, in keeping with the intuition that additional amino acid context narrows down the possibilities for future tokens. The same trends hold when increasing the number of conditioning tags and taking the mean over sequence lengths with the same of tags (in Figure 4). This indicates that conditioning tags also provide signal that improves model predictions.

Training curves suggest that protein generation would benefit from even larger models and longer training. With 1B parameters, ProGen is comparable in size to the largest language models that have been publicly released for any modality, and, to the best of our knowledge, it is the largest model trained on amino acid sequences. Figure 2 shows that despite its size and the amount of compute used to train, ProGen has yet to overfit the training data. This suggests that models for protein generation could still benefit from even larger models and additional compute.

BLOSUM62 soft accuracy reveals that ProGen prediction errors often follow natural amino acid substitutions that likely conserve higher level structure. Though ProGen models proteins as pure sequences, protein function is more directly determined by the secondary and tertiary structures that these sequences encode in three-dimensional space. Model performance based on BLOSUM62 soft accuracy (Section 3.4) is more than 20% higher than using hard accuracy, which indicates that when ProGen errors may often be substitutions that are acceptable in nature because they still reflect the proper higher-level properties. This suggests that ProGen has learned how to work within function-preserving mutational invariances—we continue to validate this finding for primary, secondary, and conformational structure in Section 4.2.

4.2. Generating with ProGen

In this section, we focus on assessing ProGen as a generative model. Generation quality is directly correlated with evolutionary viability and functional qualities, which can be inferred through protein structure. For this reason, we assess generation quality by using metrics for primary sequence similarity, secondary structure accuracy, and conformational energy (Section 3.4).

We also include several mutation baselines (Section 3.4) that allow us to compare the similarity of generated proteins to a target, reference protein across all metrics. In reference to these mutation baselines, ProGen quality improves as we move from primary sequence to full conformational structure metrics, thereby suggesting the model has learned mutational invariances in structure which present as errors in lower-level metrics.

ProGen achieves higher sequence similarity scores with an amino acid repetition penalty. Figure 5 depicts the results of experimenting with various combinations of top-k sampling and repetition penalties (see Section 3.4 for details). Over all context lengths, ProGen performs best

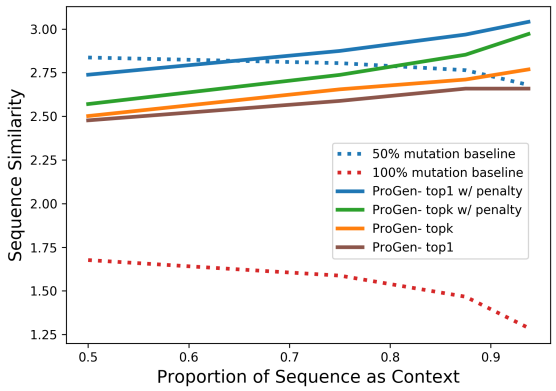

Figure 5. Across all context lengths, greedily sampling with a repetition penalty provides the best results according to sequence similarity.

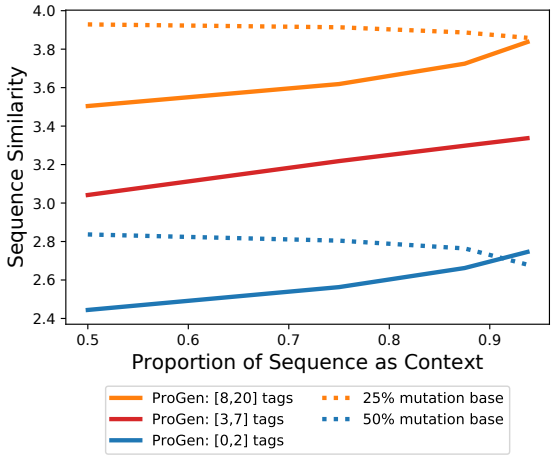

Figure 6. A greater number of conditioning tags enables higher quality generation. With at least 8 conditioning tags, generation quality approaches the 25% mutation baseline.

with $k = 1$ and the repetition penalty applied to recently generated amino acids. Consequently, we use these settings for all following generation experiments. With this nearly greedy sampling, ProGen manages to generate proteins with sequence similarity comparable to randomly mutating 50% of the amino acids that are not seen in the given context.

Sequence similarity suggests that ProGen merely approaches the 25% mutation baseline, but secondary structure accuracy suggests that ProGen surpasses it. In Figure 6, we analyze this sequence similarity across differing numbers of conditioning tags. Sequences associated with at least 3 conditioning tags begin to exceed the 50% mutation baseline, and as amino acid context increases, sequences with at least 8 conditioning tags approach the 25% mutation baseline. Notably, even in the best case, according to sequence similarity, ProGen doesn't surpass the 25% mutation baseline. By contrast, according to secondary structure accuracy, sequences with at least 8 conditioning tags surpass the 25% mutation baseline (Figure 7). This discrepancy between sequence similarity and secondary structure accuracy further corroborates our claim from Section 4: errors registered by lower-level metrics often correspond to acceptable substitutions according to higher-level metrics that more directly correspond to functional viability.

After threading and relaxation, samples generated by ProGen are likely to exhibit desired structure and function. As a measure of generation quality, we thread ProGen sequences through known structures and examine if they exhibit favorable, low energy states. Figure 8 shows the differences between the energy levels of native proteins, ProGen samples, the native proteins with 50% and 100% of amino acids randomly mutated, as well as the all-alanine

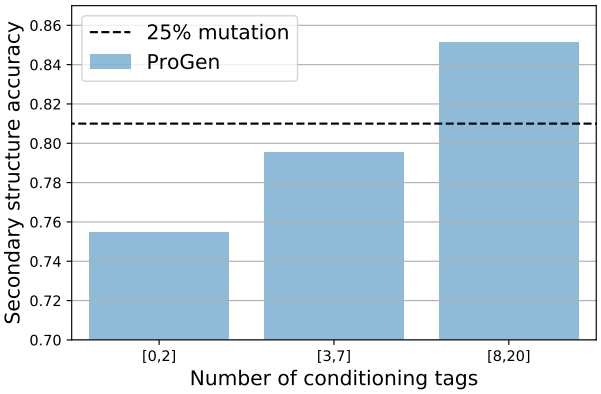

Figure 7. ProGen generates sequences that conserve secondary structure of the protein. Increasing the number of conditioning tags yields better secondary structure accuracy than the 25% mutation baseline.

baseline. Proteins completed by ProGen are much closer to the energy levels of the native protein than all baselines. Generated samples exhibit energy levels near or even below their associated relaxed native templates.

4.3. Case Study: Completing VEGFR2 kinase domain

VEGFR2 is responsible for fundamental cell processes such as cell proliferation, survival, migration, and differentiation. VEGFR2 was excluded from training as a subsequence belongs to a held out protein family in OOD-test. We study how well ProGen generates in the context of a protein com-

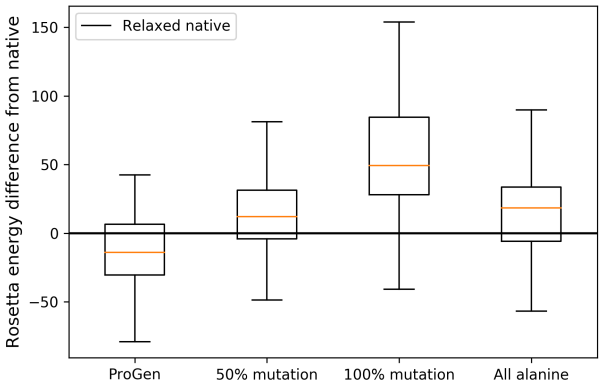

Figure 8. Conformational energies for ProGen generated proteins surpasses all baselines and adheres closely to the energy of the native template.

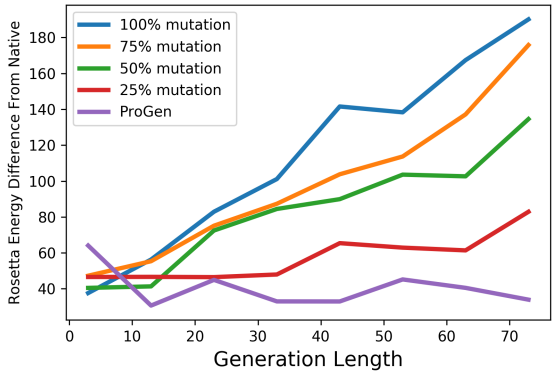

Figure 9. ProGen completion quality for VEGFR2 remains steadily near native conformational energy levels across generation lengths.

pletion task. We consider the amino acid sequence beginning at residue 806 and ending at residue 1168 of VEGFR2 (PDB ID: 2XIR). For different generation lengths, we sample from ProGen to complete the sequence up to residue 1168 with the remainder of the sequence provided as context. Figure 9 shows that the conformational energy calculated after threading and relaxation of ProGen samples are lower compared to all baselines, indicating better structural conservation. Generation quality remains near the native relaxed protein independent of generation length.

The generated samples across Figure 9 exhibit a mean sequence identity of 73.1% with the native sequence. This correlates to a lower sequence identity than the 25% mutation baseline (74% identity) but with better Rosetta energies. This suggests meaningful deviation from the native protein while achieving the ultimate goal of preserving low energy.

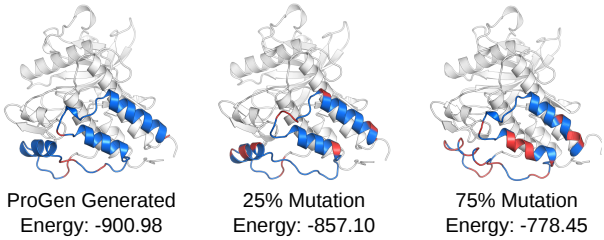

Figure 10. ProGen makes fewer mistakes and prioritizes conservation of secondary structure as compared to baselines. Blue is low energy (stable) and red high (unstable).

Figure 10 shows one sample from ProGen as well as one from each of the 25% and 75% mutation baselines. The ProGen sample exhibits lower energy overall, and energy is highest for amino acids that do not have secondary structure. This suggests that ProGen learned to prioritize the most structurally important segments of the protein.

4.4. Case Study: Zero-shot fitness selection for protein GB1

The ultimate goal of protein engineering is to engineer *functional* proteins. One promising avenue is via directed evolution, which iterates through rounds of mutation and screening to converge on a high-fitness (i.e. functioning) protein. Machine learning has shown initial promise to aid in the subsequent rounds of directed evolution by *in silico* screening of proteins (Wu et al., 2019), but it still relies on random mutation in an exponentially large search space. Ideally, a generative model, such as ProGen, that has learned the distribution of evolutionarily-relevant proteins can directly generate high-fitness proteins.

We examine the empirical fitness landscape of protein G domain B1 (GB1) binding to an antibody (Wu et al., 2016). Protein G is important for the purification, immobilization, and detection of immunoglobulins (antibodies), proteins used by our immune system to neutralize pathogenic viruses and bacteria. Ideally, we would want the ability to generate GB1 proteins with high binding affinity and stability. The data includes 149,361 of a total 160,000 possible variants from NNK/NNS saturation mutagenesis at four positions known to interact epistatically. Reported fitness values correspond to a measure of both stability (i.e. the fraction of folded proteins) and function (i.e. binding affinity to IgG-Fc) by coupling mRNA display with next-generation sequencing. Protein sequences with high fitness values are desired.

Without supervised training of ProGen on the GB1 data or unsupervised fine-tuning of ProGen on a subset of similar immunoglobulin-binding proteins, we pass each variant

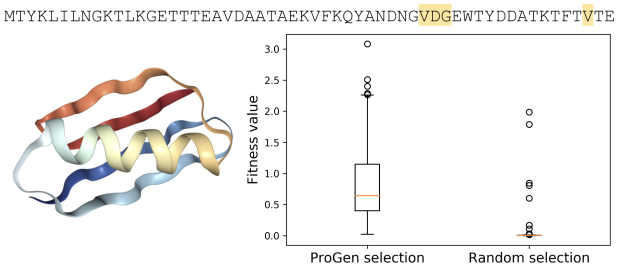

Figure 11. Without training on the Wu et al. (2016) dataset, ProGen can identify which protein variants exhibit high fitness. The dataset reports fitness values for protein variants of GB1 binding to an antibody. Each sample corresponds to mutating one of four highlighted residues, in the above sequence, to a standard amino acid. At the left, the crystallized structure of GB1 is shown. At the right, the fitness value of samples selected through ProGen vs random selection are shown.

through ProGen and select the top one hundred variants with the lowest perplexity values. In Figure 11, we demonstrate ProGen is effective in zero-shot selection of high-fitness protein sequences. In comparison, random mutation, which is the main technique used by directed evolution and ML-assisted directed evolution, statistically generates samples with low or zero fitness. With effective sampling techniques, ProGen can be utilized to generate a spread of samples that are statistically high fitness. These results imply that ProGen has not only learned the distribution of structurally-relevant proteins, but also functionally-relevant proteins.

5. Conclusion

We introduced ProGen, a controllable protein generation language model trained on the full evolutionary diversity of one of the largest sequence databases. The model generates proteins that exhibit near native structure energies which likely implies functional viability. ProGen has the potential to play a new, complementary role alongside other state-of-the-art methods in protein engineering. For example, in directed evolution, initial sequences may be sampled from ProGen according to desired conditioning tags. In later rounds of evolution, protein completion with context for particular residue spans, or hotspots, may provide higher fitness samples. In *de novo* protein design, using ProGen with conditioning tags may allow for designing new proteins with existing folding motifs in new protein families or host organisms. This same strategy may be used in conjunction with threading and structure-based protein design. Because conditioning tags orient ProGen in sequence space, ProGen may even be used as a model to sample from the distribution of evolutionarily viable proteins near one particular protein. This may provide useful augmentations around data for non-homologous domains where existing techniques, such as MSAs, fall short.

6. Acknowledgements

We would like to thank Alex Chu for assistance in the threading and minimization experiments, Melvin Gruesbeck for figure design, and Jesse Vig for visualizing the attention heads of ProGen.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016.

Alley, E. C., Khimulya, G., Biswas, S., AlQuraishi, M., and Church, G. M. Unified rational protein engineering with sequence-based deep representation learning. *Nature methods*, 16(12):1315–1322, 2019.

Anand, N. and Huang, P. Generative modeling for protein structures. In *Advances in Neural Information Processing Systems*, pp. 7494–7505, 2018.

Arnold, F. H. Design by directed evolution. *Accounts of chemical research*, 31(3):125–131, 1998.

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., et al. Gene ontology: tool for the unification of biology. *Nature genetics*, 25(1):25–29, 2000.

Ba, J., Kiros, R., and Hinton, G. E. Layer normalization. *CoRR*, abs/1607.06450, 2016.

Bairoch, A., Boeckmann, B., Ferro, S., and Gasteiger, E. Swiss-prot: juggling between evolution and stability. *Briefings in bioinformatics*, 5(1):39–55, 2004.

Bairoch, A., Apweiler, R., Wu, C. H., Barker, W. C., Boeckmann, B., Ferro, S., Gasteiger, E., Huang, H., Lopez, R., Magrane, M., et al. The universal protein resource (uniprot). *Nucleic acids research*, 33(suppl_1):D154–D159, 2005.

Bateman, A., Coin, L., Durbin, R., Finn, R. D., Hollich, V., Griffiths-Jones, S., Khanna, A., Marshall, M., Moxon, S., Sonnhammer, E. L., et al. The pfam protein families database. *Nucleic acids research*, 32(suppl_1):D138–D141, 2004.

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

Boeckmann, B., Bairoch, A., Apweiler, R., Blatter, M.-C., Estreicher, A., Gasteiger, E., Martin, M. J., Michoud, K., O'Donovan, C., Phan, I., et al. The swiss-prot protein knowledgebase and its supplement trembl in 2003. *Nucleic acids research*, 31(1):365–370, 2003.

Boomsma, W. and Frellsen, J. Spherical convolutions and their application in molecular modelling. In *Advances in Neural Information Processing Systems*, pp. 3433–3443, 2017.

Child, R., Gray, S., Radford, A., and Sutskever, I. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Costello, Z. and Martin, H. G. How to hallucinate functional proteins. *arXiv preprint arXiv:1903.00458*, 2019.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Duchi, J., Hazan, E., and Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(Jul):2121–2159, 2011.

Federhen, S. The ncbi taxonomy database. *Nucleic acids research*, 40(D1):D136–D143, 2012.

Greener, J. G., Moffat, L., and Jones, D. T. Design of metalloproteins and novel protein folds using variational autoencoders. *Scientific reports*, 8(1):1–12, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Henikoff, S. and Henikoff, J. G. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, 89(22):10915–10919, 1992.

Huang, P.-S., Boyken, S. E., and Baker, D. The coming of age of de novo protein design. *Nature*, 537(7620): 320–327, 2016.

Inan, H., Khosravi, K., and Socher, R. Tying word vectors and word classifiers: A loss framework for language modeling. *arXiv preprint arXiv:1611.01462*, 2016.

Ingraham, J., Garg, V., Barzilay, R., and Jaakkola, T. Generative models for graph-based protein design. In *Advances in Neural Information Processing Systems*, pp. 15794–15805, 2019.

Keskar, N. S., McCann, B., Varshney, L. R., Xiong, C., and Socher, R. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.

Leinonen, R., Diez, F. G., Binns, D., Fleischmann, W., Lopez, R., and Apweiler, R. Uniprot archive. *Bioinformatics*, 20(17):3236–3237, 2004.

McCann, B., Bradbury, J., Xiong, C., and Socher, R. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pp. 6294–6305, 2017.

Needleman, S. B. and Wunsch, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, 1970.

O'Connell, J., Li, Z., Hanson, J., Heffernan, R., Lyons, J., Paliwal, K., Dehzangi, A., Yang, Y., and Zhou, Y. Spin2: Predicting sequence profiles from protein structures using deep neural networks. *Proteins: Structure, Function, and Bioinformatics*, 86(6):629–633, 2018.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

Pettit, L. D. and Powell, K. The iupac stability constants database. *Chemistry international*, 2006.

Press, O. and Wolf, L. Using the output embedding to improve language models. *arXiv preprint arXiv:1608.05859*, 2016.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.

Rae, J. W., Potapenko, A., Jayakumar, S. M., and Lillicrap, T. P. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*, 2019.

Rao, R., Bhattacharya, N., Thomas, N., Duan, Y., Chen, P., Canny, J., Abbeel, P., and Song, Y. Evaluating protein transfer learning with tape. In *Advances in Neural Information Processing Systems*, pp. 9686–9698, 2019.

Riesselman, A. J., Shin, J.-E., Kollasch, A. W., McMahon, C., Simon, E., Sander, C., Manglik, A., Kruse, A. C., and Marks, D. S. Accelerating protein design using autoregressive generative models. *bioRxiv*, pp. 757252, 2019.

Rives, A., Goyal, S., Meier, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., and Fergus, R. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *bioRxiv*, pp. 622803, 2019.

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., and Catanzaro, B. Megatron-lm: Training multi-billion parameter language models using gpu model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Suzek, B. E., Wang, Y., Huang, H., McGarvey, P. B., Wu, C. H., and Consortium, U. Uniref clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932, 2015.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 5998–6008. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

Vig, J. A multiscale visualization of attention in the transformer model. *arXiv preprint arXiv:1906.05714*, 2019.

Wu, N. C., Dai, L., Olson, C. A., Lloyd-Smith, J. O., and Sun, R. Adaptation in protein fitness landscapes is facilitated by indirect paths. *Elife*, 5:e16965, 2016.

Wu, Z., Kan, S. J., Lewis, R. D., Wittmann, B. J., and Arnold, F. H. Machine learning-assisted directed protein evolution with combinatorial libraries. *Proceedings of the National Academy of Sciences*, 116(18):8852–8858, 2019.

Zellers, R., Holtzman, A., Rashkin, H., Bisk, Y., Farhadi, A., Roesner, F., and Choi, Y. Defending against neural fake news. *arXiv preprint arXiv:1905.12616*, 2019.

Zimmermann, L., Stephens, A., Nam, S.-Z., Rau, D., Kübler, J., Lozajic, M., Gabler, F., Söding, J., Lupas, A. N., and Alva, V. A completely reimplemented mpi bioinformatics toolkit with a new hhpred server at its core. *Journal of molecular biology*, 430(15):2237–2243, 2018.

A. Appendix

A.1. Measuring out-of-distribution

The objective of our work is to enable high-quality protein generation. To test the effectiveness of our trained model, we had two test subsets: ID-Test and OOD-Test. ID-Test is a random split of the non-redundant sample database and can be viewed as a typical *in-distribution* test set of held-out samples.

In contrast, OOD-Test represents an *out-of-distribution* set. OOD-Test consists samples that contained a matching subsequence residing in one of twenty Pfam protein families that were held out of Train and ID-Test.

	3-GRAM SAE	5-GRAM SAE
TRAIN AND ID-TEST	0.027	0.095
TRAIN AND OOD-TEST	0.399	1.112
ID-TEST AND OOD-TEST	0.387	1.104

Table 2. The training data and ID-Test data seem to be drawn from a similar distribution, but OOD-Test is markedly different from the others. SAE refers to the sum of absolute errors for normalized 3-gram and 5-gram histograms. If two histograms were entirely divergent, the SAE would yield a value of 2.

To quantify the out-of-distribution nature of OOD-Test, we computed a normalized histogram of 3-grams and 5-grams across samples in the Train, ID-Test, and OOD-Test datasets. The sum of absolute errors (SAE) was computed for a pair of histograms as shown in Table 2. Two normalized histograms that align perfectly would have an SAE of 0 and two normalized histograms that are completely divergent would have an SAE of 2. The results imply that the OOD-Test is drawn from a significantly different distribution.

The held-out protein families included `PF18369`, `PF04680`, `PF17988`, `PF12325`, `PF03272`, `PF03938`, `PF17724`, `PF10696`, `PF11968`, `PF04153`, `PF06173`, `PF12378`, `PF04420`, `PF10841`, `PF06917`, `PF03492`, `PF06905`, `PF15340`, `PF17055`, `PF05318`.

A.2. Generation with only conditioning tags

We observe that ProGen can be used to generate proteins with only conditioning tags and no initial amino acid context. For the following example, we prompt ProGen to greedily generate a protein sequence with the tags `Flavoprotein` and `FMN`. As defined by the UniprotKB keyword, the `FMN` tag refers to "a protein involved in flavin adenine mononucleotide (FMN) synthesis or protein which contains at least one FMN as prosthetic group/cofactor (flavoproteins) or cosubstrate, such as many oxidation-reduction enzymes".

The generated sequence of length 400 is then passed to the HHblits package by Zimmermann et al. (2018) to search for a multiple sequence alignment (MSA). As shown in Figure 13, there are multiple sequences that align well with the ProGen sequence. Figures 14-16 demonstrate the alignments have high E-values and have related properties. The lower the E-value, the lower the probability of a random match and the higher the probability that the alignment match is related to the searched sequence.

A.3. Model visualizations

ProGen was trained from a randomly initialized embedding layer with no prior knowledge of residue biochemical properties. Through per-token training on millions of protein sequences, ProGen seems to have inherently learned the natural clustering of amino acids that align with our understanding of biophysicochemical properties. In Figure 12, the trained embedding weights for the standard amino acids tokens are reduced to three dimensions with principle component analysis (PCA).

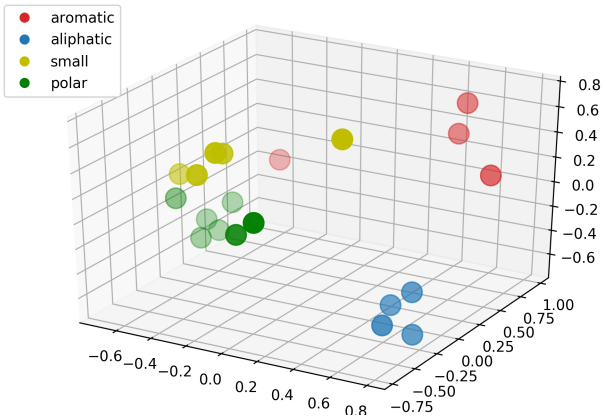

Figure 12. Principle component analysis (PCA) of the ProGen's amino acid embeddings aligns with our intuition of amino acid properties.

Using Vig (2019), we visualize the attention head patterns of ProGen. For both Figure 17 and Figure 18, we are visualizing the attention weight patterns in each head of ProGen for α-actinin protein (PDB: 4D1E) residues 510 to 528, which exhibits an alpha helical structure. In Figure 17, we visualize layers 1 to 3 and attention heads 1 to 12 of ProGen. The attention mechanism exhibits well-differentiated local and global patterns which may indicate specialization of each head on different tasks.

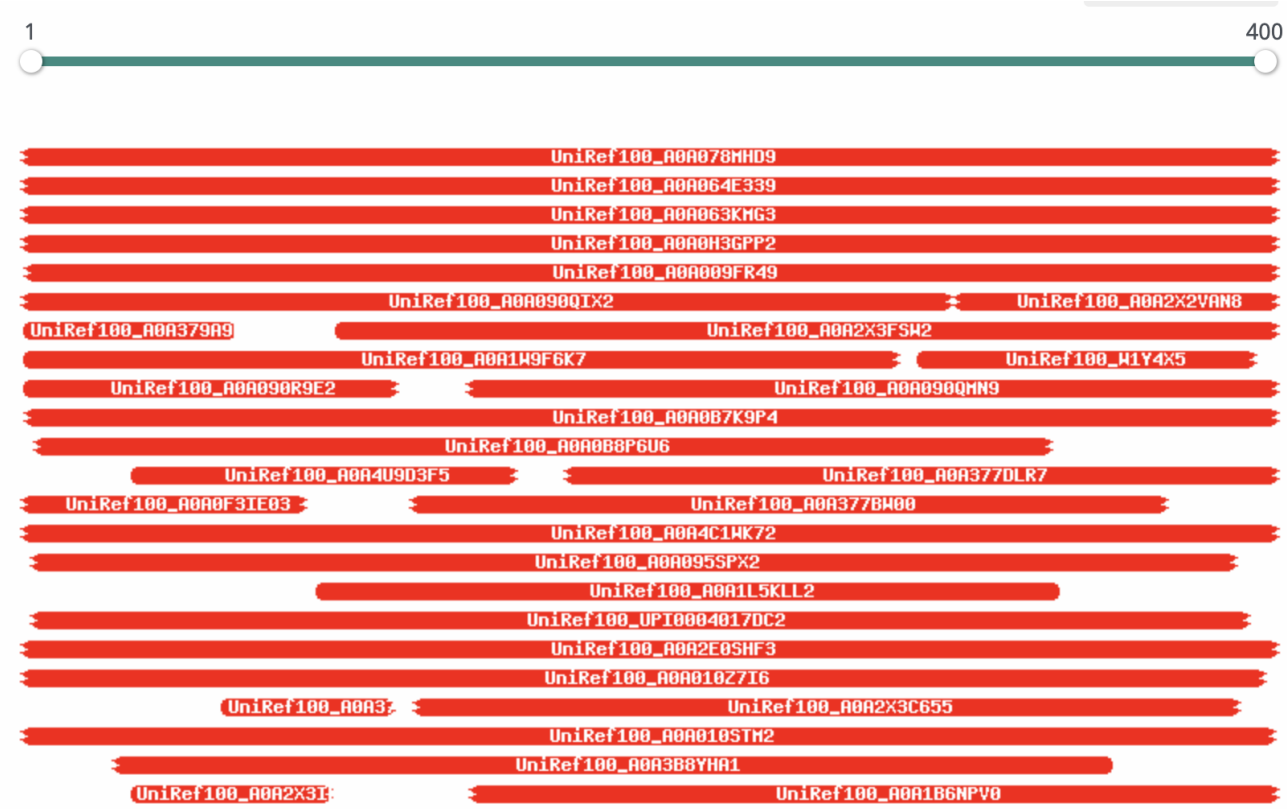

Figure 13. There are multiple sequences that align well with the ProGen generated FMN sequence from only conditioning tags. Many of the matching alignments have properties reflective of FMN proteins (e.g. oxidoreductases). A red color corresponds to a significantly low E-value, implying a matching homolog. The MSA was directly taken using HHblits.

1. UniRef100_A0A078MHD9 Nitrite reductase subunit B n=1 Tax=Pseudomonas saudimassiliensis TaxID=1461581 RepID=A0A078MHD9_9PSED

Probability: 100%, E-value: 1.5e-152, Score: 1145.78, Aligned Cols: 399, Identities: 74%, Similarity: 1.199

```
Q  1      MSKVRLAIIGNGMVGHRFIEDLLDKSDAANFDITVFCEEPRIAYDRVHLSSYFSHHTAEELSLVREGFYDKHGIKVLVGERAITI   85 (400)
          mskvrlaiignmvghrfiedlldksdaanfditvfceepriaydrvhlssyfshhtaeelslvregfydkhgikvlvgeraiti
          |+|.||.|||||||||+|+|.|.++....+|+|||||||||.||||||+||+++++|+|+|++++||++|||++.+|+|++.|
          ~~k~rLVVVGNGMVGH~f~E~Lv~~~~~~~~~ItVf~EEpr~AYDRVhLSeyFsg~~aedLsL~~~~~Y~~~gI~l~lg~rv~~I
T  111    MSKQRLVVIGNGMVGHRFIEQLVAKGAHQQYQITVFCEEPRPAYDRVHLSEYFSGRTAEDLSLVREGFYEKHGITLHLGERVVEI   195 (843)

Q  86     NRQEKVIHSSAGRTVFYDKLIMATGSHPFVPPISGNDTK-C--FRNLEDAKFLYDNANSTGKQAVVIGGGLLGLEAAGALKNLGM   167 (400)
          nrqekvihssagrtvfydklimatgshpfvppisgndtk-c--frnledakflydnanstgkqavviggglllgleaagalknlgm
          +|++|.++++.|+++.|||||+||||.||||||.|+|.. |  +|.+||...+...+.. .|..||||||||||||.|||+||.
          DR~~K~V~T~~G~~~~YDkLVLATGSyPFVPPIpG~d~~gcfVYRTIEDL~aIra~a~~~ak~GvVIGGGLLGLEAA~ALk~LGL
T  196    DRQEKTVTTAAGRTLPYDKLVLATGSYPFVPPIPGADREGCFVYRTIEDLDAIRACARR-AKRGVVVGGGLLGLEAANALKDLGL   279 (843)

Q  168    ETHVVEFAPRLMAVQLDDRGGAMLREKIESTGVRLHTGKNTQEIVNGEQAAHRLKFADGSELETDFIVFSAGIRPQDELARQCGL   252 (400)
          ethvvefaprlmavqlddrggamlrekiestgvrlhtgkntqeivngeqaahrlkfadgseletdfivfsagirpqdelarqcgl
          ||||||||||||++|||+.||++||+|||..||++||+|+||||+.|++..||+.|+||++||||+||||||||||||||+|||
          eTHVVEFAPrLMpvQLDe~GG~~Lr~kIE~LGV~VHT~k~T~eI~~g~~~~~rm~FaDGt~LetDmIVFSAGIRPrDeLAR~cGL
T  280    ETHVVEFAPRLMPVQLDEGGGAQLRRKIEALGVTVHTGKNTQEIVDGEEARHRMNFADGSELETDMIVFSAGIRPRDELARQCGL   364 (843)

Q  253    ALGPRGGIAIDDHCLTSDPDVYAIGECASWHGRVYGLVAPGYKMAQVAVDHILGNENAFKGADMSTKLKLLGVDVGGIGDAHGRT   337 (400)
          algprggiaiddhcltsdpdvyaigecaswhgrvyglvapgykmaqvavdhilgnenafkgadmstklkllgvdvggigdahgrt
          ++||||||+|||+|.|||||||||||+|+||+||||||+||++++|||.|.+++|.|||||||||||||||+.||||||+|
          ~vG~RGGIvIdd~CrTSDpdIyAIGECAlw~grifGLVAPGY~MArvaA~~L~g~~~~FtGADmSTKLKLlGVDVaSiGDAhg~t
T  365    AVGERGGIVIDDHCRTSDPDIYAIGECALWNGRIYGLVAPGYKMARVAADHLLGGDAAFTGADMSTKLKLLGVDVASIGDAHGRT   449 (843)

Q  338    PGARSYVYLDESKEVYKRLIVSEDNKTLLGAVLVGDTSDYGNLLQLVLNNIDLPQHPDSLILP   400 (400)
          pgarsyvyldeskevykrlivsednktllgavlvgdtsdygnllqlvlnnidlpqhpdslilp
          ||++||+|+|+++++||||+||+|+|+|||||||||+|+|++|+|+++|.|.||.+|++||||
          pga~sy~y~D~~~~iYKkLVvS~Dgk~LLGaVLVGDas~Y~~Llq~~~N~i~LP~~Pe~LIlP
T  450    PGARSYVYLDERKGVYKRLVVSEDGKRLLGAVLVGDASDYGTLLQLVLNGIPLPEDPESLILP   512 (843)
```

Figure 14. First alignment (ranked by E-value) of a ProGen generated FMN protein. An E-value less than $1e^{-4}$ and identity greater than 40% is desired to consider the match as potentially homologous. The sequence labeled as Q is the ProGen protein and the sequence labeled as T is the matched sequence.

Figure 15. Second alignment (ranked by E-value) of a ProGen generated FMN protein. An E-value less than $1e^{-4}$ and identity greater than 40% is desired to consider the match as potentially homologous. The sequence labeled as Q is the ProGen protein and the sequence labeled as T is the matched sequence.

Figure 16. Third alignment (ranked by E-value) of a ProGen generated FMN protein. An E-value less than $1e^{-4}$ and identity greater than 40% is desired to consider the match as potentially homologous. The sequence labeled as Q is the ProGen protein and the sequence labeled as T is the matched sequence.

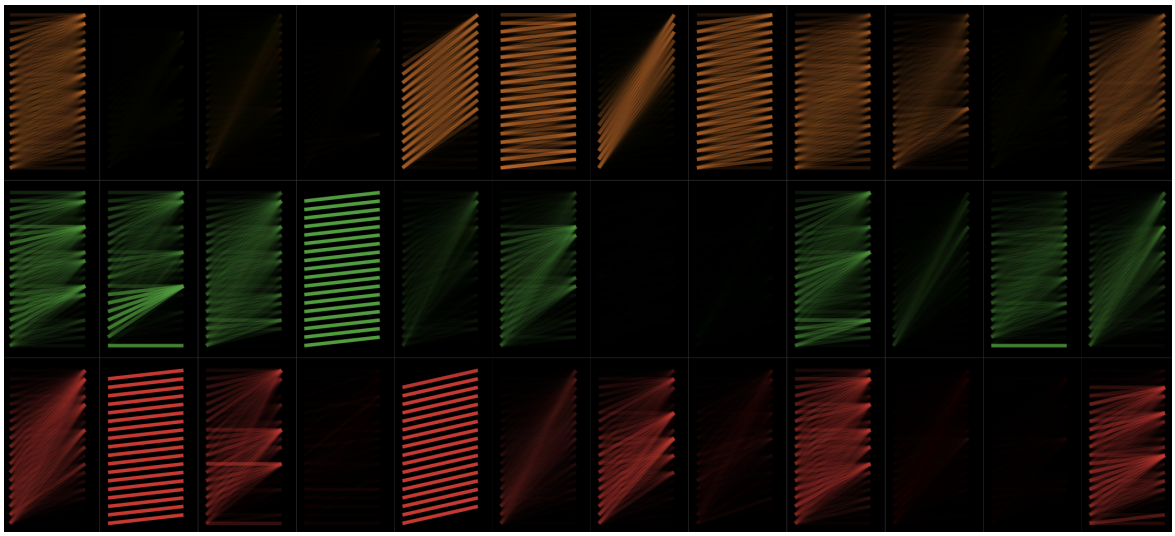

Figure 17. Attention patterns of ProGen for a given sequence. Layers 1-3 (rows) and attention heads 1-12 (columns) are displayed. The attention mechanism exhibits well-differentiated local and global patterns which may indicate specialization of each head on different tasks. Two corresponding attention heads from this visualization are shown in Figure 18.

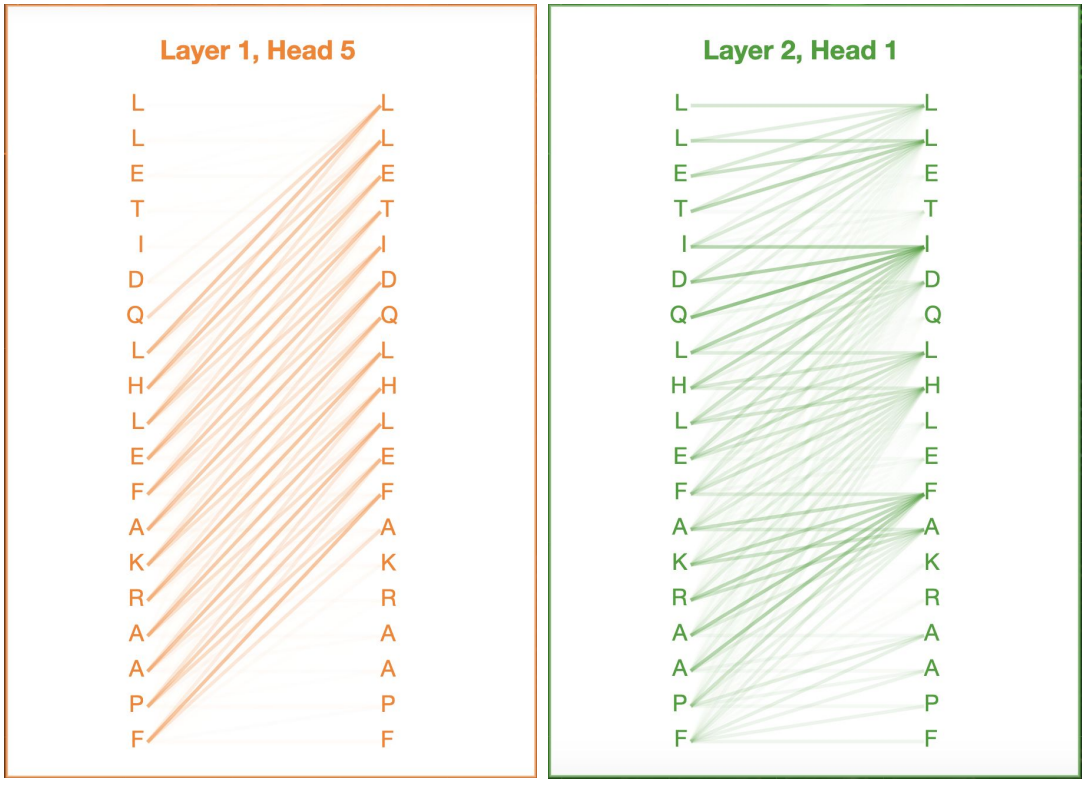

Figure 18. Local attention pattern for two example attention heads. Lines indicate attention to previous tokens for a given predicted token.