DON'T THROW AWAY THAT LINEAR HEAD: FEW-SHOT PROTEIN FITNESS PREDICTION WITH GENERATIVE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Predicting the fitness, i.e. functional value, of a protein sequence is an important and challenging task in biology, particularly due to the scarcity of assay-labeled data. Traditional approaches utilize transfer learning from evolutionary data, yet discard useful information from a generative model's learned probability distribution. We propose generative fitness fine-tuning, termed gf-tuning, to utilize the generative model's log probabilities as logits for a pairwise ranking loss—allowing for the full distribution learned in unsupervised training to be repurposed for fine-tuning on assay-labeled fitness data. We demonstrate that gf-tuning achieves better performance than existing baselines across a variety of few-shot fitness prediction settings, including both low homology and highly epistatic systems as well as generalizing from single to multiple mutations. Generative fitness finetuning offers an effective strategy for few-shot fitness prediction which could enable advances to better understand and engineer proteins.

1 INTRODUCTION

Proteins are complex molecules that perform a spectacular variety of functions that drive biological processes. Their versatility gives them widespread medical, industrial, and environmental use cases. The field of protein engineering aims to design functional protein sequences that perform user-intended purposes. However, existing techniques for protein engineering have limitations. Directed evolution (Arnold, 1998) engineers proteins via wet laboratory procedures, but relies on random mutations to generate candidate proteins for testing. Machine learning models have been suggested to guide the selection of candidate proteins and accelerate rounds of directed evolution (Yang et al., 2019; Wu et al., 2019). Toward this goal, models need to effectively capture the complex sequence-fitness relationship of proteins. If successful, machine learning models that better predict protein fitness can be utilized for suggesting beneficial non-random mutations. Whether making the iterative optimization cycle of directed evolution more efficient or designing combinations of functions for altogether novel proteins, high-performing fitness predictors could enable solutions toward many of the long-standing problems in bioengineering.

Natural evolution provides a useful starting point for predicting protein fitness. The vast majority of possible sequences of amino acids have no biological function and would not fold properly to form a three-dimensional structure. Evolutionary pressures have shaped a space of proteins that exhibit functionality, or high fitness, that is information-rich for representation learning (Alley et al., 2019; Elnaggar et al., 2020). Unsupervised generative models have shown the ability to learn from evolution to predict protein fitness without access to labeled data (Hopf et al., 2017; Riesselman et al., 2018; Madani et al., 2020; Meier et al., 2021). Potts models that use first and second order statistics of aligned protein sequences (Hopf et al., 2017; Russ et al., 2020), variational auto-encoders that model proteins using a latent space (Riesselman et al., 2018), generative adversarial networks that train a protein generator to fool a protein discriminator (Repecka et al., 2021), masked-language models that predict masked amino acid tokens using bi-directional context (Rives et al., 2021; Meier et al., 2021), and auto-regressive language models that model the distribution over protein sequences via left-to-right prediction (Madani et al., 2020; 2021), can all be used model the sequence space of proteins. By learning the space of protein sequences that are naturally plausible, these models

can assign scores based on model likelihoods that often correlate with the true underlying fitness of proteins to perform their intended function.

While natural evolution provides a useful starting point, engineering proteins with novel or enhanced function beyond what exists in nature generally would require using assay-labeled protein sequences, where the fitness of a small number of proteins is measured in the laboratory (Biswas et al., 2021). Assay-labeled measurements are expensive to obtain. Therefore, effective few-shot learning, where a model needs to learn from a small number of labeled examples, is a necessary and pertinent hurdle to address. Specifically, strategies to adapt models that learn from evolution in the unsupervised setting via transfer learning to the downstream task of predicting fitness in the few-shot setting will be increasing important for protein fitness prediction.

Traditional approaches for unsupervised transfer learning in NLP pretrain a language model (LM) on unlabeled sequences, and reuse the hidden layers of the neural network in combination with a new randomly initialized output layer (Radford et al., 2018; Devlin et al., 2018). Similar approaches also exist for protein fitness prediction, but throw away information—the final linear language modeling head is able to predict the likelihood of possible candidate residues (amino acids), but this information is discarded when the LM is reinitialized with a regression head (Rao et al., 2019; Alley et al., 2019; Biswas et al., 2021; Dallago et al., 2021). The likelihoods of amino acid tokens under LMs have been shown to be useful for making zero-shot protein fitness predictions (Madani et al., 2020; Meier et al., 2021), and to generate functional proteins (Madani et al., 2021). Ideally, an LM should retain this likelihood information when finetuning to downstream fitness prediction tasks.

In our work, we present generative fitness finetuning (gf-tuning) as an approach to reuse the full probability distribution learned during unsupervised training to finetune to assay labeled data. We perform rigorous experiments comparing gf-tuning to baselines across multiple few-shot fitness prediction tasks using the same hyper-parameter optimization for all models and averaging over multiple random settings. We empirically validate our motivation by demonstrating gf-tuning outperforms methods that throw away the discriminator head and all other baselines. Aside from operating with limited labeled examples, our approach is particularly well-suited for the more challenging settings, such as low-homology proteins and modeling epistatic sequences with non-linear interactions between positions. Whether its application in scoring protein sequence candidates or designing altogether novel sequences, gf-tuned generative models may enable solutions for human health and the environment.

2 FEW-SHOT PROTEIN FITNESS PREDICTION

The few-shot protein fitness prediction problem setting that we consider in this paper assumes that we have a few-shot dataset of sequences x labeled with continuous fitness values y, $\mathcal{D}_f = \{(x^{(1)}, y^{(1)}), .., (x^{(|\mathcal{D}_f|)}, y^{(|\mathcal{D}_f|)})\}$. In all the tasks considered in this paper, $\mathcal{D}_f$ was acquired in previous work by applying mutations to a wildtype protein (i.e. protein that exists in nature), and obtaining fitness labels for these mutants in laboratory. We also assume that the model has access to a large pretraining dataset of unlabeled protein sequences $\mathcal{D}_u = \{x^{(1)}, .., x^{(|\mathcal{D}_u|)}\}$, and optionally may have access to a smaller dataset of proteins that are evolutionarily related to the wildtype protein, $\mathcal{D}_e = \{x^{(1)}, .., x^{(|\mathcal{D}_e|)}\}$, where $|\mathcal{D}_u| > |\mathcal{D}_e| > |\mathcal{D}_f|$. The task is to learn a predictive function $\hat{y} = F(x)$ that results in a high correlation between $\hat{y}$ and the ground truth fitness labels y. Models are typically evaluated using $Spearman(\hat{y}, y)$, which gives a rank correlation equivalent to the Pearson correlation of the rank variables. This measures the model's ability to accurately rank held out proteins by fitness.

2.1 UNSUPERVISED FITNESS PREDICTION

Unsupervised fitness prediction is an important starting point for our proposed approach. Models can perform protein fitness prediction without training on labeled data by learning to model the probability distribution over natural protein sequences. A probability distribution $P_\theta(x)$ can be pretrained to fit a large database of proteins using the cost function $L(\mathcal{D}_u) = -\mathbb{E}_{x\sim\mathcal{D}_u}[\log P_\theta(x)]$. Optionally, it can be finetuned to evolutionarily related sequences (which we refer to as evolutionary finetuning throughout the paper) by fitting the pretrained unsupervised model to the cost function $L(\mathcal{D}_e) = -\mathbb{E}_{x\sim\mathcal{D}_e}[\log P_\theta(x)]$. Since sequences in $\mathcal{D}_e$ will generally have a related protein func-

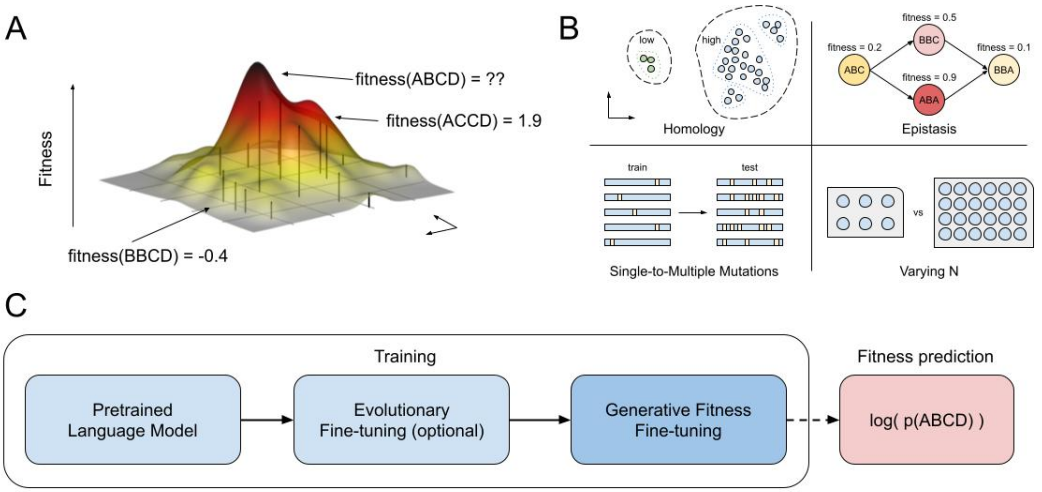

Figure 1: A) Protein sequences exhibit functional values that form a complex fitness landscape as illustrated by Romero & Arnold (2009). Predicting the fitness of a given protein sequence is an important challenge in biology. B) Protein fitness prediction tasks can be further subdivided based on their problem setting. The number of homologous sequences, level of epistasis, extrapolation to higher edit distances, and amount of labeled data in few-shot scenarios are major factors to consider for fitness modeling. C) We introduce gf-tuning, a procedure to tailor a generative LM toward optimal fitness prediction.

tion to the downstream task, finetuning to these sequences can improve the unsupervised fitness prediction ability of the pretrained generative model.

Generative models that have learned the distribution of valid proteins for a particular protein family will assign higher probabilities to valid proteins than invalid ones, allowing them to predict the effect of mutations. As unsupervised models give a strong initialization and useful inductive biases, it is desirable for a model to be finetuned to labeled data to leverage those inductive biases in the most effective way.

2.2 Finetuning LMs with a regression head

Pretrained LMs can be finetuned to sequence regression and classification tasks by retraining the network with a classification or regression head on top of the features of the final layer of the network (Radford et al., 2018; Devlin et al., 2018). In the case of protein language modeling, the methods in the previous section are first applied to pretrain a protein LM by maximizing the log-probability of natural protein sequences under the model. The LM can then be used to map a sequence $x_{1:T} = \{x_1, .., x_t, .., x_T\}$ (where T is the sequence length) to a set of corresponding hidden states $h_{1:T}$, where each $h_t \in \mathbb{R}^d$ and d is the hidden state dimensionality. During pretraining, a linear head $W \in R^{|\mathcal{V}| \times d}$ is used to predict a distribution over the output vocabulary V for each time step, given by $softmax(Wh_t)$. The output distribution can be trained to predict the next token x_{t+1} as in autoregressive language modeling if there is a left-to-right dependency in the predictive function, or recover masked x_t tokens as in masked language modeling. During finetuning, the sequence $h_{1:T}$ is pooled down to a single vector $h_{pool} \in \mathbb{R}^d$ using a pooling function. The pooling function uses the mean or max of each feature across the sequence, or simply uses the hidden state at a certain sequence position or special token. A regression head is then applied on top of this pooled feature representation of the sequence. The regression head can be a neural network, or simply a linear output layer. With a linear regression head, the prediction is given by taking the inner product of a learnable parameter vector w with the pooled sequence features via $\hat{y} = w^\top h_{pool}$. The network is then trained to predict fitness values via the mean squared error between y and $\hat{y}$. The full network can be trained to predict protein fitness end to end (Rao et al., 2019; Dallago et al., 2021), or the embedding of the network can be used as features for another model (Alley et al., 2019; Biswas et al., 2021). Finetuning with a regression head throws away probabilistic information learned by

the LM's output layer during unsupervised pretraining about which sequences are more plausible, which has proven to be useful in unsupervised fitness prediction (Riesselman et al., 2018; Madani et al., 2020; Meier et al., 2021).

2.3 FINETUNING WITH LINEAR REGRESSION-AUGMENTED DENSITY MODELS

Linear regression-augmented density models (Hsu et al., 2021) reuse probabilistic information from unsupervised pretraining by using the log-likelihood from the generative model as a feature for linear regression. This method assumes that the sequences in D_f are all aligned to be the same length, and uses linear ridge regression on the one hot amino-acid representation with an additional feature given by the density of a generative model that has been trained on unsupervised data. Given density weight β, and embeddings for each position $w_{1:T}$ where each $w_t \in \mathbb{R}^{|\mathcal{V}|}$, the predictions given by an augmented density model are given by

$$\hat{y} = \beta \log p(x) + \sum_{t=1}^{T} w_t^\top e_{x_t}, \tag{1}$$

where e_{x_t} gives the one-hot encoding of a residue at position t. The model is fit using ridge regression where the regularization parameter for β is set to be significantly lower than for $w_{1:T}$, forcing the linear model to rely more on the density to make its prediction. In our re-implementation, we use a large scalar value that is multiplied by $\log p(x)$ so that the value of β learned can be much smaller, reducing the effect of regularization on this feature to be negligible. For our baselines with augmented density models, we multiply $\log p(x)$ by 1000 to allow the ridge regression parameter to be shared for all weights, while having little to no effect on regularization of β (since all fitness values in our experiments are much smaller than $1000 \log p(x)$ for all models, allowing the learned β parameter to be used to predict fitness while having a very small value).

2.4 FINETUNING USING WILDTYPE RESIDUAL REGRESSION

Meier et al. (2021) introduced a method for finetuning masked-LMs to fitness prediction using the sum of residuals between mutant log probabilities and wildtype log probabilities at mutation positions as predictions for mutational effect for regression. For our re-implementation, we used mutant marginal probabilities, where predictions are conditioned on the mutant sequence, given by

$$\hat{y} = \sum_{t \in M} logP_\theta(x_t^{mt} | x^{mt}) - logP_\theta(x_t^{wt} | x^{mt}). \tag{2}$$

Mean squared error loss is then applied to $L(\mathcal{D}_f) = \mathbb{E}_{(x,y)\sim\mathcal{D}_f}[(\hat{y}^{(i)} - (y^{(i)} - y^{wt}))^2]$. We refer this method as wild-type residual regression in our experiments. Masked LMs applied to mutational effect prediction assume additive effects of mutations, which would likely lead to non-optimal performance for epistatic proteins (See Section 4). Applying a mean squared error loss to residuals may also force the model to unlearn useful information learned during pretraining, as the unsupervised model will generally have a high mean squared error loss at initialization.

3 GENERATIVE FITNESS FINETUNING

Generative fitness finetuning ("gf-tuning") re-purposes the probability distribution learned during unsupervised training as a pairwise classifier to classify the relative fitness of protein sequence pairs. We use an auto-regressive LM to model $P_\theta(x)$, which computes this as

$$P_\theta(x_{1:T}) = \prod_{t=1}^{T} P_\theta(x_t | x_{<t}), \tag{3}$$

where T is the sequence length. Using an auto-regressive LM allows the model to be better suited for modelling the joint distribution when predicting multiple mutations as compared with a masked LM, which could be expected to be helpful for epistatic proteins since auto-regressive LMs do not assume additive effects of mutations.

All of our LMs are initialized as ProGen (Madani et al., 2020), an autoregressive LM that was pretrained on 280 million proteins. In settings with more evolutionarily related proteins (high homology), we also adapt ProGen to evolutionarily related sequences with evolutionary finetuning (as described in Section 2.1). We then apply gf-tuning to finetune ProGen to assay labeled data. While we use an auto-regressive LM, gf-tuning can be applied to any parameterized probability distribution over protein sequences.

Our proposed cost function trains $P_\theta(x^{(i)})$ to correctly classify whether the fitness of a randomly selected sequence x_i is higher than the fitness of a randomly selected sequence x_j. We directly use the probability density given by the LM to make predictions, which classifies sequences with a higher probability as being higher fitness. The training cost function $\mathcal{L}(\mathcal{D}_f)$ for gf-tuning is given by

$$(x^{(i)}, y^{(i)}),\ (x^{(j)}, y^{(j)}) \sim \mathcal{D}_f \tag{4}$$

$$P_\theta(y^{(i)} > y^{(j)}) = \frac{P_\theta(x^{(i)})^{\alpha/T_i}}{P_\theta(x^{(i)})^{\alpha/T_i} + P_\theta(x^{(j)})^{\alpha/T_j}}. \tag{5}$$

$$\bar{y}_{ij} = \begin{cases} 1, & \text{if } y^{(i)} > y^{(j)} \\ 0.5, & \text{if } y^{(i)} = y^{(j)} \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

$$\mathcal{L}(\mathcal{D}_f) = \mathbb{E}[-\bar{y}_{ij} \log P_\theta(y^{(i)} > y^{(j)}) - (1 - \bar{y}_{ij}) \log(1 - P_\theta(y^{(i)} > y^{(j)})] \tag{7}$$

where α is a hyper-parameter. The method of scoring for pairwise comparisons is based on the Bradley-Terry model (Bradley & Terry, 1952), and our contribution is training a generative model $P(x)$ to assign scores to x. The intuition is that the probability assigned to $P_\theta(y^{(i)} > y^{(j)})$ will be equivalent to the likelihood of observing x_i before x_j if drawing samples from $P_\theta(x^{(i)})$ in the case where $\alpha = T_i = T_j$. A strong pretrained LM that is well suited for the downstream task will often be able to correctly classify most protein pairs with this equation without any supervised training (see Section 2.1), giving it an advantage at initialization over methods that use a randomly initialized regression head. Finetuning the model with this objective should improve its ability to perform this classification, thus allowing it to assign scores that can be better used to rank proteins by fitness. The full pipeline of our proposed approach is given in Figure 1.

4 TASKS

We categorize protein fitness prediction problems along several dimensions to better understand the strengths and weaknesses of each model that we test. Different few-shot protein fitness prediction problems can have very different characteristics, and we aim to capture the range of practical scenarios with our choice of tasks and data sets.

High Homology vs Low Homology: We consider tasks in both the high homology and low homology domain. High homology protein domains have more evolutionarily related sequences that evolved to perform a similar function, and can potentially act as more useful unlabeled sequences for unsupervised training. Unsupervised baselines are therefore expected to perform better in high homology domains, and few-shot models that can better leverage an unsupervised initialization are at an advantage. Low homology domains do not have as many evolutionarily related proteins that perform a similar function, but can still leverage general pretraining across protein databases of many families. We approximate the homology of a protein using the number sequences in the multiple sequence alignment (MSA) of evolutionarily related proteins. Since many proteins in databases can be very small edit distances away from each other, sequence diversity is an important consideration for homology determination. Therefore, we also consider the total number of clusters for sequence clustering with a 50% sequence identity threshold for 80% coverage in sequence alignment using mmseqs2 (Steinegger & Söding, 2017) as another metric.

High Epistasis vs Low Epistasis: We consider tasks with varying degrees of epistatis. In epistatic protein domains, a mutation at one position can greatly influence the mutational effect of a mutation at another position. We consider protein fitness landscapes that can be approximated well by an additive model (that predicts the mutational effect of mutation A and mutation B together to be

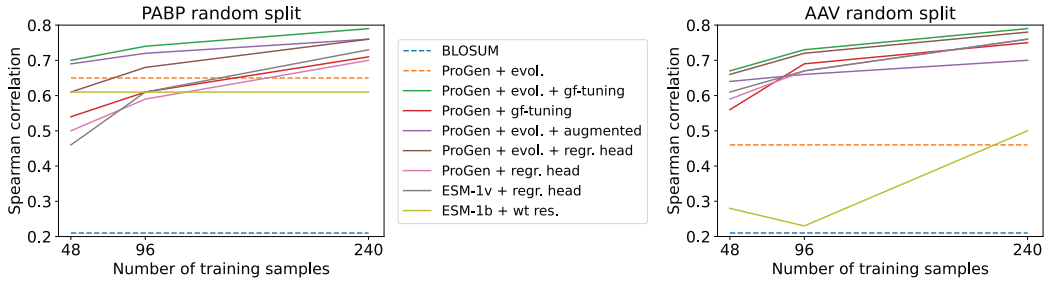

Figure 2: After evolutionary finetuning, gf-tuning achieves the best overall performance in fitness prediction on higher homology datasets, which contain a large number of evolutionarily related sequences. These results suggest that gf-tuning is the most robust way to adapt the representation learned from related sequences to assay labeled data.

the sum of the mutational effect of mutation A and mutation B) to be non-epistatic, and protein fitness landscapes that cannot be modeled well with an additive model to be epistatic. Epistatic fitness prediction problems are more difficult because they require predicting non-linear interactions between mutations to perform well. To score the level of epistasis by our definition, we use the test set Spearman correlation of an additive model that uses the effects of all single mutations to predict multiple mutations as a metric to approximate the (inverse of) epistatis of a protein fitness prediction task.

Random Splits vs Generalizing from Single Mutant: We evaluate all tasks on two different kinds of train/test splits: 1. training and test set randomly distributed. 2. training is on single mutants only, testing is on multiple mutants only. Training on single mutants and evaluating on multiple mutants requires the model to generalize beyond its training set. Single mutation synthesis and assay data can also be easier to perform and acquire in the laboratory, so a model that can generalize from single to multiple mutations is of practical use.

Varying-N for Few-shot Learning: We evaluate all tasks in the few-shot scenario because obtaining labeled data for protein fitness is expensive and models that can perform well few-shot are useful in practice. We consider test set performance for several different training set sizes for each model ($n = 48$, 96, and 240 for all experiments). Leveraging unsupervised pretraining is more important for smaller training set sizes, as there is less information that can be learned during supervised finetuning.

Datasets: We used four fitness prediction datasets to benchmark models in this paper, which we selected to consider all configurations of high homology vs. low homology, and high epistatis vs. low epistatis (statistics for homology and epistasis are given for each dataset in Table 2). The majority of available protein fitness prediction datasets are single mutant only (Gray et al., 2018), so we only include datasets with multiple mutations because a model's ability to predict broader fitness landscapes is of greater practical use. All tasks focus on the few-shot scenario with a varying number of training examples, and consider both the random and singles-to-multiples train/test splits described in the previous subsection. The four datasets used in our experiments are: 1. **PABP** (Melamed et al., 2013) - High homology, low epistasis. 2. **AAV** (Bryant et al., 2021) - High homology, high epistasis. 3. **GFP** (Sarkisyan et al., 2016) - Low homology, low epistasis. 4. **GB1** (Olson et al., 2014) - Low homology, high epistasis. See Appendix A.4 for more details.

5 Experiments

We apply gf-tuning to LMs in the settings and datasets outlined in Section 4, and compare with baselines. For each model in each setting, we plot the test set Spearman correlation between model predictions and test set fitness labels as a function of training set size (running experiments for 48, 96, and 240 training examples, except for GB1 when training on single mutants only, since there are only 76 single mutants). We evaluate on a random subset of 2000 test examples for each plot in order to reduce the computation needed to evaluate many models. For each model in each setting, we obtain results using a consistent training and hyper-parameter optimization procedure across

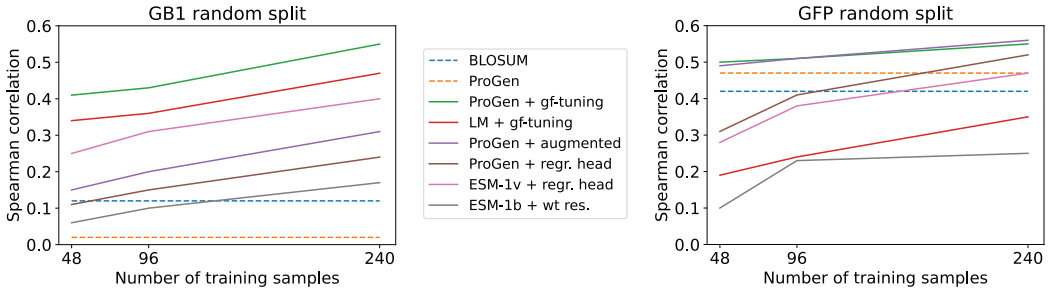

Figure 3: ProGen with gf-tuning performs significantly better than baseline models on low homology domains, which contain a smaller number of evolutionary sequences. Performance is strongest on the lowest homology setting (GB1). In a medium-low homology setting on GFP, augmented density modeling is competitive with gf-tuning.

all models described in Appendix A.2, and average all results over 4 different seeds (that result in 4 different n-sized training sets). In addition to our plots, we also report GB1 and AAV singles to multiples using the full test set to directly compare our approach to published few-shot fitness prediction benchmarks from FLIP (Dallago et al., 2021).

We consider several of the baselines described in Section 2. In the higher homology settings (PABP and AAV), we apply evolutionary finetuning to ProGen to adapt the model to evolutionary related sequences from an MSA to the wildtype protein for the target task. For PABP, we reuse the MSA released by Riesselman et al. (2018), and create our own using similar methodology for AAV. We also apply both ProGen (Madani et al., 2020) and ESM-1v (Meier et al., 2021) with mean pooling and a regression head (see Section 2.2). We do not apply evolutionary finetuning to ESM-1v because according to Meier et al. (2021) finetuning ESM-1v to the MSA on its own hurts downstream task performance, and additional training on the pretraining data is needed during evolutionary finetuning to prevent over-fitting. We also benchmark linear regression augmented density modeling (Section 2.3) applied to ProGen, and ESM-1b with wild-type residual regression 2.4. We use ProGen (with evolutionary finetuning in high-homology tasks) as an unsupervised baseline that does not finetune to assay-labeled data. Finally, we apply BLOSUM62 substitution matrices (Henikoff & Henikoff, 1992) to predict mutational effect, as described in Appendix A.3.

Higher Homology Proteins: We benchmark gf-tuning on the higher homology PABP and AAV datasets using random splits. Our main proposed approached in this setting applies evolutionary finetuning of ProGen on MSAs of the wildtype for each task, followed by gf-tuning to the assay labelled data (ProGen + evol. + gf tuning). We also test baselines including: 1. Applying ProGen with evolutionary finetuning unsupervised (ProGen + evol.) 2. Applying ProGen with gf-tuning without evolutionary finetuning (ProGen + gf-tuning) 3. Applying linear regression-augmented ProGen with evolutionary fine-tuning (ProGen + evol. + augmented) 4. ProGen finetuned to MSE with a regression head with evolutionary fine-tuning (ProGen + evol. + regr. head 5. and without evolutionary fine-tuning (ProGen + regr. head). 6. ESM-1v with a regression head (ESM-1v + regr. head) 7. ESM-1b + wild-type residual regression (ESM-1b + wt res.) 8. BLOSUM62 substitution scores (BLOSUM)

The results are given in Figure 2. Methods that used additional evolutionary finetuning on related sequences (ProGen + evol. + gf-tuning, ProGen + evol. + regr. head, ProGen + evol. + augmented) generally outperformed methods that did not, as might be expected in the high homology setting where there are a large number of functionally similar sequences to use for unsupervised training. Methods that "throw away the linear head" (methods with + regr.) struggle in the lowest n settings for PABD, performing worse than purely unsupervised, indicating that they are not able to leverage the unsupervised initialization well in this setting. ProGen + evol. + gf-tuning achieved the strongest average Spearman correlation across tasks, suggesting that is more robust than the baselines for combining evolutionary data with assay labeled data.

Lower Homology Proteins: We benchmark gf-tuning on the lower homology GFP and GB1 datasets using random splits. Our main proposed approach in this setting applies gf-tuning to pre-

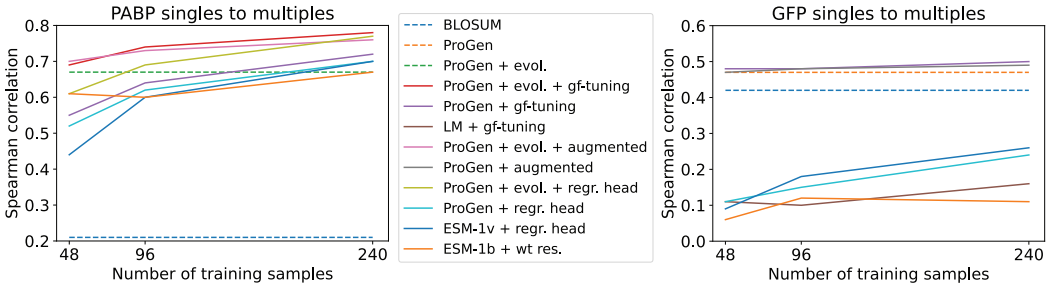

Figure 4: ProGen with gf-tuning is competitive with linear regression-augmented ProGen on fitness prediction of multiple mutations on non-epistatic datasets, when trained on single mutations alone. Since mutation effects are additive in such proteins, a linear model represents a strong baseline.

trained ProGen. We did not use evolutionary finetuning for any method because the effective MSA size is very small for these low homology proteins. We also benchmark the following baselines: 1. Applying ProGen unsupervised (ProGen) 2. Applying gf-tuning to a randomly intialized LM the same size as ProGen 3. Applying linear regression-augmented ProGen (ProGen + evol. + augmented) 4. ProGen finetuned to MSE with a regression head (ProGen + regr. head) 5. ESM-1v with a regression head (ESM-1v + regr. head) 6. ESM-1b with MSE loss on the sum of differences in prediction from wildtype (ESM-1b + wt res.) 7. BLOSOM62 substitution scores (BLOSUM)

Results are given in Figure 3. ProGen with gf-tuning and linear-regression augmented ProGen give results roughly on par for GFP, but gf-tuning outperforms all baselines on GB1 and achieves the highest average performance. gf-tuning also significantly outperforms methods that "throw away the linear head" (methods that use + regr.) on both datasets. Removing pretraining hurts the performance of gf-tuning on GB1, implying that the pretraining step helps ProGen achieve such a strong performance, even though zero-shot ProGen results in a correlation near zero.

Generalizing from Single to Multiple Mutations: We benchmark models on their ability to predict the fitness of sequences that are multiple mutations away from wildtype by only training on sequences that are a single mutation away from wildtype. We report results on datasets with low epistastis (PABP and GFP) in Figure 4 and high epistasis (AAV and GB1) in Figure 5. On the low epistasis datasets where the additive effects of single mutations on fitness are predictive of the fitness of multiple mutations, augmented ProGen and ProGen with gf-tuning were the two strongest models across training set sizes and performed roughly equivalently. Augmented density modeling is a difficult baseline to beat in a setting where there are few non-linear interactions between mutations because it fits to the downstream task with linear regression. In the high epistasis setting, gf-tuning performed significantly better than other models, achieving an average Spearman across training set sizes higher than the next strongest baseline by 0.13 and 0.07 on GB1 and AAV respectively. Augmented ProGen did not perform as well relatively in this setting, likely because it is not able to learn to model non-linear interactions between mutations. These results suggest that gf-tuning is better suited than previous approaches for adapting unsupervised models for few-shot fitness prediction in epistatic landscapes, which is the most difficult setting tested in these experiments.

We also benchmark our main approach (ProGen + gf-tuning for GB1, ProGen + gf-tuning + evol. for AAV) on the FLIP benchmarks (Dallago et al., 2021) for generalizing from single mutations to multiple mutations on GB1 (which uses all multi-mutation proteins as opposed to our split of 2000 randomly sampled) and AAV (which has a full training set of 1170 single mutants). Results in comparison to the FLIP baselines are reported in Table 1.

Many of the strongest baselines from FLIP "throw away the linear head", including all baselines based on ESM-1v and ESM-1b. gf-tuning outperforms all flip baselines on both datasets by a significant margin (improving the Spearman by 0.07 and 0.30 over the next strongest baseline on GB1 and AAV respectively), further justifying our conclusion that gf-finetuning is more effective than previous methods for modeling epistatic proteins from single mutations.

6 Conclusion

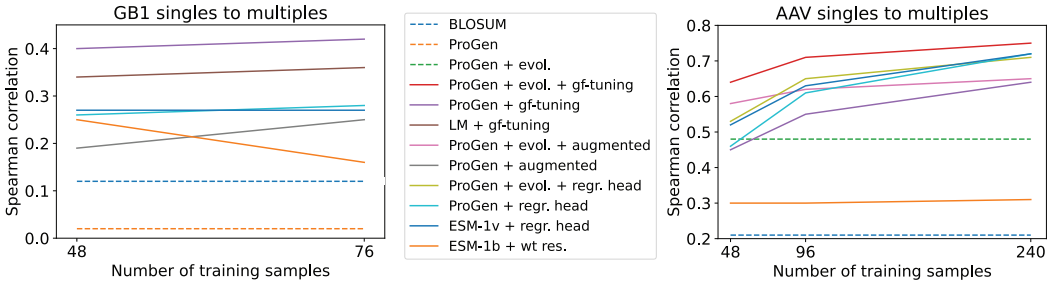

Figure 5: gf-tuning is significantly better than baselines on epistatic protein datasets, when training on single mutations and testing on multiple mutations. Since adding the effects of single mutations is a weak predictor of protein fitness for multiple mutations in these proteins, models that predict these effects additively (e.g., augmented density modeling) or models that may treat the effect of each mutation independently (e.g., masked language modeling) do not perform as well.

Predicting the fitness, or functional value, of a protein sequence is a challenging task that can greatly advance our ability to understand and engineer protein biology. As assay-label acquisition is foreseeably expensive for most proteins, machine learning models will need to operate in the few-shot scenario for supervised fitness learning. In addition to utilizing varying-N of evolutionary and assay-labeled samples, a careful consideration of complexity in differing domain settings (homology, epistasis, single-to-multiple substitutions) are of practical concern to machine learning practitioners and protein engineers alike.

In our work, we propose generative fitness fine-tuning, or gf-tuning, with the motivation of better retaining the probability distribution learned during unsupervised learning to supervised fitness training. We provide the intuition that the traditional approach of discarding a pretrained LM's linear head and using a regression head throws away useful information about protein fitness. For four canonical datasets spanning multiple settings, we validate gf-tuning through model ablations. To ensure the rigor of experimental comparisons in sensitive few-shot tasks, all models undergo hyper-parameter optimization and averaging across multiple seeds for fair comparison. Our experiments show that gf-tuning outperforms both our own baselines and baselines in the literature, and that this improvement is largest on the most difficult fitness prediction tasks.

Table 1: gf-tuning outperforms all published baselines on the few-shot FLIP benchmark tasks (Dallago et al., 2021). The few-shot tasks in FLIP correspond to 1-vs-rest fitness prediction for GB1 and AAV with a Spearman correlation metric. The gf-tuned model was trained four times and the average and best performance is reported.

Model	**GB1 1-vs-rest**	**AAV 1-vs-rest**
ProGen + gf-tuning (best)	**0.45**	**0.79**
ProGen + gf-tuning (avg)	**0.39**	**0.78**
ESM-1b - per AA	0.28	0.03
ESM-1b - mean	0.32	0.04
ESM-1b - mut mean	-0.08	0.31
ESM-1v - per AA	0.28	0.10
ESM-1v - mean	0.32	0.18
ESM-1v - mut mean	0.19	0.44
ESM-untrained - per AA	0.06	0.18
ESM-untrained - mean	0.05	0.01
ESM-untrained - mut mean	0.21	0.26
Ridge	0.28	0.22
CNN	0.17	0.48
Levenshtein	0.17	-0.11
BLOSUM62	0.15	NA

Models trained with gf-tuning could be applied to score natural and designed protein sequences before dedicating wet lab resources for synthesis and characterization. Aside from its discriminative uses, a gf-tuned generative model could conceivably be utilized for generation purposes of artificial proteins. More broadly, we hope gf-tuned generative models can enable machine learning to guide protein engineering efforts to enable solutions in human health and the environment.

7 ETHICS STATEMENT

Predicting the fitness of a protein sequence is a powerful tool that be useful for the design of novel proteins. If our technique or a future iteration thereof is adopted broadly, care should be taken in terms of the end use-cases of these designed samples and downstream effects to ensure safe, non-nefarious, and ethical applications. For projects in any domain, active oversight during project initiation, experimental optimization, and deployment phases should be put in place to ensure safe usage and limitation of unintended harmful effects.

REFERENCES

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.

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

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.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Drew H Bryant, Ali Bashir, Sam Sinai, Nina K Jain, Pierce J Ogden, Patrick F Riley, George M Church, Lucy J Colwell, and Eric D Kelsic. Deep diversification of an aav capsid protein by machine learning. *Nature Biotechnology*, 39(6):691–696, 2021.

Christian Dallago, Jody Mou, Kadina Elizabeth Johnston, Bruce Wittmann, Nick Bhattacharya, Samuel Goldman, Ali Madani, and Kevin K Yang. Flip: Benchmark tasks in fitness landscape inference for proteins. 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rihawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, et al. Prottrans: towards cracking the language of life's code through self-supervised deep learning and high performance computing. *arXiv preprint arXiv:2007.06225*, 2020.

Vanessa E Gray, Ronald J Hause, Jens Luebeck, Jay Shendure, and Douglas M Fowler. Quantitative missense variant effect prediction using large-scale mutagenesis data. *Cell systems*, 6(1):116–124, 2018.

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

Thomas A Hopf, John B Ingraham, Frank J Poelwijk, Charlotta PI Schärfe, Michael Springer, Chris Sander, and Debora S Marks. Mutation effects predicted from sequence co-variation. *Nature biotechnology*, 35(2):128–135, 2017.

Chloe Hsu, Hunter Nisonoff, Clara Fannjiang, and Jennifer Listgarten. Combining evolutionary and assay-labelled data for protein fitness prediction. *bioRxiv*, 2021.

Ali Madani, Bryan McCann, Nikhil Naik, Nitish Shirish Keskar, Namrata Anand, Raphael R Eguchi, Po-Ssu Huang, and Richard Socher. Progen: Language modeling for protein generation. *arXiv preprint arXiv:2004.03497*, 2020.

Ali Madani, Ben Krause, Eric R Greene, Subu Subramanian, Benjamin P Mohr, James M Holton, Jose Luis Olmos, Caiming Xiong, Zachary Z Sun, Richard Socher, et al. Deep neural language modeling enables functional protein generation across families. *bioRxiv*, 2021.

Joshua Meier, Roshan Rao, Robert Verkuil, Jason Liu, Tom Sercu, and Alexander Rives. Language models enable zero-shot prediction of the effects of mutations on protein function. *bioRxiv*, 2021.

Daniel Melamed, David L Young, Caitlin E Gamble, Christina R Miller, and Stanley Fields. Deep mutational scanning of an rrm domain of the saccharomyces cerevisiae poly (a)-binding protein. *Rna*, 19(11):1537–1551, 2013.

C Anders Olson, Nicholas C Wu, and Ren Sun. A comprehensive biophysical description of pairwise epistasis throughout an entire protein domain. *Current biology*, 24(22):2643–2651, 2014.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Xi Chen, John Canny, Pieter Abbeel, and Yun S Song. Evaluating protein transfer learning with tape. *Advances in neural information processing systems*, 32:9689, 2019.

Donatas Repecka, Vykintas Jauniskis, Laurynas Karpus, Elzbieta Rembeza, Irmantas Rokaitis, Jan Zrimec, Simona Poviloniene, Audrius Laurynenas, Sandra Viknander, Wissam Abuajwa, et al. Expanding functional protein sequence spaces using generative adversarial networks. *Nature Machine Intelligence*, 3(4):324–333, 2021.

Adam J Riesselman, John B Ingraham, and Debora S Marks. Deep generative models of genetic variation capture the effects of mutations. *Nature methods*, 15(10):816–822, 2018.

Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C Lawrence Zitnick, Jerry Ma, et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15), 2021.

Philip A Romero and Frances H Arnold. Exploring protein fitness landscapes by directed evolution. *Nature reviews Molecular cell biology*, 10(12):866–876, 2009.

William P Russ, Matteo Figliuzzi, Christian Stocker, Pierre Barrat-Charlaix, Michael Socolich, Peter Kast, Donald Hilvert, Remi Monasson, Simona Cocco, Martin Weigt, et al. An evolution-based model for designing chorismate mutase enzymes. *Science*, 369(6502):440–445, 2020.

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.

Martin Steinegger and Johannes Söding. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, 35(11):1026–1028, 2017.

Zachary Wu, SB Jennifer Kan, Russell D Lewis, Bruce J Wittmann, and Frances H Arnold. Machine learning-assisted directed protein evolution with combinatorial libraries. *Proceedings of the National Academy of Sciences*, 116(18):8852–8858, 2019.

Kevin K Yang, Zachary Wu, and Frances H Arnold. Machine-learning-guided directed evolution for protein engineering. *Nature methods*, 16(8):687–694, 2019.

A APPENDIX

A.1 TABLES AND FIGURES

Table 2: Measurements of homology and epistasis of protein fitness prediction tasks. We consider datasets with a higher number of sequences and sequence clusters to be higher homology, because this indicates a large number and diversity of evolutionarily related proteins. We consider datasets with a lower the Spearman of the additive model to be more epistatic because this means that mutational effect cannot be predicted as well by adding the effects of single mutations.

Dataset	Homology [# sequences]	Homology [# clusters]	Epistasis [additive spearman]
PABP	152041	3866	0.90
GFP	806	23	0.59
AAV	2211	123	0.41
GB1	131	8	0.25

A.2 MODEL TRAINING AND HYPER-PARAMETER DETAILS

All models that use gf-tuning or wildtype residual regression were trained for 5 epochs, all models that used a regression head were trained for 20 epochs. All deep learning models used an Adagrad optimizer (Duchi et al., 2011). Augmented models were fit using ridge regression in SciKit-learn [1].

For each model in each setting, we obtain results using 4 different hyper-parameter settings (4 different learning rates for deep learning models ($1e{-}6, 1e{-}5, 1e{-}4, 1e{-}3$), 4 different ridge regularization parameters for augmented models $(1, 4, 16, 64)$, and average results over 4 different seeds (that result in 4 different n-sized training sets). The data points in our plots give the highest average Spearman correlation (across the 4 seeds) for each model in each task setting out of the 3 hyper-parameter settings tested.

For models that use gf-tuning, we use an automatic procedure to tune the α hyperparameter from Equation 5. At the start of gf-training, we measure the loss function $\mathcal{L}(\mathcal{D}_f)$ on the assay labeled data for varying values of α in the set $\{0.0625, 0.125, 0.25, .5, 1, 2, 4, 8, 16, 32, 64\}$. We chose the value of α that minimizes $\mathcal{L}(\mathcal{D}_f) - .1log_2(\alpha)$, which encourages higher values of α that lead to lower training losses. We found this procedure to work well for selecting α across tasks.

When applying evolutionary finetuning, we train for 1 epoch and save 6 checkpoints at evenly spaced intervals. Our pipeline then chooses the checkpoint that has the highest spearman corrlation on the few-shot assay labeled training set for further finetuning. In the low homology setting, we apply gf-tuning to ProGen without additional evolutionary finetuning and consider an ablation on ProGen's pretraining by applying gf-tuning to a randomly initializing an LM.

A.3 FITNESS PREDICTION WITH BLOSUM62

BLOSUM62 matrices predict the log-odds ratio for every possible substitution at each position using a look-up table of how frequent different mutation are in nature. The BLOSUM62 matrix $B \in \mathbb{R}^{|V|x|V|}$ has a value at $B_{i,j}$ proportional to the log-likelihood of a mutation of residue i being mutated to residue j, where the diagonal values of B are proportional to the log-likelihoods of each residue being retained. For each mutant position, we take the row of the BLOSUM matrix corresponding to the wildtype residue, and take the difference of the column for the mutant residue and the column for the wild-type residue. We then sum these scores to get a prediction.

A.4 DATASET DETAILS

- PABP - High homology, low epistasis. PABP contains functional values for single and double mutations from wildtype.

[1] `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html`

- AAV - High homology, high epistasis. AAV contains many proteins that are 10 or more mutations away from wildtype. We omit sequences that contain deletions from training and testing to simplify modeling.
- GFP - Low homology, low epistasis. We use the commonly benchmark test set used as part of the TAPE (Rao et al., 2019) which contains only sequences with more than 3 mutations from wildtype. In the random split setting, we randomly sample sequences from the TAPE training set split. In the single to multiples setting, we allow training on single mutations from wildtype only.
- GB1 - Low homology, high epistasis. GB1 contains fitness values for all possible permutations of residues at four sequence positions, with the rest of the sequence held constant.