Sparse Autoencoders for Low-N Protein Function Prediction and Design

Darin Tsui
Georgia Institute of Technology
darint@gatech.edu

Kunal Talreja
Georgia Institute of Technology
ktalreja6@gatech.edu

Amirali Aghazadeh
Georgia Institute of Technology
amiralia@gatech.edu

Abstract

Predicting protein function from amino acid sequence remains a central challenge in data-scarce (low-N) regimes, limiting machine learning–guided protein design when only small amounts of assay-labeled sequence-function data are available. Protein language models (pLMs) have advanced the field by providing evolutionary-informed embeddings and sparse autoencoders (SAEs) have enabled decomposition of these embeddings into interpretable latent variables that capture structural and functional features. However, the effectiveness of SAEs for low-N function prediction and protein design has not been systematically studied. Herein, we evaluate SAEs trained on fine-tuned ESM2 embeddings across diverse fitness extrapolation and protein engineering tasks. We show that SAEs, with as few as 24 sequences, consistently outperform or compete with their ESM2 baselines in fitness prediction, indicating that their sparse latent space encodes compact and biologically meaningful representations that generalize more effectively from limited data. Moreover, steering predictive latents exploits biological motifs in pLM representations, yielding top-fitness variants in 83% of cases compared to designing with ESM2 alone.

1 Introduction

Machine learning (ML)–guided protein engineering seeks to predict and optimize protein function by leveraging evolutionary information and assay-labeled sequence data to model the underlying sequence–function landscape [1–3]. In practice, however, ML models are often constrained by the scarcity of experimental data. Functional assays are costly and time-consuming, so only a small number of variants (low-N) can typically be characterized, creating a fundamental bottleneck for ML-guided design [4–6].

Protein language models (pLMs), trained on large evolutionary sequence datasets, provide embeddings that achieve state-of-the-art performance in zero-shot function prediction [7–9]. These embeddings are widely believed to capture amino acid interactions underlying protein function [10–12], yet they remain difficult to interrogate. More recently, sparse autoencoders (SAEs) have emerged as a powerful interpretability framework, factorizing pLM embeddings into sparse, biologically meaningful latent variables. In high-N regimes (e.g., $N > 800$ labeled sequences), these latents have been shown to align with structural and functional motifs [13–16] and can be steered to design sequences with targeted functional properties [17–19]. Despite these advances, the function prediction and steering performance of SAEs in realistic data-scarce (low-N) settings has not been systematically evaluated. *We hypothesize that the sparse latent space of SAEs, originally introduced as a strategy to*

Published in the AI for Science Workshop (NeurIPS 2025).

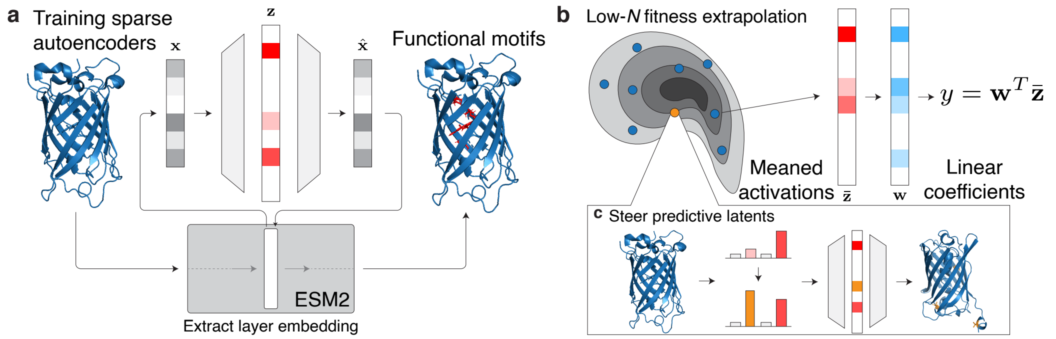

Figure 1: **Overview of downstream low-N tasks for SAEs.** **a**, We train SAEs on the layer embeddings of ESM2. By projecting the model embedding $\mathbf{x}$ to the latent representation $\mathbf{z}$, and reconstructing the model embedding as $\hat{\mathbf{x}}$, the activations in $\mathbf{z}$ correspond to specific biological motifs. **b**, In low-N fitness extrapolation, a linear probe is trained on top of the SAE's latent space to predict protein fitness from N many training sequences. **c**, Using the learned linear probe weights, we steer predictive latents to design highly-functional variants.

enhance interpretability, also encodes compressed and regularized representations that enable accurate fitness prediction and effective protein design from limited data. To test this, we reposition SAEs from proof-of-concept interpretability tools to actionable predictors and design engines, evaluating their performance on downstream protein engineering tasks under low-N conditions. Specifically, we assess their utility across diverse fitness extrapolation challenges that reflect real-world design constraints, and we further examine their ability to design high-functioning variants through latent steering. Our main contributions are as follows:

- We train SAEs on fine-tuned ESM2 embeddings across five proteins with diverse functions.
- We show that SAEs, with as few as 24 sequences, outperform their ESM2 baselines in 58% of fitness extrapolation tasks, while maintaining comparable performance in the remainder.
- We demonstrate that steering SAEs along their most predictive latents produces a diverse pool of highly functional variants, including the top fitness variants in 83% of cases, compared to designing with ESM2 alone.
- We analyze the best-performing steered variants in green fluorescent protein (GFP) and the IgG-binding domain of protein G (GB1), uncovering biologically meaningful motifs that SAEs exploit for steering. All codes and data are available on our GitHub repository `https://github.com/amirgroup-codes/LowNSAE`.

2 Sparse Autoencoders (SAE)

SAEs are autoencoders designed to learn meaningful representations of model embeddings in their latent space (Fig. 1a). We use SAEs with TopK activation [20] to enforce sparsity in the latent space. Given the model embedding $\mathbf{x} \in \mathbb{R}^{d_{\text{model}} \times L}$, where L is the sequence length and d_{model} is the embedding dimension, the encoder maps $\mathbf{x}$ to the SAE latent representation $\mathbf{z} \in \mathbb{R}^{d_{\text{SAE}} \times L}$ via:

$$\mathbf{z} = \text{TopK}(\mathbf{W}_{\text{enc}}(\mathbf{x} - \mathbf{b}_{\text{pre}})), \tag{1}$$

where $\mathbf{W}_{\text{enc}} \in \mathbb{R}^{d_{\text{SAE}} \times d_{\text{model}}}$ are the encoder weights and $\mathbf{b}_{\text{pre}} \in \mathbb{R}^{d_{\text{model}} \times L}$ is a bias term. The TopK function is applied column-wise to the resulting matrix, keeping only the k largest activations for each of the L sequence positions and setting all other values to zero. The decoder then reconstructs the input $\mathbf{x}$ from $\mathbf{z}$ as:

$$\hat{\mathbf{x}} = \mathbf{W}_{\text{dec}}\mathbf{z} + \mathbf{b}_{\text{pre}}, \tag{2}$$

where $\mathbf{W}_{\text{dec}} \in \mathbb{R}^{d_{\text{model}} \times d_{\text{SAE}}}$ are the decoder weights. As illustrated in Fig. 1a, where $L = 1$ for simplicity, the activations in $\mathbf{z}$ have been shown to correspond to biological motifs [13, 14].

During training, SAEs minimize both mean squared error and an auxiliary loss. The mean squared error between the original embedding $\mathbf{x}$ and its reconstruction $\hat{\mathbf{x}}$ is defined as $\mathcal{L}_{\text{MSE}} = \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2$.

To reduce the number of dead latents, defined as latents that never activate [20], an auxiliary loss is included. Given the original reconstruction loss $\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}}$, the auxiliary loss is defined as $\mathcal{L}_{\text{aux}} = \|\mathbf{e} - \hat{\mathbf{e}}\|_2^2$, where $\hat{\mathbf{e}}$ is found by multiplying the decoder matrix by the top-k_{aux} latents in $\mathbf{z}$, where k_{aux} is a hyperparameter. The total SAE training objective, $\mathcal{L}_{\text{SAE}}$, is a weighted sum of these two losses:

$$\mathcal{L}_{\text{SAE}} = \mathcal{L}_{\text{MSE}} + \alpha\mathcal{L}_{\text{aux}},$$

where α is also a hyperparameter. This joint objective enables SAEs to not only reconstruct the original model embeddings faithfully, but also maximize the number of biologically interpretable latents.

3 SAEs for Low-N Fitness Extrapolation

In this section, we first detail the datasets used and how we trained our SAEs. Then, we rigorously evaluate the ability of SAEs to generalize to unseen variants under various low-N regimes (Fig. 1b). To capture the challenges faced in real-world design settings, we define five distinct fitness extrapolation tasks that stress different aspects of the sequence–function landscape: random, position, mutation, regime, and score extrapolation.

3.1 Datasets and SAE Training Details

Datasets. We evaluated our SAEs on six deep mutational scanning (DMS) assays from ProteinGym [21], spanning five distinct proteins (Table 1). These proteins were selected to ensure robust evaluation across a variety of functions. Additionally, these DMS assays also contain multipoint mutations, which are crucial for our fitness extrapolation tasks (see Section 3.2).

Table 1: Summary of DMS assays used.

DMS	Description	Function Tested	Variants	MSA Sequences
GFP_AEQVI_Sarkisyan [22]	Green fluorescent protein	Fluorescence	51,714	396
SPG1_STRSG_Olson [23]	IgG-binding domain of protein G	Binding	536,962	44
SPG1_STRSG_Wu [24]	IgG-binding domain of protein G	Binding	149,360	3,109
DLG4_HUMAN_Faure [25]	Third PDZ domain of PSD95	Yeast growth	6,976	25,338
GRB2_HUMAN_Faure [25]	C-terminal SH3 domain of GRB2	Yeast growth	63,366	33,228
F7YBW8_MESOW_Ding [26]	Antitoxin ParD3	Growth enrichment	7,922	38,613

Training. For each DMS assay, we trained a unique SAE on a fine-tuned ESM2-650M model [7]. Each model was fine-tuned on multiple sequence alignment (MSA) sequences from Table 1 using LoRA adapters, where the MSA sequences were obtained from ProteinGym. Embeddings $\mathbf{x}$ to train the SAE were then obtained by passing the MSA sequences through the fine-tuned model. Following [14], we chose to extract embeddings from layer 24, and set $d_{\text{SAE}} = 4096$, $k = 128$, $\alpha = 1/32$, and $k_{\text{aux}} = 256$. For more details on training, see Appendices A.1 and A.2.

3.2 Experimental Setup

Low-N Regimes. To evaluate the performance of our SAEs and ESM2 in the low-N regime, we first created four distinct N sizes to train a supervised model on top of the SAE latent space and ESM2 embeddings, respectively, to predict fitness: $N \in [8, 24, 96, 384]$. These sizes correspond to standard plate-well sizes used in protein engineering experiments [4].

Fitness Extrapolation Tasks. For each of our DMS assays, we designed five different fitness extrapolation tasks based on ref. [27] to test the ability of our SAE and ESM2 to generalize to unseen variants (see Fig. 2):

1. **Random extrapolation**: We randomly sampled N sequences from the DMS for training and validation, with 10% of the DMS held out as a test set (Fig. 2b).
2. **Mutation extrapolation**: We randomly designated 80% of all possible mutations as training mutations (Fig. 2c). We sampled N sequences for training that only had training mutations. The other 20% of mutations were held out as a test set.

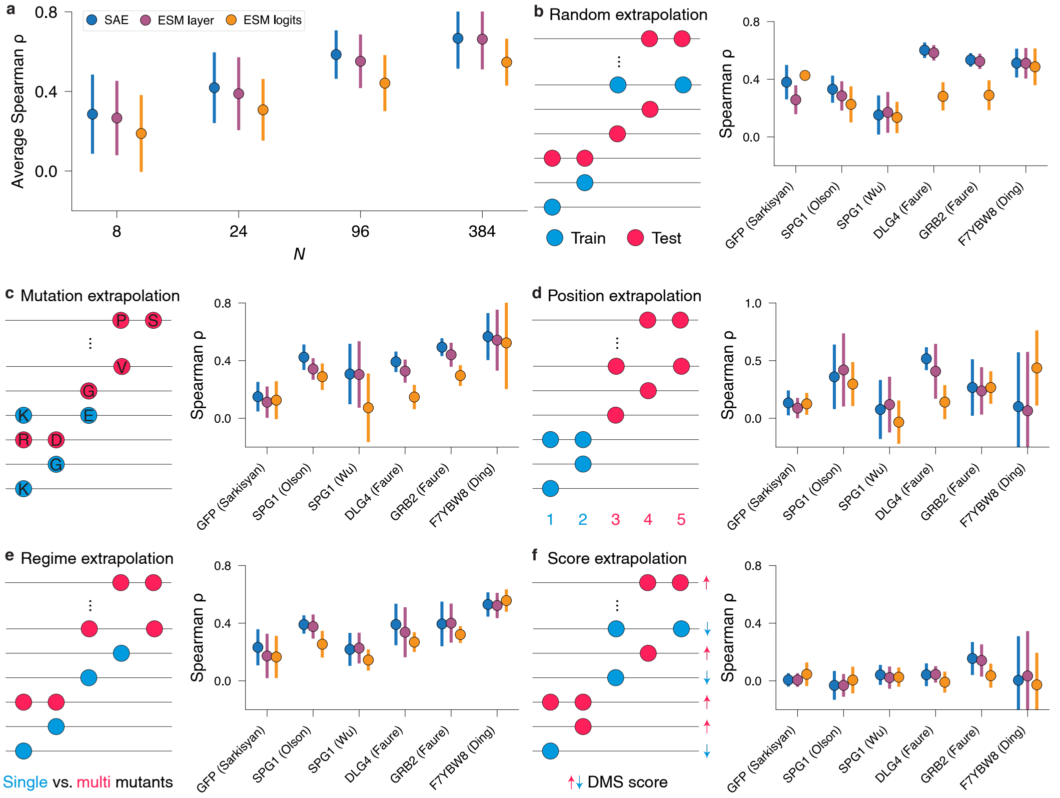

Figure 2: **Comparative performance of low-N fitness extrapolation in SAEs versus ESM2. a**, Average correlations of SAE, ESM layer, and ESM logits across all low-N regimes on random extrapolation. Fitness extrapolation correlations over each DMS assay using $N = 24$ sequences across **b**, random, **c**, mutation, **d**, position, **e**, regime, and **f**, score extrapolations. Error bars represent the standard deviation across nine independent runs with different random seeds.

3. **Position extrapolation**: We randomly designated 80% of amino acid positions as training positions (Fig. 2d). We then sampled N sequences for training that had mutations exclusively at the training positions. The other 20% of positions were held out as a test set.

4. **Regime extrapolation**: For DMS assays containing only single and double mutations, we trained on N single mutations and tested on all double mutations. For DMS assays with more than two mutations, we trained on N sequences drawn from single and double mutations, and tested on all sequences with more than two mutations (Fig. 2e).

5. **Score extrapolation**: We trained on N sequences with a fitness score lower than the wildtype and tested on all sequences with a fitness score higher than the wildtype (Fig. 2f).

Linear Probes. For each of these extrapolations, we trained a linear probe with Ridge regression on top of the SAE latent space. To benchmark against the performance of ESM2 without help from an SAE, we also trained linear probes on the ESM2 layer 24 embedding and ESM2 logits. For brevity, we refer to these methods as 1) SAE, 2) ESM layer, and 3) ESM logits. Following [14], we mean-pool the input to the linear probe over the respective embedding dimension. Formally, in SAEs, we denote $\bar{\mathbf{z}} \in \mathbb{R}^{d_{\text{SAE}}}$ to be the meaned activations of the latent space and $\mathbf{w} \in \mathbb{R}^{d_{\text{SAE}}}$ to be weights of the linear probe. The linear probe then computes the fitness score y via: $y = \mathbf{w}^T\bar{\mathbf{z}}$ (Fig. 1b). For all tasks, we set aside a portion of the training sequences to be used as validation. Further details are provided in Appendix A.3.

To ensure the robustness of our results, we ran a total of nine trials for each extrapolation. For random, position, and mutation extrapolations, we used three different random seeds to create the test set. For each of these test sets, we then randomly sampled N training sequences three times. For the regime and score extrapolations, where the test set is deterministic, we randomly sampled N training sequences nine times.

3.3 SAEs Achieve Improved Generalization to Unseen Variants Compared to ESM2

Fig. 2a shows the average Spearman correlation of SAE, ESM layer, and ESM logits under random extrapolation across all low-N regimes, while Table 2 breaks down results by DMS assay. SAEs achieve higher correlations than their ESM2 counterparts in 67% of random extrapolation experiments, and across all low-N regimes and fitness extrapolation tasks (Appendix B), they outperform in 58% of cases. These results suggest that SAE latents capture more biologically meaningful patterns and enable more reliable generalization to unseen variants. Among the different extrapolation settings, position, regime, and score extrapolation emerge as the most challenging, since they require the model to capture structural context and nonlinear interactions underlying protein function. Notably, SAEs outperform their ESM2 counterparts in 69% of position and regime extrapolation tasks, suggesting that their sparse latent space encodes fundamental biological constraints. We additionally notice that SAEs are not able to generalize to unseen variants when ESM2 does a poor job, such as in score extrapolation. This is not surprising, as intuitively, SAEs are trained to reorganize the information encoded by ESM2 into a more compact and disentangled representation. Thus, when the underlying pLM provides a limited predictive signal, the bottleneck in performance lies in the pLM rather than in the SAE.

Table 2: Average Spearman ρ across all low-N regimes under random extrapolation across each DMS assay. A full summary of results for other fitness extrapolation tasks is located in Appendix B.

Method	**DMS**	$N = 8$ ↑	$N = 24$ ↑	$N = 96$ ↑	$N = 384$ ↑
SAE	GFP_AEQVI_Sarkisyan	0.26 ± 0.15	0.38 ± 0.12	**0.56 ± 0.05**	**0.67 ± 0.01**
	SPG1_STRSG_Olson	0.16 ± 0.17	**0.33 ± 0.09**	**0.67 ± 0.03**	**0.82 ± 0.01**
	SPG1_STRSG_Wu	0.12 ± 0.16	0.15 ± 0.14	**0.34 ± 0.04**	0.35 ± 0.03
	DLG4_HUMAN_Faure	**0.45 ± 0.16**	**0.60 ± 0.05**	**0.67 ± 0.03**	**0.76 ± 0.02**
	GRB2_HUMAN_Faure	**0.31 ± 0.15**	**0.53 ± 0.05**	**0.63 ± 0.02**	**0.73 ± 0.01**
	F7YBW8_MESOW_Ding	**0.42 ± 0.19**	**0.51 ± 0.10**	0.64 ± 0.02	0.68 ± 0.02
ESM layer	GFP_AEQVI_Sarkisyan	0.21 ± 0.13	0.26 ± 0.10	0.46 ± 0.07	0.61 ± 0.01
	SPG1_STRSG_Olson	**0.17 ± 0.22**	0.28 ± 0.10	0.64 ± 0.02	0.81 ± 0.01
	SPG1_STRSG_Wu	**0.13 ± 0.15**	**0.17 ± 0.14**	0.31 ± 0.05	**0.36 ± 0.07**
	DLG4_HUMAN_Faure	0.40 ± 0.16	0.58 ± 0.05	0.66 ± 0.05	**0.76 ± 0.02**
	GRB2_HUMAN_Faure	0.28 ± 0.14	0.52 ± 0.05	0.60 ± 0.03	0.73 ± 0.02
	F7YBW8_MESOW_Ding	0.40 ± 0.15	0.51 ± 0.11	**0.65 ± 0.02**	**0.70 ± 0.01**
ESM logits	GFP_AEQVI_Sarkisyan	**0.31 ± 0.12**	**0.43 ± 0.03**	0.49 ± 0.05	0.57 ± 0.03
	SPG1_STRSG_Olson	0.12 ± 0.13	0.23 ± 0.13	0.42 ± 0.02	0.55 ± 0.01
	SPG1_STRSG_Wu	0.01 ± 0.13	0.14 ± 0.11	0.21 ± 0.08	0.30 ± 0.04
	DLG4_HUMAN_Faure	0.17 ± 0.17	0.28 ± 0.10	0.47 ± 0.05	0.60 ± 0.04
	GRB2_HUMAN_Faure	0.14 ± 0.10	0.29 ± 0.10	0.41 ± 0.07	0.59 ± 0.02
	F7YBW8_MESOW_Ding	0.38 ± 0.25	0.49 ± 0.13	0.65 ± 0.03	0.67 ± 0.02

Fig. 2b-f further illustrates the performance of SAE, ESM layer, and ESM logits across all extrapolation tasks with $N = 24$ sequences. We designated this as the smallest low-N regime for reliable extrapolation, with the SAE achieving an average correlation of 0.42. Across nearly all tasks, SAEs either match or outperform both ESM layers and ESM logits, highlighting their robustness and effectiveness in diverse low-N extrapolation settings. For additional results, see Appendix B.

4 SAEs for Low-N Protein Engineering

After demonstrating that SAEs are able to generalize to unseen variants, we then looked to assess their performance in generating high-functioning proteins (Fig. 1c). To explicitly optimize for function, we implemented a modified version of feature steering [28], which leverages the predictive scores from the linear probes. For all experiments, we used the linear probes trained on $N = 24$ sequences.

Table 3: Protein engineering results using $N = 24$ training sequences. All variants were constrained to a maximum of five mutations away from the wild type.

Method	DMS	Mean fitness ↑	Max fitness ↑	Top 10% fitness ↑	Top 20% fitness ↑
SAE	GFP_AEQVI_Sarkisyan	**3.49** ± 0.44	**3.87**	**3.75** ± 0.08	3.71 ± 0.07
	SPG1_STRSG_Olson	**2.75** ± 1.29	**4.53**	**4.47** ± 0.04	**4.29** ± 0.24
	SPG1_STRSG_Wu	**0.67** ± 0.94	**3.89**	**2.70** ± 0.79	**2.18** ± 0.76
	DLG4_HUMAN_Faure	**0.39** ± 0.22	**0.68**	**0.66** ± 0.02	**0.62** ± 0.05
	GRB2_HUMAN_Faure	**-0.10** ± 0.48	**0.67**	**0.59** ± 0.07	**0.49** ± 0.12
	F7YBW8_MESOW_Ding	0.81 ± 0.33	1.16	**1.15** ± 0.01	**1.13** ± 0.03
ESM layer	GFP_AEQVI_Sarkisyan	3.29 ± 0.66	3.72	3.71 ± 0.01	3.70 ± 0.01
	SPG1_STRSG_Olson	0.29 ± 1.95	3.19	2.74 ± 0.35	2.44 ± 0.39
	SPG1_STRSG_Wu	0.08 ± 0.30	1.69	0.81 ± 0.63	0.41 ± 0.60
	DLG4_HUMAN_Faure	-0.10 ± 0.41	0.63	0.45 ± 0.14	0.36 ± 0.13
	GRB2_HUMAN_Faure	-0.40 ± 0.39	0.30	0.24 ± 0.05	0.17 ± 0.10
	F7YBW8_MESOW_Ding	**1.06** ± 0.10	**1.16**	1.15 ± 0.02	1.12 ± 0.03
ESM logits	GFP_AEQVI_Sarkisyan	3.13 ± 0.86	3.76	3.73 ± 0.02	**3.72** ± 0.02
	SPG1_STRSG_Olson	-1.11 ± 2.21	2.27	2.05 ± 0.42	1.56 ± 0.60
	SPG1_STRSG_Wu	0.15 ± 0.37	1.69	1.13 ± 0.33	0.76 ± 0.50
	DLG4_HUMAN_Faure	-0.15 ± 0.38	0.53	0.36 ± 0.14	0.27 ± 0.13
	GRB2_HUMAN_Faure	-0.26 ± 0.44	0.58	0.44 ± 0.09	0.32 ± 0.14
	F7YBW8_MESOW_Ding	1.05 ± 0.06	1.12	1.11 ± 0.01	1.11 ± 0.01
Random	GFP_AEQVI_Sarkisyan	3.36 ± 0.70	3.75	3.72 ± 0.02	3.70 ± 0.02
	SPG1_STRSG_Olson	-1.25 ± 2.63	3.05	2.40 ± 0.49	1.99 ± 0.55
	SPG1_STRSG_Wu	0.33 ± 0.76	3.61	2.27 ± 0.82	1.36 ± 1.11
	DLG4_HUMAN_Faure	-0.34 ± 0.43	0.35	0.32 ± 0.04	0.24 ± 0.10
	GRB2_HUMAN_Faure	-0.96 ± 0.38	0.16	-0.13 ± 0.18	-0.30 ± 0.22
	F7YBW8_MESOW_Ding	0.66 ± 0.37	1.16	1.11 ± 0.03	1.06 ± 0.06

4.1 Experimental Setup

For feature steering, we first identified the most predictive latent features by examining the largest-magnitude weights of the linear probe. For each of these high-impact latents, we increased its activation by a hyperparameter multiplier. The updated latent representation was then passed through the SAE decoder and fed into ESM2 to design a new sequence. We optimized the multiplier by selecting the value that yielded the highest predicted fitness score from the linear probe. Similar to fitness extrapolation, to benchmark against the performance of ESM2, we also designed sequences using the linear probes trained on the ESM layer and ESM logits via simulated annealing, following the procedure detailed in [27]. Additionally, we included a random baseline by generating sequences with a random number of mutations and amino acid substitutions. Further details on our experimental setup are provided in Appendix A.4.

We used a multi-layer perceptron (MLP) trained on the DMS assays to evaluate the fitness of our designed variants (see Appendix A.5). To constrain our search space and ensure the MLP's predictions are a good proxy for experimental fitness, we limited all designed variants to a maximum of five mutations away from the wildtype. A notable exception to this setup is the SPG1_STRSG_Wu DMS: this assay provides ground-truth fitness values for all possible combinatorial variants over four positions. Therefore, we directly used the fitness values from SPG1_STRSG_Wu to evaluate our designed variants and limited our maximum number of mutations to four. A total of 50 variants were designed per DMS assay.

4.2 SAEs Design High-functional Variants and Capture Biological Motifs

Table 3 shows the performance of all methods in generating highly-functional variants. Across all metrics and DMS assays, SAEs outperform their ESM2 counterparts in 88% of cases. More specifically, our SAE steering approach designed the top fitness variants in five out of the six DMS assays. Additionally, SAE steering designed the highest top 10% fitness variants across all DMS assays and the highest top 20% variants in five out of six DMS assays. This suggests that SAE steering is not only capable of discovering the single top-performing variant, but also is capable of generating a diverse pool of highly functional variants.

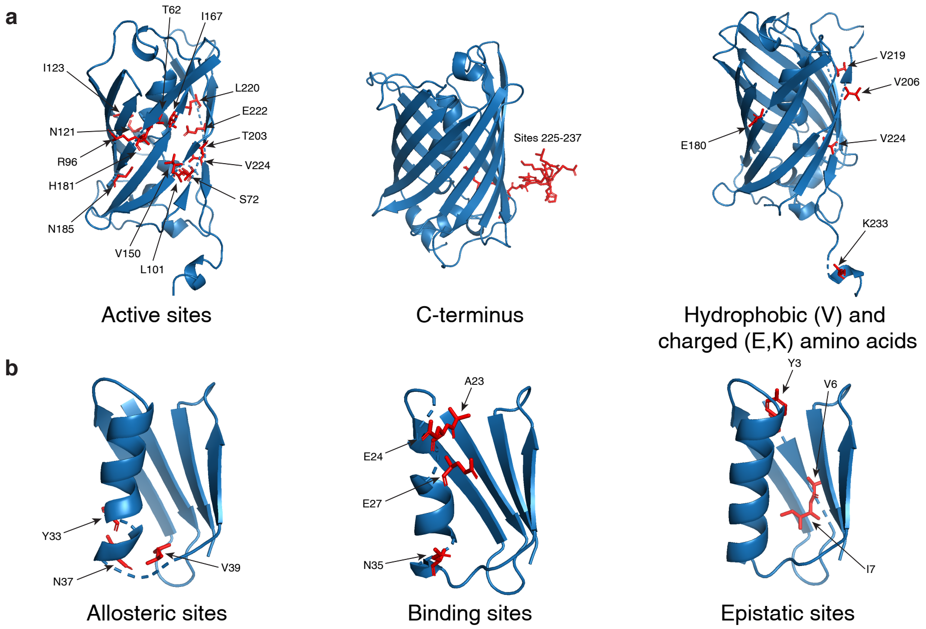

Figure 3: **Analysis of the top-performing steered variants.** **a**, Our analysis of the top-performing GFP variants revealed that steering activated latent features corresponding to key biological motifs, including active site amino acids, the C-terminus, and hydrophobic and charged amino acids. **b**, GB1 variants activated latent features associated with allosteric, binding, and epistatic sites.

To better understand why feature steering designs high-functional variants, we performed a qualitative analysis on the top-performing variants for the green fluorescent protein (GFP) and the IgG-binding domain of protein G (GB1). We identified the ten latent dimensions most strongly associated with changes in fitness. We also analyzed any shifts in their activation patterns between the wildtype and the designed variant. Finally, we projected these activated residues onto the variant's structure (generated via AlphaFold3 [29]) to identify amino acid concentrations and infer their biological relevance. Further details are provided in Appendix A.6.

Our analysis revealed that SAEs preferentially activated latent features associated with known biological motifs. For instance, in GFP, this includes latents activating on active site amino acids, which are crucial for fluorescence [30], and the C-terminus, a disordered region also known to affect fluorescence (Fig. 3a) [31]. We also found that steering favored latent features corresponding to hydrophobic and charged amino acids, which are essential for maintaining the protein's structural stability [32]. Similarly, our analysis of the top-performing variants in GB1 highlighted key functional regions (Fig. 3b). The top latents were most active at sites that are allosteric [25], which modulate protein function, or binding, which directly interact with the protein IgG [23]. Furthermore, latents activated on epistatic sites [23], demonstrating the SAE's ability to design variants in the presence of complex, non-additive mutations. These findings collectively demonstrate that SAEs, even without explicit training, successfully learn and leverage fundamental biological principles to design new variants.

5 Discussion

In this paper, we demonstrated that sparse autoencoders (SAEs) can serve as a powerful tool for low-N tasks. We demonstrated that SAEs consistently outperform their ESM2 counterparts in a variety of low-N fitness extrapolation tasks and are highly effective for generating novel, high-fitness protein variants. Our work expands the biologist's toolkit for resource-constrained applications and takes the first step toward extracting actionable biological knowledge from pLMs.

Sparsity in SAEs. Although SAEs introduce a larger dimensionality to the linear probes, they consistently outperform their ESM2 counterparts in low-N fitness extrapolation and protein engineering tasks. At first glance, this appears counterintuitive: in low-N regimes, simpler models

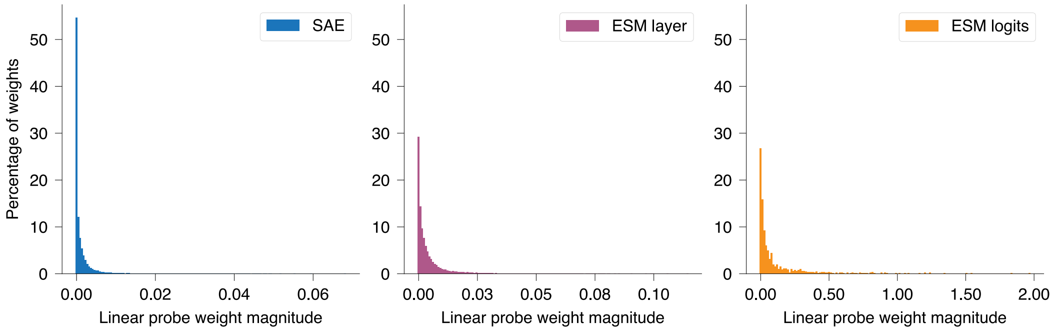

Figure 4: **Sparsity in SAEs underlies improved low-N performance.** Histogram of linear probe weight magnitudes for the SAE latent space, ESM layer embeddings, and ESM logits. Given the top 5% of weights by magnitude, SAE weights explain $37 \pm 9\%$ of the variance, compared to $27 \pm 4\%$ in ESM layer weights and $25 \pm 12\%$ in ESM logits weights. See Appendix A.7 for details on the visualization procedure.

with fewer parameters are typically less prone to overfitting. We attribute the superior performance of SAEs to their ability to compress biologically relevant information into a *sparse* latent space (Fig. 4). To quantify this effect, we measure the variance explained by the magnitude of the top 5% of probe weights. Under this definition, SAE weights explain $38 \pm 9\%$ of the variance, whereas ESM layer and ESM logits weights explain $28 \pm 3\%$ and $31 \pm 17\%$, respectively. These results suggest that SAEs compress information from ESM2 into a more compact and disentangled representation, where the biological relevant signal is concentrated in a select few latents. In the low-N regime, this compression is particularly advantageous: sparser models are less prone to overfitting and thus generalize more effectively from limited experimental data. This also allows each high-impact latent to disentangle which amino acids contribute to fitness, enhancing the effectiveness of our steering approach.

Low-N Performance Variability. Across fitness extrapolation tasks, we observe relatively high standard deviation in all methods, This is not surprising, given that the linear probes are trained in the low-N regime, making the performance sensitive to which sequences are sampled. Despite this variability, we emphasize that SAEs outperform their ESM2 counterparts in 58% of cases on average, indicating a consistent advantage. Moreover, SAEs tend to design more high-functioning variants, suggesting that their sparse latent space captures a more informative view of the functional landscape.

Limitations and Future Work. Our work analyzes the performance of SAEs across a wide range of proteins and molecular functions. However, our evaluation could be extended to other proteins with clinically relevant functions such as antibiotic resistance and viral replication, which may open new avenues for therapeutic design. We also observed that SAE performance is strongly influenced by the number of MSA sequences available for training. For example, proteins such as DLG4, GRB2, and F7YBW8, which consistently achieved high fitness extrapolation correlations, each had more than 20,000 MSA sequences. Future work should explore strategies for robust SAE training when MSAs are shallow or unavailable.

To validate the designed variants, our protein engineering experiments rely on trained fitness models *in silico*. While this is a good proxy for fitness, further validation through wet lab experiments is necessary to verify the function of designed variants. Nevertheless, our results on the SPG1_STRSG_Wu DMS assay, for which ground-truth fitness values are available for all combinatorial variants, demonstrate that our SAE steering approach still produces the highest-fitness variants compared to other methods. Additionally, we clarify that successful protein design requires not only highly-functional, but also highly-stable variants. Certain mutations that promote function may also destabilize the protein [33, 34], reinforcing the need for wet lab experiments to test designs. A promising future direction is to couple SAE steering with physics-based tools such as Rosetta [35] to jointly optimize for both function and stability.

Lastly, in our protein engineering experiments, we constrained variants to be a maximum of five mutations away from the wildtype. Beyond this radius, we found it difficult to design highly-functional sequences using just one predictive latent. Additionally, because predictive information is concentrated in a small number of latents, we restricted the amount of variants designed to 50 per DMS assay. Future work towards expanding the design space could involve steering multiple latents at once, which could enable both further mutational exploration and more diverse pools of functional variants. However, we note that in silico evaluation tools become less reliable the further away variants from the wildtype are, reinforcing the need for wet lab validation.

6 Acknowledgment

This research was supported by the National Science Foundation (NSF) Graduate Research Fellowship Program (GRFP), the Parker H. Petit Institute for Bioengineering and Biosciences (IBB) interdisciplinary seed grant, the Institute of Matter and Systems (IMS) Exponential Electronics seed grant, Georgia Tech Undergraduate Research Opportunities Program, Georgia Tech Research Corporation, and Georgia Institute of Technology start-up funds.

References

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

[2] Pascal Notin, Nathan Rollins, Yarin Gal, Chris Sander, and Debora Marks. Machine learning for functional protein design. *Nature Biotechnology*, 42(2):216–228, 2024.

[3] Kerr Ding, Michael Chin, Yunlong Zhao, Wei Huang, Binh Khanh Mai, Huanan Wang, Peng Liu, Yang Yang, and Yunan Luo. Machine learning-guided co-optimization of fitness and diversity facilitates combinatorial library design in enzyme engineering. *Nature Communications*, 15(1): 6392, 2024.

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

[5] Peter Horvath, Nathalie Aulner, Marc Bickle, Anthony M. Davies, Elaine Del Nery, Daniel Ebner, Maria C. Montoya, Päivi Östling, Vilja Pietiäinen, Leo S. Price, Spencer L. Shorte, Gerardo Turcatti, Carina von Schantz, and Neil O. Carragher. Screening out irrelevant cell-based models of disease. *Nature Reviews Drug Discovery*, 15(11):751–769, 2016.

[6] Zachary Wu, S. B. 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.

[7] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

[8] Thomas Hayes, Roshan Rao, Halil Akin, Nicholas J. Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q. Tran, Jonathan Deaton, Marius Wiggert, Rohil Badkundri, Irhum Shafkat, Jun Gong, Alexander Derry, Raul S. Molina, Neil Thomas, Yousuf A. Khan, Chetan Mishra, Carolyn Kim, Liam J. Bartie, Matthew Nemeth, Patrick D. Hsu, Tom Sercu, Salvatore Candido, and Alexander Rives. Simulating 500 million years of evolution with a language model. *Science*, 387(6736):850–858, 2025.

[9] Pascal Notin, Mafalda Dias, Jonathan Frazer, Javier Marchena-Hurtado, Aidan N. Gomez, Debora Marks, and Yarin Gal. Tranception: Protein fitness prediction with autoregressive transformers and inference-time retrieval. In *Proceedings of the International Conference on Machine Learning*, pages 16990–17017. PMLR, 2022.

[10] Darin Tsui, Aryan Musharaf, Yigit E. Erginbas, Justin S. Kang, and Amirali Aghazadeh. SHAP zero explains biological sequence models with near-zero marginal cost for future queries. *arXiv preprint arXiv:2410.19236*, 2024.

[11] Darin Tsui and Amirali Aghazadeh. On recovering higher-order interactions from protein language models. In *ICLR 2024 Workshop on Generative and Experimental Perspectives for Biomolecular Design*, 2024.

[12] Amirali Aghazadeh, Hunter Nisonoff, Orhan Ocal, David H. Brookes, Yijie Huang, O. Ozan Koyluoglu, Jennifer Listgarten, and Kannan Ramchandran. Epistatic Net allows the sparse spectral regularization of deep neural networks for inferring fitness functions. *Nature Communications*, 12(1):5225, 2021.

[13] Elana Simon and James Zou. InterPLM: Discovering interpretable features in protein language models via sparse autoencoders. *bioRxiv*, pages 2024–11, 2024.

[14] Etowah Adams, Liam Bai, Minji Lee, Yiyang Yu, and Mohammed AlQuraishi. From mechanistic interpretability to mechanistic biology: Training, evaluating, and interpreting sparse autoencoders on protein language models. In *Proceedings of the International Conference on Machine Learning*, 2025.

[15] Thomas Walton, Darin Tsui, Lauren Fogel, Dustin J. E. Huard, Rafael Siqueira Chagas, Raquel L. Lieberman, and Amirali Aghazadeh. GOLF: A generative AI framework for pathogenicity prediction of myocilin OLF variants. *bioRxiv*, pages 2025–06, 2025.

[16] Onkar Gujral, Mihir Bafna, Eric Alm, and Bonnie Berger. Sparse autoencoders uncover biologically interpretable features in protein language model representations. *Proceedings of the National Academy of Sciences*, 122(34):e2506316122, 2025.

[17] Nithin Parsan, David J. Yang, and John Jingxuan Yang. Towards interpretable protein structure prediction with sparse autoencoders. In *ICLR 2025 Workshop on Generative and Experimental Perspectives for Biomolecular Design*, 2025.

[18] Edith Natalia Villegas Garcia. Interpreting and steering protein language models through sparse autoencoders. In *ICLR 2025 Workshop on Generative and Experimental Perspectives for Biomolecular Design*, 2025.

[19] Gerard Boxo Corominas, Filippo Stocco, and Noelia Ferruz. Sparse autoencoders in protein engineering campaigns: Steering and model diffing. In *ICML 2025 Generative AI and Biology (GenBio) Workshop*, 2025.

[20] Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *International Conference on Learning Representations*, 2025.

[21] Pascal Notin, Aaron Kollasch, Daniel Ritter, Lood van Niekerk, Steffanie Paul, Han Spinner, Nathan Rollins, Ada Shaw, Rose Orenbuch, Ruben Weitzman, Jonathan Frazer, Mafalda Dias, Dinko Franceschi, Yarin Gal, and Debora Marks. ProteinGym: Large-scale benchmarks for protein fitness prediction and design. In *Advances in Neural Information Processing Systems*, volume 36, pages 64331–64379, 2023.

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

[23] 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.

[24] Nicholas C. Wu, Lei Dai, C. Anders Olson, James O. Lloyd-Smith, and Ren Sun. Adaptation in protein fitness landscapes is facilitated by indirect paths. *elife*, 5:e16965, 2016.

[25] Andre J. Faure, Júlia Domingo, Jörn M. Schmiedel, Cristina Hidalgo-Carcedo, Guillaume Diss, and Ben Lehner. Mapping the energetic and allosteric landscapes of protein binding domains. *Nature*, 604(7904):175–183, 2022.

[26] David Ding, Ada Y. Shaw, Sam Sinai, Nathan Rollins, Noam Prywes, David F Savage, Michael T. Laub, and Debora S. Marks. Protein design using structure-based residue preferences. *Nature Communications*, 15(1):1639, 2024.

[27] Sam Gelman, Bryce Johnson, Chase Freschlin, Arnav Sharma, Sameer D'Costa, John Peters, Anthony Gitter, and Philip A. Romero. Biophysics-based protein language models for protein engineering. *bioRxiv*, 2024.

[28] Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L. Turner, Callum McDougall, Monte MacDiarmid, Alex Tamkin, Esin Durmus, Tristan Hume, Francesco Mosconi, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. Transformer Circuits Thread, 2024. URL `https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html`.

[29] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J. Ballard, Joshua Bambrick, Sebastian W. Bodenstein, David A. Evans, Chia-Chun Hung, Michael O'Neill, David Reiman, Kathryn Tunyasuvunakool, Zachary Wu, Akvilė Žemgulytė, Eirini Arvaniti, Charles Beattie, Ottavia Bertolli, Alex Bridgland, Alexey Cherepanov, Miles Congreve, Alexander I. Cowen-Rivers, Andrew Cowie, Michael Figurnov, Fabian B. Fuchs, Hannah Gladman, Rishub Jain, Yousuf A. Khan, Caroline M. R. Low, Kuba Perlin, Anna Potapenko, Pascal Savy, Sukhdeep Singh, Adrian Stecula, Ashok Thillaisundaram, Catherine Tong, Sergei Yakneen, Ellen D. Zhong, Michal Zielinski, Augustin Žídek, Victor Bapst, Pushmeet Kohli, Max Jaderberg, Demis Hassabis, and John M. Jumper. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, 630(8016):493–500, 2024.

[30] Jonathan Yaacov Weinstein, Carlos Martí-Gómez, Rosalie Lipsh-Sokolik, Shlomo Yakir Hoch, Demian Liebermann, Reinat Nevo, Haim Weissman, Ekaterina Petrovich-Kopitman, David Margulies, Dmitry Ivankov, David M. McCandlish, and Sarel J. Fleishman. Designed active-site library reveals thousands of functional GFP variants. *Nature Communications*, 14(1):2890, 2023.

[31] Hyong-Kyu Kim and Bong-Kiun Kaang. Truncated green fluorescent protein mutants and their expression in Aplysia neurons. *Brain Research Bulletin*, 47(1):35–41, 1998.

[32] Mats Ormö, Andrew B. Cubitt, Karen Kallio, Larry A. Gross, Roger Y. Tsien, and S. James Remington. Crystal structure of the Aequorea victoria green fluorescent protein. *Science*, 273 (5280):1392–1395, 1996.

[33] Brian K. Shoichet, Walter A. Baase, Ryota Kuroki, and Brian W. Matthews. A relationship between protein stability and protein function. *Proceedings of the National Academy of Sciences*, 92(2):452–456, 1995.

[34] Elizabeth M. Meiering, Luis Serrano, and Alan R. Fersht. Effect of active site residues in barnase on activity and stability. *Journal of Molecular Biology*, 225(3):585–589, 1992.

[35] Julia Koehler Leman and *et al.* Macromolecular modeling and design in Rosetta: recent methods and frameworks. *Nature Methods*, 17(7):665–680, 2020.

A Additional Experimental Details

A.1 Fine-tuning ESM2

We fine-tuned the pre-trained ESM-2-650M model on the MSA of each DMS assay using LoRA (Low-Rank Adaptation) adapters to each layer. For each DMS assay, we loaded its corresponding MSA and masked 15% of the amino acids in each sequence, consistent with ESM2's original masked language modeling objective.

To prevent overfitting, we subsample the MSA by randomly selecting up to 1000 sequences to fine-tune on. The number of fine-tuning epochs was dynamically determined based on the number of sequences used. The epoch schedule was set as follows:

- <100 sequences: 20 epochs
- 100-299 sequences: 10 epochs
- 300-499 sequences: 5 epochs
- 500-799 sequences: 4 epochs
- ≥ 800 sequences: 3 epochs

We fine-tuned the model using the AdamW optimizer with a learning rate of 10^{-4}. We set the hyperparameters of LoRA to be the following: `r=8`, `lora_alpha=16`, `lora_dropout=0.05`, and `bias=None`.

A.2 Training SAEs

We trained a unique SAE for each of the DMS assays. For each DMS assay, we first load in the respective fine-tuned ESM2 model (Appendix A.1). We adapted code from `https://github.com/etowahadams/interprot` [14] to train on embeddings from layer 24. We dynamically set the number of training epochs based on the number of MSA sequences in each assay. The epoch schedule was set as follows:

- <500 sequences: 1000 epochs
- 500-999 sequences: 500 epochs
- 1000-4999 sequences: 100 epochs
- ≥ 5000 sequences: 10 epochs

A.3 Fitness Extrapolation

For all tasks except regime extrapolation, we set aside 10% of the training sequences to use as a validation set and perform a grid search over regularization strengths. In regime extrapolation, we set aside 20% as validation. Since the dataset for F7YBW8_MESOW_Ding only has 166 single and double mutations, we modify the regime split to train on single, double, and triple mutations, and test on datapoints with more than three mutations. Since the DMS for SPG1_STRSG_Wu has only 4 sites with mutations, we also modify the position split to instead take 75% of amino acid positions as training positions and 25% as test.

A full summary of results can be found in Appendix B. In each fitness extrapolation table, we report the absolute value of the Spearman correlation plus the standard deviation across all nine trials.

A.4 Protein Engineering

In this section, we provide additional details on our protein engineering experimental setup. To ensure our trained MLPs can properly score the designed variants, we only design mutations at positions that are present in the DMS assays.

Feature Steering. Given the wildtype sequence, we first pass it through ESM2 to get the layer embeddings, and then pass it through the SAE encoder to get the latent representation $\mathbf{z}$. For the i^{th} predictive latent, we multiply the i^{th} row of $\mathbf{z}$ by a hyperparameter multiplier. The modified latent vector is then passed through the SAE decoder. The resulting vector is fed through the remaining layers of ESM2 to output the logits of the mutated sequence, $\mathbf{x}_{\text{logits,mut}} \in \mathbb{R}^{L\times V}$, where L is the sequence length and V is the vocabulary size.

We compare these new logits to the logits of the wild-type sequence, $\mathbf{x}_{\text{logits,wt}} \in \mathbb{R}^{L\times V}$. For each amino acid position, we calculate the cosine similarity between the respective logit vectors. We only accept a mutation at a given position if the cosine similarity is below 0.98, ensuring that we only mutate amino acids where ESM2 has made a meaningful change.

To find the optimal multiplier, we perform a grid search over values from -3 to 3 with a step size of 0.2. For each multiplier, we use the linear probe to predict the fitness of the resulting sequence. We then select the top 50 unique sequences with the highest predicted fitness. If a sequence has been previously designed, we move to the next highest-scoring sequence to ensure we design 50 unique variants.

Simulated Annealing. We adapt the simulated annealing code from `https://github.com/gitter-lab/metl-pub/tree/main/sim-annealing` [27]. All parameters are left as default. We run simulated annealing over the linear probes trained on the ESM layer and ESM logits. The number of mutations per designed sequence was determined by sampling from the Poisson distribution $\text{Pois}(2) + 1$, ensuring that the maxmimum number of mutations possible is still five. To ensure a fair comparison, we ensured both feature steering and simulated annealing took a comparable amount of time. We set the number of simulated annealing timesteps based on the time required for feature steering to design 50 variants. This was done by first measuring the time needed to complete 1,000 simulated annealing timesteps and then scaling accordingly.

Random. To create the random baseline, we sample from the Poisson distribution of $\text{Pois}(2) + 1$ to determine the number of mutations to make. We then choose the mutated amino acid uniformly at random.

A.5 MLP Training

To create a fitness prediction model for our protein engineering tasks, we trained an MLP for each DMS assay. The MLP is a three-layer feedforward network with ReLU activation functions, taking in a flattened one-hot encoding of the entire protein sequence as input. The network architecture consists of an input layer, a hidden layer with 128 neurons, a second hidden layer with 64 neurons, and a final output layer with a single neuron to predict the fitness score.

For each DMS assay, we split the full data into a training set (80%) and a validation set (20%). We then trained the MLP for up to 1000 epochs using the AdamW optimizer with a learning rate of 10^{-3} and Mean Squared Error (MSE) as the loss function. To prevent overfitting, we employed early stopping with a patience of 10 epochs based on the validation loss.

A.6 Feature Visualization

Given the wildtype sequence, we find the latent representation $\mathbf{z} \in \mathbb{R}^{d_{\text{SAE}}\times L}$ by passing the sequence through layer 24 of ESM2 to get the embeddings and then passing the embeddings through the SAE encoder. We use the linear probe weights and find the indices that correspond to the five largest positive and negative probe weight indices for which the corresponding index in $\mathbf{z}$ is active as well. We then find the amino acids in the sequence that are being activated by the SAE: given the i^{th} latent, the amino acid activations associated with this latent are $\mathbf{z}[i, :]$. We then use the top five mutants with the highest fitness found from steering the SAE and analyze the activation difference to find the amino acids in the sequence that had the largest absolute activation difference between the wild-type and steered sequence SAE embeddings. We use PyMOL to visualize these changes.

To identify active sites in GFP, we utilize the positions provided in [30] under the Methods section titled “Refinement and mutational scan”. For GB1, we identify allosteric and binding sites based on [25] from Extended Data Fig. 7c. We additionally identify epistatic sites based on [23].

A.7 Weight Sparsity

To quantify sparsity in linear probe weights, we measure the proportion of total variance explained by the top 5% of weights ranked by magnitude. Using linear probe weights from the random extrapolation task with the first seed, we compute, for each training size N, the ratio between the variance captured by the top 5% of weights and the total variance of all weights (where the total number of weights is d_{SAE} for SAE, d_{ESM} for ESM layer, and V for ESM logits) is computed. We plot the magnitude of probe weights for each model in Fig. 4. For visualization purposes, we exclude weights that have a magnitude greater than 3. This occurs 11 times in the ESM logits but not in the ESM layer or SAE.

B Additional Experimental Results

Table 4: Average Spearman ρ across all low-N regimes under mutation extrapolation.

Method	**DMS**	N **= 8** ↑	N **= 24** ↑	N **= 96** ↑	N **= 384** ↑
SAE	GFP_AEQVI_Sarkisyan	0.06 ± 0.09	**0.15 ± 0.10**	**0.29 ± 0.08**	**0.36 ± 0.04**
	SPG1_STRSG_Olson	0.15 ± 0.13	**0.42 ± 0.09**	**0.67 ± 0.05**	**0.79 ± 0.02**
	SPG1_STRSG_Wu	0.14 ± 0.25	**0.31 ± 0.21**	**0.34 ± 0.11**	**0.46 ± 0.13**
	DLG4_HUMAN_Faure	**0.32 ± 0.13**	**0.39 ± 0.07**	**0.50 ± 0.09**	**0.61 ± 0.05**
	GRB2_HUMAN_Faure	**0.33 ± 0.22**	**0.49 ± 0.06**	**0.63 ± 0.04**	**0.67 ± 0.03**
	F7YBW8_MESOW_Ding	**0.54 ± 0.23**	**0.57 ± 0.16**	0.61 ± 0.17	0.63 ± 0.20
ESM layer	GFP_AEQVI_Sarkisyan	0.05 ± 0.09	0.11 ± 0.11	0.23 ± 0.06	0.29 ± 0.07
	SPG1_STRSG_Olson	**0.20 ± 0.19**	0.34 ± 0.07	0.63 ± 0.04	0.78 ± 0.02
	SPG1_STRSG_Wu	**0.17 ± 0.32**	0.30 ± 0.23	0.30 ± 0.09	0.33 ± 0.22
	DLG4_HUMAN_Faure	0.22 ± 0.14	0.33 ± 0.08	0.47 ± 0.07	0.55 ± 0.11
	GRB2_HUMAN_Faure	**0.33 ± 0.22**	0.44 ± 0.08	0.61 ± 0.04	**0.67 ± 0.03**
	F7YBW8_MESOW_Ding	0.51 ± 0.23	0.54 ± 0.21	0.59 ± 0.17	**0.64 ± 0.20**
ESM logits	GFP_AEQVI_Sarkisyan	**0.13 ± 0.10**	0.13 ± 0.13	0.22 ± 0.06	0.30 ± 0.03
	SPG1_STRSG_Olson	0.16 ± 0.12	0.29 ± 0.09	0.42 ± 0.06	0.58 ± 0.03
	SPG1_STRSG_Wu	0.10 ± 0.13	0.07 ± 0.24	0.08 ± 0.23	0.26 ± 0.28
	DLG4_HUMAN_Faure	0.13 ± 0.11	0.15 ± 0.08	0.22 ± 0.12	0.34 ± 0.10
	GRB2_HUMAN_Faure	0.15 ± 0.17	0.30 ± 0.07	0.50 ± 0.08	0.59 ± 0.03
	F7YBW8_MESOW_Ding	0.37 ± 0.41	0.52 ± 0.32	**0.64 ± 0.19**	**0.64 ± 0.20**

Table 5: Average Spearman ρ across all low-N regimes under position extrapolation.

Method	**DMS**	N **= 8** ↑	N **= 24** ↑	N **= 96** ↑	N **= 384** ↑
SAE	GFP_AEQVI_Sarkisyan	0.10 ± 0.09	0.13 ± 0.11	**0.25 ± 0.09**	0.26 ± 0.15
	SPG1_STRSG_Olson	**0.18 ± 0.15**	0.36 ± 0.28	**0.54 ± 0.10**	**0.65 ± 0.09**
	SPG1_STRSG_Wu	0.11 ± 0.32	0.08 ± 0.26	**0.15 ± 0.28**	**0.25 ± 0.23**
	DLG4_HUMAN_Faure	**0.37 ± 0.20**	**0.52 ± 0.10**	**0.53 ± 0.10**	0.57 ± 0.08
	GRB2_HUMAN_Faure	**0.28 ± 0.20**	0.27 ± 0.24	**0.51 ± 0.08**	0.48 ± 0.11
	F7YBW8_MESOW_Ding	**0.09 ± 0.46**	0.10 ± 0.47	0.06 ± 0.50	0.29 ± 0.33
ESM layer	GFP_AEQVI_Sarkisyan	0.04 ± 0.10	0.09 ± 0.09	0.23 ± 0.06	0.25 ± 0.12
	SPG1_STRSG_Olson	0.18 ± 0.19	**0.42 ± 0.32**	0.46 ± 0.18	0.55 ± 0.15
	SPG1_STRSG_Wu	**0.16 ± 0.29**	**0.12 ± 0.24**	0.14 ± 0.40	0.20 ± 0.30
	DLG4_HUMAN_Faure	0.15 ± 0.38	0.41 ± 0.24	0.41 ± 0.23	**0.58 ± 0.07**
	GRB2_HUMAN_Faure	0.25 ± 0.21	0.24 ± 0.21	0.40 ± 0.21	0.40 ± 0.09
	F7YBW8_MESOW_Ding	0.05 ± 0.54	0.07 ± 0.51	0.25 ± 0.37	0.05 ± 0.38
ESM logits	GFP_AEQVI_Sarkisyan	**0.12 ± 0.05**	**0.13 ± 0.10**	0.21 ± 0.09	**0.26 ± 0.07**
	SPG1_STRSG_Olson	0.17 ± 0.20	0.30 ± 0.19	0.41 ± 0.16	0.50 ± 0.16
	SPG1_STRSG_Wu	0.13 ± 0.26	0.03 ± 0.19	0.14 ± 0.27	0.05 ± 0.16
	DLG4_HUMAN_Faure	0.00 ± 0.14	0.14 ± 0.15	0.26 ± 0.20	0.39 ± 0.10
	GRB2_HUMAN_Faure	0.20 ± 0.16	**0.27 ± 0.14**	0.41 ± 0.10	**0.49 ± 0.08**
	F7YBW8_MESOW_Ding	0.07 ± 0.51	**0.44 ± 0.33**	**0.34 ± 0.38**	**0.39 ± 0.19**

Table 6: Average Spearman ρ across all low-N regimes under regime extrapolation.

Method	**DMS**	N **= 8** ↑	N **= 24** ↑	N **= 96** ↑	N **= 384** ↑
SAE	GFP_AEQVI_Sarkisyan	**0.11 ± 0.10**	**0.23 ± 0.13**	**0.36 ± 0.05**	**0.56 ± 0.05**
	SPG1_STRSG_Olson	0.21 ± 0.14	**0.39 ± 0.06**	**0.69 ± 0.05**	**0.84 ± 0.01**
	SPG1_STRSG_Wu	**0.14 ± 0.16**	0.22 ± 0.11	**0.27 ± 0.10**	**0.32 ± 0.04**
	DLG4_HUMAN_Faure	**0.31 ± 0.20**	**0.39 ± 0.14**	**0.58 ± 0.09**	0.67 ± 0.06
	GRB2_HUMAN_Faure	**0.34 ± 0.11**	0.39 ± 0.15	0.65 ± 0.07	**0.77 ± 0.01**
	F7YBW8_MESOW_Ding	**0.37 ± 0.20**	0.53 ± 0.08	0.60 ± 0.06	0.65 ± 0.03
ESM layer	GFP_AEQVI_Sarkisyan	0.06 ± 0.10	0.17 ± 0.15	0.32 ± 0.06	0.49 ± 0.04
	SPG1_STRSG_Olson	**0.23 ± 0.09**	0.38 ± 0.08	0.64 ± 0.05	0.83 ± 0.01
	SPG1_STRSG_Wu	0.14 ± 0.18	**0.23 ± 0.11**	0.22 ± 0.08	0.29 ± 0.06
	DLG4_HUMAN_Faure	0.25 ± 0.14	0.34 ± 0.17	0.49 ± 0.13	**0.70 ± 0.05**
	GRB2_HUMAN_Faure	0.30 ± 0.09	**0.40 ± 0.14**	**0.65 ± 0.06**	0.76 ± 0.01
	F7YBW8_MESOW_Ding	0.35 ± 0.17	0.52 ± 0.09	0.59 ± 0.05	**0.68 ± 0.02**
ESM logits	GFP_AEQVI_Sarkisyan	0.11 ± 0.27	0.17 ± 0.15	0.24 ± 0.17	0.40 ± 0.04
	SPG1_STRSG_Olson	0.16 ± 0.13	0.25 ± 0.09	0.42 ± 0.03	0.56 ± 0.02
	SPG1_STRSG_Wu	0.04 ± 0.09	0.14 ± 0.07	0.16 ± 0.05	0.23 ± 0.03
	DLG4_HUMAN_Faure	0.16 ± 0.12	0.27 ± 0.07	0.34 ± 0.09	0.36 ± 0.07
	GRB2_HUMAN_Faure	0.05 ± 0.12	0.32 ± 0.06	0.48 ± 0.04	0.59 ± 0.03
	F7YBW8_MESOW_Ding	0.35 ± 0.14	**0.56 ± 0.08**	**0.60 ± 0.04**	0.64 ± 0.02

Table 7: Average Spearman ρ across all low-N regimes under score extrapolation.

Method	**DMS**	N **= 8** ↑	N **= 24** ↑	N **= 96** ↑	N **= 384** ↑
SAE	GFP_AEQVI_Sarkisyan	0.01 ± 0.09	0.01 ± 0.05	0.02 ± 0.04	**0.02 ± 0.03**
	SPG1_STRSG_Olson	0.04 ± 0.13	0.03 ± 0.10	0.12 ± 0.10	0.09 ± 0.04
	SPG1_STRSG_Wu	0.07 ± 0.07	**0.04 ± 0.07**	0.10 ± 0.09	0.18 ± 0.05
	DLG4_HUMAN_Faure	**0.05 ± 0.09**	0.04 ± 0.08	0.01 ± 0.06	0.06 ± 0.05
	GRB2_HUMAN_Faure	0.01 ± 0.06	**0.16 ± 0.11**	**0.20 ± 0.07**	0.27 ± 0.05
	F7YBW8_MESOW_Ding	**0.17 ± 0.24**	0.00 ± 0.31	0.22 ± 0.20	0.27 ± 0.10
ESM layer	GFP_AEQVI_Sarkisyan	**0.01 ± 0.06**	0.01 ± 0.05	0.01 ± 0.04	0.01 ± 0.04
	SPG1_STRSG_Olson	**0.06 ± 0.08**	**0.03 ± 0.08**	**0.12 ± 0.08**	**0.13 ± 0.05**
	SPG1_STRSG_Wu	**0.11 ± 0.11**	0.02 ± 0.08	**0.11 ± 0.08**	**0.23 ± 0.06**
	DLG4_HUMAN_Faure	0.01 ± 0.09	**0.04 ± 0.06**	**0.03 ± 0.07**	**0.07 ± 0.07**
	GRB2_HUMAN_Faure	**0.02 ± 0.07**	0.14 ± 0.11	0.18 ± 0.06	**0.30 ± 0.04**
	F7YBW8_MESOW_Ding	0.14 ± 0.24	0.03 ± 0.31	0.23 ± 0.19	0.37 ± 0.09
ESM logits	GFP_AEQVI_Sarkisyan	0.01 ± 0.08	**0.05 ± 0.08**	**0.06 ± 0.04**	0.00 ± 0.06
	SPG1_STRSG_Olson	0.00 ± 0.11	0.01 ± 0.09	0.09 ± 0.10	0.05 ± 0.06
	SPG1_STRSG_Wu	0.02 ± 0.06	0.03 ± 0.07	0.08 ± 0.05	0.13 ± 0.05
	DLG4_HUMAN_Faure	0.05 ± 0.10	0.01 ± 0.07	0.02 ± 0.04	0.00 ± 0.04
	GRB2_HUMAN_Faure	0.02 ± 0.08	0.04 ± 0.08	0.12 ± 0.05	0.23 ± 0.04
	F7YBW8_MESOW_Ding	0.15 ± 0.30	**0.03 ± 0.22**	**0.30 ± 0.21**	**0.39 ± 0.14**