HotProtein: A Novel Framework for Protein Thermostability Prediction and Editing

Tianlong Chen,* Chengyue Gong*, Daniel Jesus Diaz, Xuxi Chen, Jordan Tyler Wells, Qiang Liu, Zhangyang Wang, Andrew Ellington, Alex Dimakis, Adam Klivans
The University of Texas at Austin
{tianlong.chen,cygong17,danny.diaz,xxchen,jordantwells}@utexas.edu
{lqiang,atlaswang,andy.ellington,dimakis,klivans}@utexas.edu

Abstract

The molecular basis of protein thermal stability is only partially understood and has major significance for drug and vaccine discovery. The lack of datasets and standardized benchmarks considerably limits learning-based discovery methods. We present `HotProtein`, a large-scale protein dataset with *growth temperature* annotations of thermostability, containing 182K amino acid sequences and 3K folded structures from 230 different species with a wide temperature range $-20°\mathtt{C} \sim 120°\mathtt{C}$. Due to functional domain differences and data scarcity within each species, existing methods fail to generalize well on our dataset. We address this problem through a novel learning framework, consisting of (1) Protein structure-aware pre-training (SAP) which leverages 3D information to enhance sequence-based pre-training; (2) Factorized sparse tuning (FST) that utilizes low-rank and sparse priors as an implicit regularization, together with feature augmentations. Extensive empirical studies demonstrate that our framework improves thermostability prediction compared to other deep learning models. Finally, we propose a novel editing algorithm to efficiently generate positive amino acid mutations that improve thermostability.

1 Introduction

Proteins are the bio-polymers responsible for executing most biological phenomena and, through evolution, have had their sequences optimized to carry out specific functions within specific cellular environments. A protein's stability is a multidimensional property that depends on a series of factors [2, 3] such as pH, salinity, and temperature (thermostability shown in Figure 1), making it hard to adapt a protein to function outside of its endogenous cellular environment. Protein engineering is the field where natural proteins are mutated to improve their stability in exogenous environments and their overall fitness for a particular function. In protein engineering, one of the initial goals for most engineering campaigns is to improve the thermal stability of protein [4, 5, 6]. Thermally stabilized proteins are more robust and therefore enable downstream applications in the food [7], biofuel [8], detergent [9], chemical [10], and pharmaceutical industry [11], drug design [12, 13], and bioremediation of environmental pollutants [14, 15]. Thus, to accelerate the engineering of a target protein it is critical to understand and accurately predict thermal stability changes of mutations. There has been a substantial effort from the community to

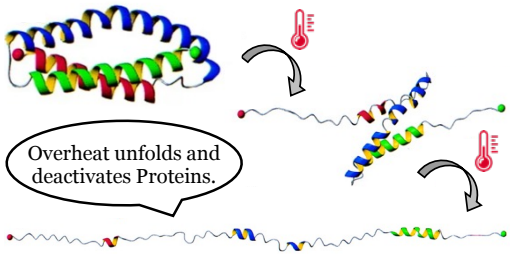

Figure 1: Overheat deactivates proteins [1].

*Equal Contribution.

NeurIPS 2022 AI for Science Workshop.

quantitatively understand and model protein thermostability [*e.g.,* 2, 3, 16, 17, 18, 19]. However, the generalizability of them is still unsatisfactory, and laborious experimental methods such as directed evolution are often preferred.

To enhance the capabilities of learning-based approaches, we present a large-scale, standardized protein benchmark, *i.e.,* `HotProtein`, with *organism-level* temperature annotations which is a lower bound of protein's melting temperature [20]. It consists of 182K protein sequences and 3K folded structures from 230 different species, covering a broad temperature range of $-20°$C-$120°$C. However, similar to [3], naively trained deep models even on our dataset do not enable generalization to unseen proteins. The presumed reasons are (1) the considerable functional heterogeneity in proteins that arise from the environmental conditions and evolutionary history and (2) the scarcity of high-quality thermostability experimental data due to the massive cost and labor required to generate such data.

To tackle these pain points, we propose a novel algorithmic pipeline to improve thermostability prediction. First, we enrich our sequence embeddings by infusing 3D structural information in a contrastive manner–we call this structure-aware pre-training (SAP). Then, we further fine-tune our model with a factorized sparse tuning (FST) approach. Here, we utilize factorized low-rank and sparse priors as implicit regularizers and leverage feature augmentation, such as mix-up [21] and worse-case augmentations [22]. FST greatly boosts the performance of tuned predictors, suggesting improved data efficiency and robustness against domain shifts [23, 24]. Extensive evaluations on both `HotProtein` and the other existing protein datasets verify our proposals' effectiveness.

Finally, to identify the top mutational predictions likely to improve thermal stability for a target protein, we develop a new optimization-based editing framework on top of a classifier or regressor, that attempts to mimic the process of directed evolution while limiting the stochasticity [16]. Unlike existing protein engineering approaches [25, 26, 27] that directly utilize the predictions to generate mutational designs, our proposal maximizes the model's objective to approach a more thermostable label to identify input mutated sequences. Our contributions can be summarized as follows:

- ⋆ We collect and present a large-scale protein dataset, *i.e.,* `HotProtein`, with organism-level temperature annotations. We use the organism's environmental growth temperature to label and classify all proteins within each organism, which we use for thermostability prediction and editing. It contains 182K amino acid sequences and 3K folded 3D structures of proteins from 230 different species, covering five thermostability types, *e.g.,*, Cryophilic, Psychrophilic, Mesophilic, Thermophilic, and Hyperthermophilic.
- ⋆ We introduce a protein structure-aware pre-training by injecting 3D structural information into sequence embeddings in a contrastive fashion. It enhances the diversity and expressivity of the protein representations, resulting in improved thermostability predicting performance.
- ⋆ We propose a robust and data-efficient tuning framework that performs weight updates in the factorized and sparse subspace together with augmented feature embedding. This leads to substantial performance improvements against data scarcity and severe distribution shifts.
- ⋆ We formulate the search for thermal stabilizing mutations as an optimization problem: for a target protein and a trained predictor, we built an editing framework that optimizes the input protein sequences to identify thermostabilizing mutations.
- ⋆ Extensive experiments conducted on both thermostability prediction and protein editing tasks, consistently demonstrate the superiority of our proposals over various existing approaches [28]. For example, when fine-tuned on experimentally determined T_m dataset, FireProtDB, our editing suggester achieves 53.93% ($\uparrow 8.96\%$) precision in positive mutation classification, 50.79 ($\uparrow 6.54$) Spearman ρ correlation coefficient in the temperature regression, and 54.24% ($\uparrow 1.83\%$) successful rate in generating positive single mutations.

2 The `HotProtein` Dataset

To obtain thermostable labels for an organism's proteome, we collect the raw data from the NCBI BioProject [31], which offers an organizational framework to access the (meta-)data about research projects, which is deposited or planned for deposition, into archival repositories.

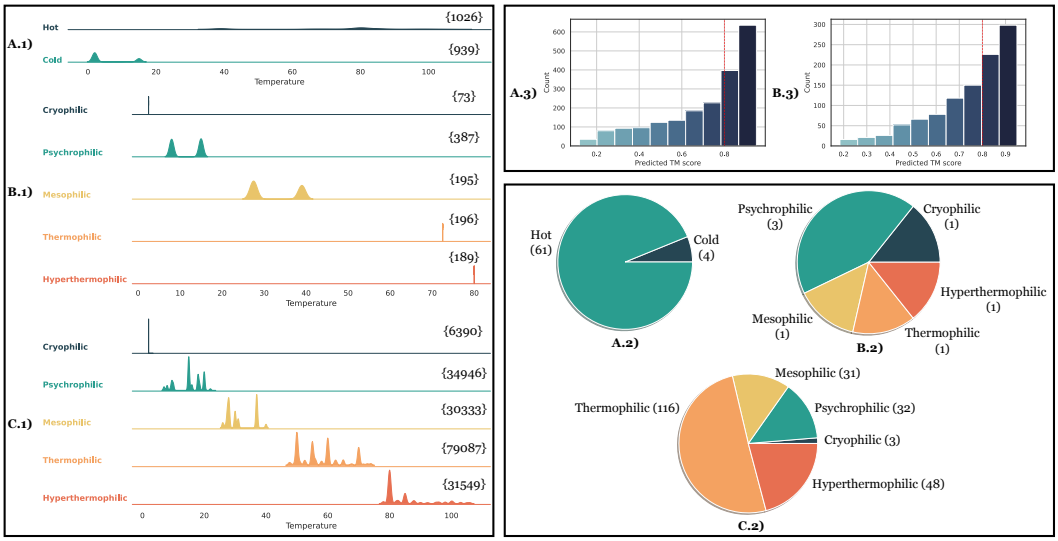

Figure 2: The overview of `HotProtein` dataset. Figures A.1 $\sim$ 3), B.1 $\sim$ 3), and C.1 $\sim$ 2) collect the statistics of `HP-S`^{2}`C2`, `HP-S`^{2}`C5`, and `HP-S` respectively. The *left* figure records the density distribution of each category over the protein's **growth (organism-level) temperature**. The density in a bin is computed as $\frac{\text{\# proteins within the bin}}{\text{\# proteins in the category}}$. $\{1026\}$ indicates that there are 1026 proteins in the corresponding class "Hot". The *upper right* figure is the predicted template modeling (P-TM) score from AlphaFoldv2 [29, 30], reflecting the quality of folded protein structures. A larger P-TM suggests a better quality and P-TM ≥ 0.8 (red lines) is a normal threshold for satisfied folded structures. The *bottom right* figure presents the species distribution of the three datasets, and # species in certain categories is included in the brackets.

Preprocess. From the NCBI bioproject XML file[2], we filter organisms where the environmental data (*i.e.,* "OptimumTemperature" and "TemperatureRange") is available. After removing duplicate organism entries (keeping the first entry), this provides us with $1,733$ unique entries. Next, we proceed to download these organisms' proteomes from UniProt[3] via their taxids and bin the proteomes based on the "TemperatureRange" classification of that organism. Finally, we remove all proteins over $1,500$ amino acids in length and utilize CD-Hit[4] to cluster protein sequences across organisms within a "TemperatureRange" class at a sequence similarity threshold of 50%. In each cluster, we only keep proteins whose sequence lengths are between $200 \sim 550$.

Annotation and Folding. To further increase the fidelity of our annotations, we remove organisms from each "TemperatureRange" bin where the "OptimumTemperature" does not fall within the the corresponding limits: ❶ *Hyperthermophilic* ($>$ 75 Celsius), ❷ *Thermophilic* ($45 \sim 75$ Celsius), ❸ *Mesophilic* ($25 \sim 45$ Celsius), ❹ *Psychrophilic* ($5 \sim 25$ Celsius), and ❺ *Cryophilic* ($-20 \sim 5$ Celsius). The filtered proteomes and their corresponding annotations form the `HotProtein` dataset and are utilized throughout the classification and regression tasks in our study.

For a random subset of the `HotProtein` dataset ($\sim$ 3K), we predict structure with AlphaFoldV2 [29, 30] to obtain their 3D coordinates. The official implementation is adopted. We report the predicted template modeling (P-TM) scores for each folded structure in Figure 2 A.3) and B.3). The P-TM score is a rough approximation of the folding quality [29], and ≥ 0.8 indicates a satisfying protein structure in common sense.

Properties. As described in Figure 2, we generate four distinct testbeds from the `HotProtein` dataset that differed in scale: (1) `HP-S`^{2}`C2` has 1026 "hot" ($\geq 45°$`C`) and 939 "cold" ($< 45°$`C`) proteins from 61 and 4 species, respectively. Both sequence and structure statistics of these proteins are provided. (2) `HP-S`^{2}`C5` consists of both sequences and structures for $\{73, 387, 195, 196, 189\}$ proteins sampled from the five categories, from *Cryophilic* to *Hyperthermophilic*. (3) `HP-S` is the entire sequence `HotProtein` dataset. It contains $\{6390, 34946, 30333, 79087, 31549\}$ sequences from $\{3, 32, 31, 116, 48\}$ different species, of five classes ordered from *Cryophilic* to *Hyperthermophilic*. (4) Moreover, `HP-SC2` as a 2-class variant is created by merging *Hyperthermophilic* and *Thermophilic*

[2] https://ftp.ncbi.nlm.nih.gov/bioproject/bioproject.xml
[3] https://www.uniprot.org/help/uniprotkb
[4] http://weizhong-lab.ucsd.edu/cd-hit/

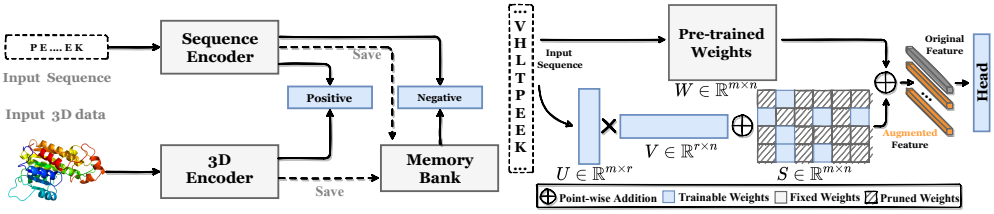

Figure 3: The overall pipeline of our proposals. The *left* describes the protein structure-aware pre-training (SAP); The *right* presents the factorized sparse tuning (FST).

as "hot" class and the other three as "cold" class. **Note that all temperature annotations are organism-level, serving as a lower bound of for a protein's melting temperature**.

3 Methodology

Denote the thermostability dataset by $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^{N}$, where $\boldsymbol{x}_i$ stands for the input while y_i is the thermostability critical temperature (either real value or class index). Here we describe 1) how we pretrain the model $\mathcal{F}$, and 2) how we finetune to pretrained model to fit $\mathcal{D}$.

3.1 Protein Structure-Aware Pre-training

Previous protein pre-trained models [*e.g.,* 32, 33, 34, 35] mainly focus on masking prediction tasks and amino acid representations. These models achieve considerable improvements on amino acid prediction tasks, *e.g.,* contact prediction, mask prediction [36], mutational effect prediction [35]. On the other hand, directly learning universal protein embedding is potentially useful for protein prediction tasks. Inspired by recent sentence representation learning works [37] and 3D structure-aware pre-training model [38], we propose a contrastive loss for representation learning.

Contrastive Loss Design. Denote the model as $\mathcal{F}$, our contrastive loss function is defined as:

$$\mathcal{L}_{\text{InfoNCE}}(\boldsymbol{x}_i, \text{Neg}(\boldsymbol{x}_i)) = -\log \frac{\exp(\mathcal{F}'(\boldsymbol{x}_i)\cdot\mathcal{F}''(\boldsymbol{x}_i)/\tau)}{\exp(\mathcal{F}'(\boldsymbol{x}_i)\cdot\mathcal{F}''(\boldsymbol{x}_i)/\tau) + Z(\boldsymbol{x}_i, \text{Neg}(\boldsymbol{x}_i))}, \tag{1}$$

where $Z(\boldsymbol{x}_i, \text{Neg}(\boldsymbol{x}_i)) = \sum_{\boldsymbol{x}_j\in\text{Neg}(\boldsymbol{x}_i)} \exp(\mathcal{F}'(\boldsymbol{x}_i)\cdot\mathcal{F}'^{\text{mom}}(\boldsymbol{x}_j)/\tau)$. Here, $\mathcal{F}'(\boldsymbol{x}_i)$ and $\mathcal{F}''(\boldsymbol{x}_i)$ are two copies of randomly perturbed models by injecting independent dropout noises into $\mathcal{F}(\boldsymbol{x}_i)$, and $\text{Neg}(\cdot)$ presents a set of negative examples, and $\mathcal{F}^{\text{mom}}$ is a slowly updated momentum model (see below) and τ is the temperature value. Notice that random mutating amino acid could introduce unknown changes [39, 40], we add perturbations to hidden representations and inject layer-wise dropout noise [41] with a rate 0.05 in practice, and follow the hyperparameter configurations in [42].

Leverage 3D Information. We combine an additional 3D structure model using 3D inputs with the sequence-based model, to enhance the performance. We therefore slightly change Equation 1 into,

$$\mathcal{L}_{\text{3D}}(\boldsymbol{x}_i, \text{Neg}(\boldsymbol{x}_i)) = -\log \frac{\exp(\mathcal{F}'(\boldsymbol{x}_i)\cdot\mathcal{F}''_{\text{3D}}(\boldsymbol{x}_i)/\tau)}{\exp(\mathcal{F}'(\boldsymbol{x}_i)\cdot\mathcal{F}''_{\text{3D}}(\boldsymbol{x}_i)/\tau) + Z(\boldsymbol{x}_i, \text{Neg}(\boldsymbol{x}_i))}, \tag{2}$$

$$Z(\boldsymbol{x}_i, \text{Neg}(\boldsymbol{x}_i)) = \sum_{\boldsymbol{x}_j\in\text{Neg}(\boldsymbol{x}_i)} \left\{ \exp(\mathcal{F}'(\boldsymbol{x}_i)\cdot\mathcal{F}'^{\text{mom}}_{\text{3D}}(\boldsymbol{x}_j)/\tau) + \exp(\mathcal{F}'_{\text{3D}}(\boldsymbol{x}_i)\cdot\mathcal{F}'^{\text{mom}}(\boldsymbol{x}_j)/\tau) \right\},$$

where $\mathcal{F}_{\text{3D}}$ stands for the 3D structure model which uses 3D coordinates as inputs, $\mathcal{F}^{\text{mom}}_{\text{3D}}$ is its momentum model. The memory bank logs the negative examples from both the sequence and 3D models. In short, a pair for sequence and 3D model representation is regarded as the positive pair, and we aim at injecting 3D representation information into the sequence model.

3.2 Factorized Sparse Tuning Under Data Scarcity and Distribution Shift

In real-world application domains, it is extremely challenging or even infeasible to collect a sufficient large-scale (engineered) protein dataset, due to the massive time and resource cost of transformation, protein expression, and purification. Therefore, data scarcity is one of the crucial bottlenecks in predicting protein properties. Another obstacle lies in the substantial data distribution shifts. For examples, normal functional protein regions are formed by around 100 amino acids [43], while COVID-19 related proteins usually contain more than $1,000$ amino acids [44]. Meantime, proteins

with different (sometimes even the same [44]) functionalities have distinctive and idiographic structures, sharing few common characteristics. To tackle these two issues, we propose a factorized sparse tuning pipeline (FST). It leverages the low-rank and sparse priors as implicit regularizations for enhanced data-efficiency [*e.g.,* 45, 46, 47, 48, 49, 50] and robustness to domain shifts [23].

Enforcing the Low-rank Prior through Factorization. Given a pre-trained model $W_p \in \mathbb{R}^{m\times n}$, we perform a low-rank decomposition to its weight update ΔW represented as $W_p + \Delta W = W_p + UV$, where $U \in \mathbb{R}^{m\times r}$, $V \in \mathbb{R}^{r\times n}$, and the rank $r \ll \min\{d, k\}$. Usually, $r = 4$ or 8 (*i.e.,* 0.6% of total parameters) is sufficient to achieve a great performance in our case. During forward, the input is fed into both dense pre-training W_p and its low-rank representation UV. The obtained features are summed in a coordinate-wise manner, as demonstrated in Figure 3 (*right*). Take the original feature $h = W_p\boldsymbol{x}$ as an example. Our modified features can be described as below:

$$\hat{h} = W_p\boldsymbol{x} + \Delta W\boldsymbol{x} = W_p\boldsymbol{x} + UV\boldsymbol{x}. \tag{3}$$

As for the backpropagation, W_p is frozen and low-rank matrices $\{U, V\}$ receive gradient updates. We adopt Xavier normal [51] and zero initialization for U and V respectively, and therefore the low-rank update is zero in the beginning. As suggested by [52], $UV\boldsymbol{x}$ can be scaled via an extra hyperparameter α, which works similarly to the learning rate. In our case, we find α is not sensitive and set it as 4, *i.e.,* the default value used in [52].

Enforcing the Sparsity Prior. We introduce the sparsity into tuning processes as an implicit structural prior by modeling a sparse weight update with $S \in \mathbb{R}^{m\times n}$. As indicated in Figure 3, our refined features $\tilde{h}$ are depicted as follows:

$$\tilde{h} = W_p\boldsymbol{x} + (UV + S)\boldsymbol{x}, \quad S = \begin{cases} s_{i,j}, & (i,j) \in \Omega \\ 0, & (i,j) \in \Omega^{\mathrm{C}} \end{cases}, \tag{4}$$

where a "residual" feature $S\boldsymbol{x}$ is point-wisely added to $\hat{h}$ from Equation 3, $i \in \{1, 2, \cdots, m\}$, and $j \in \{1, 2, \cdots, n\}$. The set Ω determines the position of trainable $s_{i,j}$ and pruned elements, where the latter is 0 across the whole training. We compute the initial sparse matrix of S by $i)$ first solving a robust principal component decomposition [53] of W_p with an efficient algorithm (*i.e.,* GreBsmo [54]), and $ii)$ then eliminating the elements with the least magnitude of obtained sparse solutions. In this way, the step $i)$ produces the initial values of $s_{i,j}$ and the step $ii)$ constructs the set Ω, where we observe $|\Omega| = 64$ (*i.e.,* 0.01% of total parameters) is good enough in our case. As also revealed in [55], combining the low-rank and sparse weight updates is capable of delivering superior performance, compared to either of them.

Feature Augmentation. Another group of common fixes to data scarcity and domain shift problems is data augmentation [56]. However, most of existing data-level augmentations are unrealistic to protein sequences, including regional dropout [54, 57] and CutMix [58] in computer vision; synonym replacement [59, 60, 61], random insertion, swap, and deletion [62] in natural language processing. The reason is that even a single amino acid mutation for the protein sequence may dramatically change its functionality [40, 39]. To avoid ambiguity, we utilize feature-level augmentations which manipulate the model's intermediate feature embedding. Specifically, we examine two effective mechanisms: (1) *Mixup* feature augmentation. It creates a fused feature $\tilde{h}_{\text{aug}} = \lambda \times \tilde{h}_1 + (1-\lambda) \times \tilde{h}_2$ and its associated soft label $y_{\text{aug}} = \lambda \times y_1 + (1-\lambda) \times y_2$, where $\{\tilde{h}_1, y_1\}$, $\{\tilde{h}_2, y_2\}$ are {feature embedding, label} of two different sequences and $\lambda = 0.2$ in our experiments. (2) *Worst-case* feature augmentation. It injects worst-case noises δ and builds an augmented feature $\tilde{h}_{\text{aug}} = \tilde{h} + \delta$, where δ is generated by $\max_\delta \mathcal{L}(\mathcal{F}(\boldsymbol{x}, \tilde{h} + \delta), y)$ with the gradient ascent [22]. $\mathcal{L}$, $\mathcal{F}$ and $\boldsymbol{x}$ denote the objective function, model, and input sequence, respectively.

4 Experiments

4.1 Implementation Details

Metrics. The performance is evaluated on the test splits. {Accuracy, Precision} and {Spearman, Pearson} correlation coefficients are used for classification and regression tasks. For `HP-S`^{2}`C2` and `HP-S`^{2}`C5`, 10-fold evaluation is conducted; while on `HP-S` and `HP-SC2`, we run three replicates with different random seeds. Average performance and its 95% confidence intervals are reported. **Training details** are provided in Appendix C.

Table 1: Performance of predicting thermostability with classification. Accuracy (%) is reported for all three datasets, and Precision (%) is calculated for the 2-class classification on HP-S^{2}C2 and HP-SC2. "FST", "Aug.", and "SAP" denote factorized sparse tuning, feature augmentation, and protein structure-aware pre-training, respectively. FST adopts $(r, |\Omega|) = (4, 64)$ on HP-S^{2}C2/C5 and $(r, |\Omega|) = (8, 64)$ on HP-S/SC2. N.A. means "not applicable". 95% confidence interval are computed via the 10-fold evaluation on HP-S^{2}C2/C5 and 3 replicates on HP-S/SC2.

Methods	HP-S^{2}C2		HP-S^{2}C5	HP-S	HP-SC2	
	Accuracy	Precision	Accuracy	Accuracy	Accuracy	Precision
3D GCN [63]	78.88±1.57	73.39±2.76	67.40±2.11	N.A.	N.A.	N.A.
TAPE [64]	83.31±1.10	76.42±3.06	66.44±2.30	64.75±0.23	76.37±0.25	80.64±0.50
ESM-IF1 [38]	79.08±0.85	76.49±3.96	58.75±2.46	N.A.	N.A.	N.A.
ESM-1B [28]	91.19±0.47	84.18±1.71	83.26±1.54	69.50±0.16	86.24±0.22	88.14±1.62
ESM-1B + Aug.	91.74±0.79	86.09±2.12	84.32±1.41	69.54±0.39	86.26±0.22	88.27±1.43
ESM-1B + FST	91.85±0.45	84.85±1.04	85.96±1.13	72.97±0.28	87.54±0.38	88.71±1.73
ESM-1B + FST + Aug.	91.91±0.64	86.10±1.14	86.08±1.33	73.09±0.10	87.50±0.12	88.83±1.27
ESM-1B + FST + Aug. + SAP	**92.36±0.58**	**86.51±1.67**	**86.25±1.03**	**73.21±0.13**	**87.57±0.10**	**89.07±1.29**

Table 2: Performance of thermostability regression. Correlation coefficients are reported for all three datasets. 95% confidence interval are computed via the 10-fold evaluation on HP-S^{2}C2/C5 and 3 replicates on HP-S.

Methods	HP-S^{2}C2		HP-S^{2}C5		HP-S	
	Spearman	Pearson	Spearman	Pearson	Spearman	Pearson
3D GCN [63]	0.490±0.019	0.469±0.019	0.291±0.053	0.301±0.074	N.A.	N.A.
TAPE [64]	0.432±0.061	0.386±0.065	0.367±0.063	0.364±0.047	0.504±0.013	0.453±0.031
ESM-IF1 [38]	0.589±0.040	0.547±0.036	0.373±0.036	0.377±0.035	N.A.	N.A.
ESM-1B [28]	0.890±0.018	0.893±0.0238	0.712±0.043	0.804±0.023	0.807±0.001	0.809±0.001
ESM-1B + Aug.	0.895±0.014	0.909±0.010	0.714±0.034	0.811±0.034	0.808±0.002	0.809±0.001
ESM-1B + FST	0.898±0.008	0.900±0.009	0.742±0.039	0.815±0.024	0.819±0.002	0.825±0.004
ESM-1B + FST + Aug.	0.892±0.011	0.912±0.013	0.747±0.026	0.818±0.025	0.820±0.001	0.825±0.003
ESM-1B + FST + Aug. + SAP	**0.906±0.010**	**0.923±0.012**	**0.754±0.035**	**0.837±0.019**	**0.823±0.001**	**0.827±0.003**

4.2 Predicting Thermostability via Classification and Regression

Comparison to Existing Approaches. We evaluate our proposals and compare with existing approaches on HP-S^{2}C2/C5 and HP-S/SC2 for both classification and regression tasks. 3D GCN [63] and TAPE [64] are classical structure- and sequence-based models, while ESM-IF1 [38] and ESM-1B [28] emerges recently as current state-of-the-art approaches, dealing with structure and sequence inputs respectively. From results shown in Table 1 and 2, several consistent observations can be drawn: ❶ *Compared with baselines.* Our proposal, *i.e.,* ESM-1B+FST+Aug.+SAP, greatly surpasses various baseline by a margin of $\{1.17\% \sim 13.48\%$ accuracy, $2.33\% \sim 13.12\%$ precision, $0.016 \sim 0.474$ Spearman, and $0.030 \sim 0.537$ Pearson correlation$\}$ on HP-S^{2}C2, $\{2.99\% \sim 27.50\%$ accuracy, $0.042 \sim 0.463$ Spearman, and $0.033 \sim 0.536$ Pearson correlation$\}$ on HP-S^{2}C5, $\{3.71\% \sim 8.46\%$ accuracy, $0.016 \sim 0.319$ Spearman, and $0.018 \sim 0.374$ Pearson correlation$\}$ on HP-S, and $\{1.33\% \sim 11.2\%$ accuracy and $0.93\% \sim 8.43\%$ precision on HP-SC2. These generalization improvements on hold-out proteins validate the effectiveness of our methods in tackling the functional domain shift and data-scarcity issues, *i.e.,* pre-training on a general-purpose dataset UniRef50 (22M samples) and tuning on the specific-domain dataset HotProtein (182K samples). Moreover, our methods perform much stably in general, evidenced by the reduced confidence interval of multiple runs. ❷ *Structure versus sequence-based models.* On the classification task, sequence-based models like ESM-1B and TAPE show clear performance advantages in most cases. As for the regression task, ESM-1B achieves an overwhelming superiority among the four baseline approaches, while the next best model is ESM-IF1 which takes protein structures as inputs. It suggests that powerful mechanisms for utilizing 3D structure information are still missing on our challenging HotProtein dataset. We make a pioneer attempt by leveraging 3D information to enhance sequence-based models. ❸ *Does SAP, FST, and Aug helps?* We examine these three components in an incremental manner and we observe: i) the performance gains from SAP demonstrate the benefits of treating 3D protein structures as auxiliary information; ii) Both Aug. & FST strengthen the model tuning and a combination of them enjoys extra improvements; iii) Among these ingredients of our proposal, FST contributes the most to superior performance. Specifically, Aug consistently obtains improvements in terms of the average performance; FST usually leads to statistically significant improvements with respect to the 95% confidence interval, especially on the HP-S/SC2 dataset. ❹ *Small v.s. large datasets.* Feature level augmentations are more beneficial for small-scale datasets, while FST and SAP bring performance gains for both small and large datasets.

Table 3: Ablation study on proposed components. Accuracy (%) for classification and Spearman & Pearson correlation coefficients for regression are reported. 95% confidence interval are computed via the 10-fold evaluation on HP-S^{2}C5 and 3 replicates on HP-S.

Methods		HP-S^{2}C5			HP-S		
		Accuracy	Spearman	Pearson	Accuracy	Spearman	Pearson
Based on ESM-1B [28]		83.26±1.54	0.712±0.043	0.804±0.022	69.50±0.16	0.807±0.001	0.809±0.001
Feature Aug.	Random	82.97±1.46	0.708±0.043	**0.814±0.023**	69.46±0.28	0.807±0.001	**0.809±0.001**
	Mixup	83.94±1.43	**0.720±0.042**	0.810±0.027	69.27±0.27	0.805±0.001	0.806±0.003
	Worst-case	**84.32±1.41**	0.714±0.034	0.811±0.034	**69.54±0.39**	**0.808±0.002**	**0.809±0.001**
# Rank in FST	$r = 4$	**85.96±1.13**	**0.742±0.039**	**0.815±0.024**	72.75±0.17	0.818±0.001	0.821±0.002
	$r = 8$	85.00±2.16	0.703±0.032	0.801±0.017	72.97±0.28	0.819±0.002	**0.825±0.004**
	$r = 16$	83.75±1.92	0.725±0.040	0.788±0.019	**73.20±0.40**	**0.821±0.001**	**0.825±0.002**
Sparsity in FST	$\|\Omega\| = 16$	85.19±1.02	0.702±0.034	0.794±0.027	72.82±0.43	0.815±0.004	0.822±0.005
	$\|\Omega\| = 32$	85.57±1.16	0.706±0.035	0.767±0.060	72.85±0.49	0.818±0.002	0.820±0.002
	$\|\Omega\| = 64$	**85.96±1.13**	**0.742±0.039**	**0.815±0.024**	**72.97±0.28**	**0.819±0.002**	**0.825±0.004**
	$\|\Omega\| = 128$	85.79±1.08	0.729±0.042	0.807±0.029	72.75±0.49	0.818±0.004	0.822±0.006
3D infor. in SAP	w.o.	83.89±1.10	0.718±0.023	0.815±0.019	69.80±0.48	0.808±0.001	0.811±0.002
	w.	**85.58±1.42**	**0.727±0.035**	**0.817±0.022**	**71.52±2.29**	**0.815±0.001**	**0.819±0.003**

Ablation Study. A comprehensive ablation is presented in Table 3, where we inspect different feature augmentations, the rank number in FST, the sparsity in FST, and the necessity of 3D infor. in SAP, on top of the vanilla ESM-1B model. ❶ *Diverse feature augmentations.* Besides mixup and worst-case feature augmentations, a straightforward baseline ("Random") that applies a Gaussian noise from $\mathcal{N}(0, 0.1^2)$ to features, is also implemented. We see: random feature augmentation usually incurs a performance degradation; mixup benefits models tuned on the small-scale dataset (HP-S^{2}C5), while hurts on the large-scale dataset (HP-S); worst-case feature augmentations consistently improve the generalization of tuned models on both HP-S^{2}C5 and HP-S, which is adopted by default in all other experiments of Table 1, 2, and 4. ❷ *The # rank in FST.* The larger dataset prefers FST with a higher rank such as $r = 8$ or 16, compared to the smaller dataset in which FST with $r = 4$ works the best. The finding is coincided with the ones in [52, 65]. ❸ *The sparsity in FST.* FST with $|\Omega| = 64$ is a "sweet point". Superfluous tuning elements in Ω may lead to inferior results. ❹ *3D infor. in SAP.* Directly introducing a contrastive loss in Equation 1 to the pre-training, has already boosted ESM-1B's performance, implying an improved protein embedding learning. Coupling the 3D information of protein structures, obtains an additional quality bonus for the pre-training.

More ablation studies. We summarize the results of additional ablation studies here and refer readers to the Appendix D for details: ❶ We achieve the best results on extra test benchmarks and class-balanced test sets. ❷ SAP outperforms other approaches to inject structure information. ❷ Training from scratch yields worse results than pretraining. ❸ On ESM-1B, we test several other baselines. We notice that additional ℓ_2 or ℓ_1 regularization, end-to-end fine-tuning, partially frozen tuning, and other feature aggregation methods all come to worse results than our current ESM-1B.

4.3 Protein Editing Towards Improved Thermostability

Based on the models we trained in previous sections, three approaches for protein editing are proposed. We first describe the setting we use to edit protein, and then report the performance of different models for three settings, classification, regression and Editing. To demonstrate the effectiveness of protein editing towards improved thermostability, we further evaluate our proposals on FireProtDB[5] which is a manually curated database of the protein stability data for single-point mutants. It contains 0.2K natural protein amino acid sequences, *i.e.,* $\mathcal{P} = \{p^{(i)}\}$, $i \in \{1, \cdots, x\}$, and their 3.9K mutated sequences, *i.e.,* $\hat{\mathcal{P}} = \{\hat{p}_j^{(i)}\}$, $i \in \{1, \cdots, x\}$ and $j \in \{1, \cdots, n^{(i)}\}$, where $n^{(i)}$ is the number of mutated variants for the original protein sequence $p^{(i)}$. We take $\mathcal{P}$ as the training set and $\hat{\mathcal{P}}$ as the hold-out testing set. Detailed configurations are referred to the Appendix C.

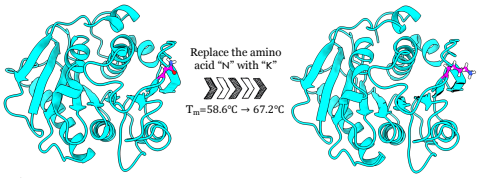

Figure 4: Visualization of an AI design that thermostabilized a protein. The amino acid "N" was predicted by the MutCompute web server to be mutated to a "K", which was experimentally shown to improve thermostability by 8.6°C and was critical for the engineering of FAST-PETase[14]. ThermoPETase structure (left) (PDB: 6ij6) and the mutated one (right) was obtained via [66].

[5] https://loschmidt.chemi.muni.cz/fireprotdb/

Table 4: Evaluation of protein editing suggestions. Accuracy (%) and Precision (%) for classifiers, Kendall τ & Spearman ρ rank correlation coefficients for regressors, and successful rate (%) for adversarially learned mutations of protein sequences are reported. 95% confidence interval is computed via three trials.

FireProtDB		Classifier		Regressors		Learned Mutations
		Accuracy	Precision	Kendall τ	Spearman ρ	Successful Rate
Zero shot	TAPE	44.75	30.18	0.07	5.64	34.78
	ESM-1B	62.26	41.73	8.34	9.49	40.39
	Ours	66.18	43.60	13.65	19.96	43.46
Fine-tune on $\mathcal{P}$	TAPE	55.18$\pm$0.38	38.85$\pm$0.36	32.33$\pm$1.35	43.78$\pm$1.87	44.59$\pm$0.45
	ESM-1B	69.30$\pm$0.53	44.97$\pm$0.47	34.76$\pm$1.42	44.25$\pm$1.89	52.41$\pm$0.38
	Ours	**71.42$\pm$0.34**	**53.93$\pm$0.39**	**36.97$\pm$0.78**	**50.79$\pm$1.22**	**54.24$\pm$0.41**

Editing Suggestions via Classifiers. We deliver the protein editing suggestion via proposed classifiers. The five-class classification is performed in both fine-tuning and testing stages with $\mathcal{P}$ and $\hat{\mathcal{P}}$, respectively. Furthermore, we regard a mutation as *positive* if $\hat{p}_j^{(i)}$ is predicted to a class that has higher temperatures than the category of its original counterpart $p^{(i)}$ (*e.g.,* from *Psychrophilic* to *Mesophilic*); otherwise, we label the mutation as *negative*. Then, we compute the associated accuracy and precision to evaluate the quality of editing suggestions from our proposals.

Editing Suggestions via Regressors. We can also suggest possible editing via regressors. Specifically, we fine-tune models to regress the protein's temperature in $\mathcal{P}$, predict the possible temperature of their mutated variants in $\hat{\mathcal{P}}$, and measure the ordinal association between prediction and ground truth temperatures of $\hat{\mathcal{P}}$.

Optimizing Editing Suggestions. Instead of classifying whether the mutation increases or decreases the thermostability, we propose an efficient editing algorithm, by optimizing towards an improved thermostability: $\max_{|\mathcal{M}|\leq n} \mathcal{L}(\mathcal{F}(\mathcal{M}(\boldsymbol{x})), y_t)$, where $\boldsymbol{x}$ is the input, y_t denotes the target label of the class with a higher temperature (*e.g.,* from *Mesophilic* to *Thermophilic*), $\mathcal{M}$ stands for the mutation function whose number of mutated amino acids $|\mathcal{M}|$ is constrained by n, $\mathcal{F}$ and $\mathcal{L}$ is our classifier and the objective function. In practice, we set y_t as the highest-temperature class, *Hyperthermophilic*. We set $\mathcal{L}$ as probability change times the saliency score following PWWS [67]. More are in Sec. C.1.

Experimental Results. As demonstrated in Table 4, ❶ ESM-1B outperforms TAPE when it is zero-shot transferred to test data. Ours performs the best. We achieve 66.13% accuracy, 13.65% Kendall correlation, and 19.96% Spearman correlation, which is 3.86%, 5.31%, 10.47% relative higher than the ESM-1B baseline. ❷ After finetuning on $\mathcal{P}$, ESM-1B obtains slightly better results than TAPE on both regression and classification metrics. Our model still outperforms both in all the tested cases. ❸ When we do protein editing after fine-tuning, ours achieves the best (54.24% successful rate, 3.49% relative improvements than ESM-1B). It indicates that a better classifier also benefits protein engineering. Visualization of an edited protein is displayed in Figure 4.

5 Conclusion

Predicting thermostabilizing mutations is a primary goal of most protein engineering campaigns. However, the lack of thermally annotated protein data and effective algorithms has hindered the development of a thermostability prediction model that can generalize across protein-space. This work provides attempts to address both points via a large-scale protein dataset (`HotProtein`) with *species-specific, lower-bound thermostability annotations* and a novel algorithmic framework designed to tackle the intrinsic challenges of functional domain shifts and data scarcity in thermostability protein engineering. Extensive results validate that our dataset and algorithmic framework provide meaningful improvements over baseline models. Lastly, based on established superior predictors, we search a protein's single-point mutation landscape towards identifying thermostabilizing variants. We will keep updating `HotProtein` by collecting more sequences, improving our sequence-clustering and filtering, and folding more structures.

Related Works. Due to space limitations, we place discussions about protein engineering background, directed evolution, guided directed evolution, other protein thermostability datasets, sequence-/structure-based protein models, and sparse & low-rank subspace fine-tuning in Appendix A.

Acknowledge

This work is in part supported by the NSF AI Institute for Foundations of Machine Learning (IFML).

References

[1] Emanuele Paci and Martin Karplus. Unfolding proteins by external forces and temperature: the importance of topology and energetics. *Proceedings of the National Academy of Sciences*, 97(12):6521–6526, 2000.

[2] Fabrizio Pucci, Jean Marc Kwasigroch, and Marianne Rooman. Scoop: an accurate and fast predictor of protein stability curves as a function of temperature. *Bioinformatics*, 33(21):3415–3422, 2017.

[3] Huali Cao, Jingxue Wang, Liping He, Yifei Qi, and John Z Zhang. Deepddg: predicting the stability change of protein point mutations using neural networks. *Journal of chemical information and modeling*, 59(4):1508–1514, 2019.

[4] GD Haki and SK Rakshit. Developments in industrially important thermostable enzymes: a review. *Bioresource technology*, 89(1):17–34, 2003.

[5] Marieke E Bruins, Anja EM Janssen, and Remko M Boom. Thermozymes and their applications. *Applied biochemistry and biotechnology*, 90(2):155–186, 2001.

[6] Sven Frokjaer and Daniel E Otzen. Protein drug stability: a formulation challenge. *Nature reviews drug discovery*, 4(4):298–306, 2005.

[7] Swati Kapoor, Aasima Rafiq, and Savita Sharma. Protein engineering and its applications in food industry. *Critical reviews in food science and nutrition*, 57(11):2321–2329, 2017.

[8] Jie Huang, Peng Zhao, Xin Jin, Yiwen Wang, Haotian Yuan, and Xinyuan Zhu. Enzymatic biofuel cells based on protein engineering: recent advances and future prospects. *Biomaterials science*, 8(19):5230–5240, 2020.

[9] C Von der Osten, S Branner, S Hastrup, L Hedegaard, MD Rasmussen, H Bisgaard-Frantzen, S Carlsen, and JM Mikkelsen. Protein engineering of subtilisins to improve stability in detergent formulations. *Journal of biotechnology*, 28(1):55–68, 1993.

[10] Changhee Cho, So Young Choi, Zi Wei Luo, and Sang Yup Lee. Recent advances in microbial production of fuels and chemicals using tools and strategies of systems metabolic engineering. *Biotechnology advances*, 33(7):1455–1466, 2015.

[11] Amro Abd-Al-Fattah Amara. Pharmaceutical and industrial protein engineering: where we are? *Pakistan Journal of Pharmaceutical Sciences*, 26(1), 2013.

[12] Carla CCR De Carvalho. Enzymatic and whole cell catalysis: finding new strategies for old processes. *Biotechnology advances*, 29(1):75–83, 2011.

[13] Marirosa Mora and John L Telford. Genome-based approaches to vaccine development. *Journal of Molecular Medicine*, 88(2):143–147, 2010.

[14] Hongyuan Lu, Daniel J Diaz, Natalie J Czarnecki, Congzhi Zhu, Wantae Kim, Raghav Shroff, Daniel J Acosta, Bradley R Alexander, Hannah O Cole, Yan Zhang, et al. Machine learning-aided engineering of hydrolases for pet depolymerization. *Nature*, 604(7907):662–667, 2022.

[15] Miguel Alcalde, Manuel Ferrer, Francisco J Plou, and Antonio Ballesteros. Environmental biocatalysis: from remediation with enzymes to novel green processes. *TRENDS in Biotechnology*, 24(6):281–287, 2006.

[16] Fabrizio Pucci and Marianne Rooman. Stability curve prediction of homologous proteins using temperature-dependent statistical potentials. *PLoS computational biology*, 10(7):e1003689, 2014.

[17] Gang Li, Kersten S Rabe, Jens Nielsen, and Martin KM Engqvist. Machine learning applied to predicting microorganism growth temperatures and enzyme catalytic optima. *ACS synthetic biology*, 8(6):1411–1420, 2019.

[18] Fabrizio Pucci and Marianne Rooman. Physical and molecular bases of protein thermal stability and cold adaptation. *Current opinion in structural biology*, 42:117–128, 2017.

[19] Soroosh Pouyan, Milad Lagzian, and Mohammad Hossein Sangtarash. Enhancing thermostabilization of a newly discovered α-amylase from bacillus cereus gl96 by combining computer-aided directed evolution and site-directed mutagenesis. *International Journal of Biological Macromolecules*, 197:12–22, 2022.

[20] Anna Jarzab, Nils Kurzawa, Thomas Hopf, Matthias Moerch, Jana Zecha, Niels Leijten, Yangyang Bian, Eva Musiol, Melanie Maschberger, Gabriele Stoehr, et al. Meltome atlas—thermal proteome stability across the tree of life. *Nature methods*, 17(5):495–503, 2020.

[21] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *International Conference on Machine Learning*, pages 6438–6447. PMLR, 2019.

[22] Tianlong Chen, Yu Cheng, Zhe Gan, Jianfeng Wang, Lijuan Wang, Zhangyang Wang, and Jingjing Liu. Adversarial feature augmentation and normalization for visual recognition. *arXiv preprint arXiv:2103.12171*, 2021.

[23] Tao Li, Yingwen Wu, Sizhe Chen, Kun Fang, and Xiaolin Huang. Subspace adversarial training. *arXiv preprint arXiv:2111.12229*, 2021.

[24] Tianlong Chen, Yu Cheng, Zhe Gan, Jingjing Liu, and Zhangyang Wang. Data-efficient gan training beyond (just) augmentations: A lottery ticket perspective. *Advances in Neural Information Processing Systems*, 34, 2021.

[25] Vincent GH Eijsink, Sigrid Gåseidnes, Torben V Borchert, and Bertus Van Den Burg. Directed evolution of enzyme stability. *Biomolecular engineering*, 22(1-3):21–30, 2005.

[26] Rafael Couñago, Stephen Chen, and Yousif Shamoo. In vivo molecular evolution reveals biophysical origins of organismal fitness. *Molecular cell*, 22(4):441–449, 2006.

[27] Hein J Wijma, Robert J Floor, and Dick B Janssen. Structure-and sequence-analysis inspired engineering of proteins for enhanced thermostability. *Current opinion in structural biology*, 23(4):588–594, 2013.

[28] 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.

[29] Mohammed AlQuraishi. Alphafold at casp13. *Bioinformatics*, 35(22):4862–4865, 2019.

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

[31] Tanya Barrett, Karen Clark, Robert Gevorgyan, Vyacheslav Gorelenkov, Eugene Gribov, Ilene Karsch-Mizrachi, Michael Kimelman, Kim D Pruitt, Sergei Resenchuk, Tatiana Tatusova, et al. Bioproject and biosample databases at ncbi: facilitating capture and organization of metadata. *Nucleic acids research*, 40(D1):D57–D63, 2012.

[32] Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C. Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *bioRxiv*, 2019.

[33] Jesse Vig, Ali Madani, Lav R Varshney, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani. Bertology meets biology: Interpreting attention in protein language models. *arXiv preprint arXiv:2006.15222*, 2020.

[34] Roshan Rao, Jason Liu, Robert Verkuil, Joshua Meier, John F. Canny, Pieter Abbeel, Tom Sercu, and Alexander Rives. Msa transformer. *bioRxiv*, 2021.

[35] 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.

[36] Nadav Brandes, Dan Ofer, Yam Peleg, Nadav Rappoport, and Michal Linial. Proteinbert: A universal deep-learning model of protein sequence and function. *Bioinformatics*, 38(8):2102–2110, 2022.

[37] Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*, 2021.

[38] Chloe Hsu, Robert Verkuil, Jason Liu, Zeming Lin, Brian Hie, Tom Sercu, Adam Lerer, and Alexander Rives. Learning inverse folding from millions of predicted structures. *bioRxiv*, 2022.

[39] David Shortle. One sequence plus one mutation equals two folds. *Proceedings of the National Academy of Sciences*, 106(50):21011–21012, 2009.

[40] Marcus Resch, Harald Striegl, Eva Maria Henssler, Madhumati Sevvana, Claudia Egerer-Sieber, Emile Schiltz, Wolfgang Hillen, and Yves A Muller. A protein functional leap: how a single mutation reverses the function of the transcription regulator tetr. *Nucleic acids research*, 36(13):4390–4401, 2008.

[41] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[42] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[43] Milo M. Lin and Ahmed H. Zewail. Hydrophobic forces and the length limit of foldable protein domains. *Proceedings of the National Academy of Sciences*, 109:9851–9856, 2012.

[44] Xuhua Xia. Domains and functions of spike protein in sars-cov-2 in the context of vaccine design. *Viruses*, 13(1):109, 2021.

[45] Najeeb Khan and Ian Stavness. Sparseout: Controlling sparsity in deep networks. In *Canadian conference on artificial intelligence*, pages 296–307. Springer, 2019.

[46] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[47] Carl Rasmussen and Zoubin Ghahramani. Occam's razor. *Advances in neural information processing systems*, 13, 2000.

[48] Wenda Zhou, Victor Veitch, Morgane Austern, Ryan P Adams, and Peter Orbanz. Non-vacuous generalization bounds at the imagenet scale: a pac-bayesian compression approach. *arXiv preprint arXiv:1804.05862*, 2018.

[49] Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. In *International Conference on Machine Learning*, pages 254–263. PMLR, 2018.

[50] Shuai Zhang, Meng Wang, Sijia Liu, Pin-Yu Chen, and Jinjun Xiong. Why lottery ticket wins? a theoretical perspective of sample complexity on sparse neural networks. *Advances in Neural Information Processing Systems*, 34, 2021.

[51] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

[52] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[53] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):1–37, 2011.

[54] Tianyi Zhou and Dacheng Tao. Greedy bilateral sketch, completion & smoothing. In *Artificial Intelligence and Statistics*, pages 650–658. PMLR, 2013.

[55] Xiyu Yu, Tongliang Liu, Xinchao Wang, and Dacheng Tao. On compressing deep models by low rank and sparse decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7370–7379, 2017.

[56] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[57] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13001–13008, 2020.

[58] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.

[59] Oleksandr Kolomiyets, Steven Bethard, and Marie-Francine Moens. Model-portability experiments for textual temporal analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, volume 2, pages 271–276. ACL; East Stroudsburg, PA, 2011.

[60] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

[61] William Yang Wang and Diyi Yang. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 2557–2563, 2015.

[62] Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*, 2019.

[63] Vladimir Gligorijević, P Douglas Renfrew, Tomasz Kosciolek, Julia Koehler Leman, Daniel Berenberg, Tommi Vatanen, Chris Chandler, Bryn C Taylor, Ian M Fisk, Hera Vlamakis, et al. Structure-based protein function prediction using graph convolutional networks. *Nature communications*, 12(1):1–14, 2021.

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

[65] Xuxi Chen, Tianlong Chen, Yu Cheng, Weizhu Chen, Zhangyang Wang, and Ahmed Hassan Awadallah. Dsee: Dually sparsity-embedded efficient tuning of pre-trained language models, 2021.

[66] Andrew Waterhouse, Martino Bertoni, Stefan Bienert, Gabriel Studer, Gerardo Tauriello, Rafal Gumienny, Florian T Heer, Tjaart A P de Beer, Christine Rempfer, Lorenza Bordoli, et al. Swiss-model: homology modelling of protein structures and complexes. *Nucleic acids research*, 46(W1):W296–W303, 2018.

[67] Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 1085–1097, 2019.

[68] Brian Kuhlman, Gautam Dantas, Gregory C Ireton, Gabriele Varani, Barry L Stoddard, and David Baker. Design of a novel globular protein fold with atomic-level accuracy. *science*, 302(5649):1364–1368, 2003.

[69] Po-Ssu Huang, Scott E Boyken, and David Baker. The coming of age of de novo protein design. *Nature*, 537(7620):320–327, 2016.

[70] Brian Kuhlman and Philip Bradley. Advances in protein structure prediction and design. *Nature Reviews Molecular Cell Biology*, 20(11):681–697, 2019.

[71] Vladimir Gligorijevic, Daniel Berenberg, Stephen Ra, Andrew Watkins, Simon Kelow, Kyunghyun Cho, and Richard Bonneau. Function-guided protein design by deep manifold sampling. *bioRxiv*, 2021.

[72] Richard W Shuai, Jeffrey A Ruffolo, and Jeffrey J Gray. Generative language modeling for antibody design. *bioRxiv*, 2021.

[73] Alex Hawkins-Hooker, David T Jones, and Brooks Paige. Msa-conditioned generative protein language models for fitness landscape modelling and design. In *Machine Learning for Structural Biology Workshop, NeurIPS*, 2021.

[74] Samuel C Hoffman, Vijil Chenthamarakshan, Kahini Wadhawan, Pin-Yu Chen, and Payel Das. Optimizing molecules using efficient queries from property evaluations. *Nature Machine Intelligence*, 4(1):21–31, 2022.

[75] Christoffer Norn, Basile IM Wicky, David Juergens, Sirui Liu, David Kim, Doug Tischer, Brian Koepnick, Ivan Anishchenko, Foldit Players, David Baker, et al. Protein sequence design by conformational landscape optimization. *Proceedings of the National Academy of Sciences*, 118(11):e2017228118, 2021.

[76] Sheng Wang, Siqi Sun, Zhen Li, Renyu Zhang, and Jinbo Xu. Accurate de novo prediction of protein contact map by ultra-deep learning model. *PLoS computational biology*, 13(1):e1005324, 2017.

[77] Kevin Karplus. Sam-t08, hmm-based protein structure prediction. *Nucleic acids research*, 37(suppl_2):W492–W497, 2009.

[78] David T Jones and Shaun M Kandathil. High precision in protein contact prediction using fully convolutional neural networks and minimal sequence features. *Bioinformatics*, 34(19):3308–3315, 2018.

[79] Diego del Alamo, Davide Sala, Hassane Mchaourab, and Jens Meiler. Sampling the conformational landscapes of transporters and receptors with alphafold2. *bioRxiv*, 2021.

[80] Raghav Shroff, Austin W Cole, Daniel J Diaz, Barrett R Morrow, Isaac Donnell, Ankur Annapareddy, Jimmy Gollihar, Andrew D Ellington, and Ross Thyer. Discovery of novel gain-of-function mutations guided by structure-based deep learning. *ACS synthetic biology*, 9(11):2927–2935, 2020.

[81] Inyup Paik, Phuoc HT Ngo, Raghav Shroff, Daniel J Diaz, Andre C Maranhao, David JF Walker, Sanchita Bhadra, and Andrew D Ellington. Improved bst dna polymerase variants derived via a machine learning approach. *Biochemistry*, 2021.

[82] Stefano Panno, Slavica Matić, Antonio Tiberini, Andrea Giovanni Caruso, Patrizia Bella, Livio Torta, Raffaele Stassi, and Salvatore Davino. Loop mediated isothermal amplification: principles and applications in plant virology. *Plants*, 9(4):461, 2020.

[83] Frances H Arnold. Directed evolution: bringing new chemistry to life. *Angewandte Chemie International Edition*, 57(16):4143–4148, 2018.

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

[85] Bruce J Wittmann, Yisong Yue, and Frances H Arnold. Informed training set design enables efficient machine learning-assisted directed protein evolution. *Cell systems*, 12(11):1026–1045, 2021.

[86] Thaiesha A Wright, Jamie M Stewart, Richard C Page, and Dominik Konkolewicz. Extraction of thermodynamic parameters of protein unfolding using parallelized differential scanning fluorimetry. *The journal of physical chemistry letters*, pages 553–558, 2017.

[87] Chi-Wei Chen, Jerome Lin, and Yen-Wei Chu. istable: off-the-shelf predictor integration for predicting protein stability changes. In *BMC bioinformatics*, volume 14, pages 1–14. BioMed Central, 2013.

[88] Emidio Capriotti, Piero Fariselli, and Rita Casadio. I-mutant2. 0: predicting stability changes upon mutation from the protein sequence or structure. *Nucleic acids research*, 33(suppl_2):W306–W310, 2005.

[89] Carlos HM Rodrigues, Douglas EV Pires, and David B Ascher. Dynamut: predicting the impact of mutations on protein conformation, flexibility and stability. *Nucleic acids research*, 46(W1):W350–W355, 2018.

[90] Douglas EV Pires, David B Ascher, and Tom L Blundell. Duet: a server for predicting effects of mutations on protein stability using an integrated computational approach. *Nucleic acids research*, 42(W1):W314–W319, 2014.

[91] Vijaya Parthiban, M Michael Gromiha, and Dietmar Schomburg. Cupsat: prediction of protein stability upon point mutations. *Nucleic acids research*, 34(suppl_2):W239–W242, 2006.

[92] Elizabeth H Kellogg, Andrew Leaver-Fay, and David Baker. Role of conformational sampling in computing mutation-induced changes in protein structure and stability. *Proteins: Structure, Function, and Bioinformatics*, 79(3):830–838, 2011.

[93] Douglas EV Pires, David B Ascher, and Tom L Blundell. mcsm: predicting the effects of mutations in proteins using graph-based signatures. *Bioinformatics*, 30(3):335–342, 2014.

[94] Manuel Giollo, Alberto JM Martin, Ian Walsh, Carlo Ferrari, and Silvio CE Tosatto. Neemo: a method using residue interaction networks to improve prediction of protein stability upon mutation. *BMC genomics*, 15(4):1–11, 2014.

[95] Hahnbeom Park, Philip Bradley, Per Greisen Jr, Yuan Liu, Vikram Khipple Mulligan, David E Kim, David Baker, and Frank DiMaio. Simultaneous optimization of biomolecular energy functions on features from small molecules and macromolecules. *Journal of chemical theory and computation*, 12(12):6201–6212, 2016.

[96] Joost Schymkowitz, Jesper Borg, Francois Stricher, Robby Nys, Frederic Rousseau, and Luis Serrano. The foldx web server: an online force field. *Nucleic acids research*, 33(suppl_2):W382–W388, 2005.

[97] Yves Dehouck, Aline Grosfils, Benjamin Folch, Dimitri Gilis, Philippe Bogaerts, and Marianne Rooman. Fast and accurate predictions of protein stability changes upon mutations using statistical potentials and neural networks: Popmusic-2.0. *Bioinformatics*, 25(19):2537–2543, 2009.

[98] Emmi Jokinen, Markus Heinonen, and Harri Lähdesmäki. mgpfusion: predicting protein stability changes with gaussian process kernel learning and data fusion. *Bioinformatics*, 34(13):i274–i283, 2018.

[99] Catherine L Worth, Robert Preissner, and Tom L Blundell. Sdm—a server for predicting effects of mutations on protein stability and malfunction. *Nucleic acids research*, 39(suppl_2):W215–W222, 2011.

[100] Bian Li, Yucheng T Yang, John A Capra, and Mark B Gerstein. Predicting changes in protein thermodynamic stability upon point mutation with deep 3d convolutional neural networks. *PLoS computational biology*, 16(11):e1008291, 2020.

[101] Emmanuel Paradis and Klaus Schliep. ape 5.0: an environment for modern phylogenetics and evolutionary analyses in r. *Bioinformatics*, 35(3):526–528, 2019.

[102] Felix Krueger and Simon R Andrews. Bismark: a flexible aligner and methylation caller for bisulfite-seq applications. *bioinformatics*, 27(11):1571–1572, 2011.

[103] Ramin Dehghanpoor, Evan Ricks, Katie Hursh, Sarah Gunderson, Roshanak Farhoodi, Nurit Haspel, Brian Hutchinson, and Filip Jagodzinski. Predicting the effect of single and multiple mutations on protein structural stability. *Molecules*, 23(2):251, 2018.

[104] Lei Jia, Ramya Yarlagadda, and Charles C Reed. Structure based thermostability prediction models for protein single point mutations with machine learning tools. *PloS one*, 10(9):e0138022, 2015.

[105] 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.

[106] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[107] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

[108] 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.

[109] Adam Riesselman, Jung-Eun Shin, Aaron Kollasch, Conor McMahon, Elana Simon, Chris Sander, Aashish Manglik, Andrew Kruse, and Debora Marks. Accelerating protein design using autoregressive generative models. *BioRxiv*, page 757252, 2019.

[110] Dhananjay Kimothi, Akshay Soni, Pravesh Biyani, and James M Hogan. Distributed representations for biological sequence analysis. *arXiv preprint arXiv:1608.05949*, 2016.

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

[112] Tristan Bepler and Bonnie Berger. Learning protein sequence embeddings using information from structure. *arXiv preprint arXiv:1902.08661*, 2019.

[113] 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.

[114] Baris E Suzek, Yuqi Wang, Hongzhan Huang, Peter B McGarvey, Cathy H Wu, and UniProt Consortium. Uniref clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932, 2015.

[115] Martin Steinegger and Johannes Söding. Clustering huge protein sequence sets in linear time. *Nature communications*, 9(1):1–8, 2018.

[116] Sara El-Gebali, Jaina Mistry, Alex Bateman, Sean R Eddy, Aurélien Luciani, Simon C Potter, Matloob Qureshi, Lorna J Richardson, Gustavo A Salazar, Alfredo Smart, et al. The pfam protein families database in 2019. *Nucleic acids research*, 47(D1):D427–D432, 2019.

[117] Brian L Hie, Duo Xu, Varun R Shanker, Theodora UJ Bruun, Payton A Weidenbacher, Shaogeng Tang, and Peter S Kim. Efficient evolution of human antibodies from general protein language models and sequence information alone. *bioRxiv*, 2022.

[118] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 506–516, 2017.

[119] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[120] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[121] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[122] Demi Guo, Alexander M Rush, and Yoon Kim. Parameter-efficient transfer learning with diff pruning. *arXiv preprint arXiv:2012.07463*, 2020.

[123] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[124] Beidi Chen, Tri Dao, Kaizhao Liang, Jiaming Yang, Zhao Song, Atri Rudra, and Christopher Re. Pixelated butterfly: Simple and efficient sparse training for neural network models. *arXiv preprint arXiv:2112.00029*, 2021.

[125] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021.

[126] Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. *arXiv preprint arXiv:1804.08838*, 2018.

[127] Samet Oymak, Zalan Fabian, Mingchen Li, and Mahdi Soltanolkotabi. Generalization guarantees for neural networks via harnessing the low-rank structure of the jacobian. *arXiv preprint arXiv:1906.05392*, 2019.

[128] Guy Gur-Ari, Daniel A Roberts, and Ethan Dyer. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*, 2018.

[129] Ziheng Wang, Jeremy Wohlwend, and Tao Lei. Structured pruning of large language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6151–6162, 2020.

[130] Beidi Chen, Tri Dao, Eric Winsor, Zhao Song, Atri Rudra, and Christopher Ré. Scatterbrain: Unifying sparse and low-rank attention approximation. *arXiv preprint arXiv:2110.15343*, 2021.

[131] Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. Whitening sentence representations for better semantics and faster retrieval. *arXiv preprint arXiv:2103.15316*, 2021.

[132] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, page 1100612. International Society for Optics and Photonics, 2019.

[133] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. 2018.

[134] Carol A Rohl, Charlie EM Strauss, Kira MS Misura, and David Baker. Protein structure prediction using rosetta. In *Methods in enzymology*, volume 383, pages 66–93. Elsevier, 2004.

[135] Brandon Frenz, Steven M Lewis, Indigo King, Frank DiMaio, Hahnbeom Park, and Yifan Song. Prediction of protein mutational free energy: benchmark and sampling improvements increase classification accuracy. *Frontiers in bioengineering and biotechnology*, page 1175, 2020.

[136] Rahul Nikam, A Kulandaisamy, K Harini, Divya Sharma, and M Michael Gromiha. Prothermdb: thermodynamic database for proteins and mutants revisited after 15 years. *Nucleic Acids Research*, 49(D1):D420–D424, 2021.

[137] Jan Stourac, Juraj Dubrava, Milos Musil, Jana Horackova, Jiri Damborsky, Stanislav Mazurenko, and David Bednar. Fireprotdb: database of manually curated protein stability data. *Nucleic acids research*, 49(D1):D319–D324, 2021.

[138] Julia Koehler Leman, Brian D Weitzner, Steven M Lewis, Jared Adolf-Bryfogle, Nawsad Alam, Rebecca F Alford, Melanie Aprahamian, David Baker, Kyle A Barlow, Patrick Barth, et al. Macromolecular modeling and design in rosetta: recent methods and frameworks. *Nature methods*, 17(7):665–680, 2020.

A Related Works

Protein Engineering. In general, the factors that make one protein functional and simultaneously thermostable compared to another are complex and unknown for a specific target protein. The literature on computational models for in silico screening of the thermodynamic impact of a mutation is exhaustive. In the development of biopharmaceuticals and biocatalyst, improving the thermodynamic properties of the target protein is a common early goal in most protein engineering campaigns. A protein's thermodynamic stability is commonly represented by its Gibb's free energy and/or melting temperature. The protein engineering community has been designing computational tools to enable in silico screening of mutations for several decades and these tools can be classified into biophysical models and statistical/machine learning-based models. Computational protein engineering models are used to guide the search for optimized sequences. Despite recent progress, these methods have limited utility at reliably ranking sequences, especially at discerning small changes in thermodynamic properties. Although physics-based methods have been shown to reach reasonable accuracy, they are computationally demanding and low-throughput making them intractable to apply to large-scale in silico screening of protein variants.

Recent Success in AI + Protein Engineering. Protein engineering with machine learning is usually formulated as an energy-guided refinement process by maximizing a pre-defined energy function with changing input data (*e.g.,* torsion angle, 1D amino acid sequence, or 2D contact map) [29, 68, 69, 70]. A group of works train a generative model or autoencoder and then optimize the continuous hidden representation to maximize some given objectives [*e.g.,* 71, 72, 73, 74]. Another category of works train models to map the input data to the target property (e.g. temperature, energy, etc.), and then optimize the discrete input space with combinatorial optimization [75]. These methods have been applied to different kinds of input data, *e.g.,* 1D structural features [75, 76, 77], torsion angles, and contact maps [78, 79].

Beyond promising performance in simulation experiments, in the last two years, unsupervised structure-based deep learning techniques have also shown their ability to learn from natural protein structures and then generalize to guide the engineering of structurally diverse proteins. [70, 80]. For example, Paik et al. [81] used MutCompute to improve the thermostability of a DNA polymerase in order to enable faster loop-mediated isothermal amplification (LAMP) [82] assays to be carried out at 73°C, a temperature where the commercial counterpart enzyme, BST 2.0, is inactive. Additionally, Lu et al. [14] engineered two Polyethylene terephthalate (PET) hydrolases (PETase and Cutinase) with MutCompute to improve its thermal and pH stability, resulting in FAST-PETase: a PET hydrolase that can fully degrade post-consumer PET within 48 hours.

Directed Evolution and Guided Directed Evolution. The most common strategy to engineer and stabilize a protein is directed evolution. Directed evolution leverages natural evolution where random mutagenesis is synergized with screening to identify variants with improved fitness for the desired phenotype, essentially performing a greedy local search to optimize protein fitness [83]. For directed evolution, it is necessary to design a high-throughput assay that selects mutants enriched for your target phenotype, which is not always possible. Recent progress has been made in combining machine learning approaches with the data collection capability of directed evolution in order to accelerate iterative rounds of directed evolution [84, 85]. Here, machine learning is used to ease the experimental screening burden by evaluating proteins in silico. Briefly, machine learning-guided directed evolution (MLDE) works by iteratively training an ML model on a small number of variants (10^1-10^2) from a combinatorial library and is then used to infer the remaining variance in the library where the variants with the highest predicted fitness are then experimentally evaluated and then added to the training set for the subsequent round. However, the need to consistently generate libraries and expand the training set does not completely alleviate the screening burden. Furthermore, Wittman et al. [85] showed empirical evidence that sequence-based embeddings from transformer models can accelerate MLDE and that an MSA transformer [34] embedding outperformed sequence embeddings of larger models such as ProtBert-BFD and ESM-1B [28].

Protein Thermostability Prediction. To enhance a protein's stability, $\Delta\Delta$G and ΔT_m are common metrics by molecular biologists, enzymologists, and protein engineers. $\Delta\Delta$G evaluates the changes in free energy between a protein and a mutated variant. While ΔT_m evaluates the change in thermal tolerance between two protein variants. The two are related through the Van 't Hoff equation [86] and it is common to obtain $\Delta\Delta$G from T_m measurements [*e,g,* 87, 88, 89, 90, 91].

Most studies lack accurate large-scale thermostability data. For example, deepDDG [3] is trained on $5,766$ manually-curated $\Delta\Delta$G measurements across 242 proteins, while one of their test set contains 173 experimental melting temperature changes (ΔT_m) to assess how well deepDDG correlated with ΔT_m [3]. Previously, researchers use empirical physics-based energy contributions [92], torsion angles [91], or graph-based distance patterns [93] as features and apply different models, *e.g.,* physical models, residue interaction networks [94], SVMs [87], to predict the thermodynamics $\Delta\Delta$G. However, none of these methods have shown strong generalization.

Protein Thermostability Dataset. In protein engineering, improving the stability of a protein is a goal in nearly all protein engineering campaigns and can be a deciding factor in the commercialization of a biocatalyst. Thus, there is plenty of literature on computational algorithms attempting to model this phenotype. Over the last two decades, several algorithms that predict the thermodynamic impact of mutations have been developed [e.g., 95, 96, 97, 98, 99, 3, 100, 101, 102, 103] and can be classified as either biophysical models or statistical/machine learning models. Biophysical models utilize amino acid interactions and conformational/rotamer sampling of the protein structure to determine the changes in stability. The primary two biophysical models are Rosetta [95] and FoldX [96]. These methods have shown good performance and utility in several applications but fail to capture narrow changes in stability ($<$ 1kcal/mol), which is commonly observed for many mutations. Furthermore, they are computationally demanding and have low throughput, hindering their application to large-scale in silico screening for identifying stabilizing point mutations [100, 101]. Machine learning tools trained to evaluate the impact of a mutation on the stability of a protein are growing in popularity [e.g., 98, 97, 99, 3, 100, 101, 102, 103]. Some notable machine learning models include PopMusic [97], DeepddG [3], ThermoNet [100], SDM [99], and mGPfusion [98]. These models are trained in a supervised fashion to predict available experimental data, usually sourced from ProTherm [104] or manually curated datasets from the literature, using either evolutionary or structural input features with some models utilizing both types of features. Thus, all use a relatively small training set, with DeepDDG [3] having the largest at 5700 manually curated data points, due to the limited amount of experimentally validated point mutations. So far, we have yet to see a stability predicting algorithm that leverages a language model's learned representation to predict the stability effects of mutations.

Sequence- and Structured-based Protein Models. Motivated by self-supervised pre-training in natural language community [*e.g.,* 105, 106, 107], a variety of recent works formulate the protein pre-training tasks as sequence self-supervised learning, *e.g.,* with auto-encoding [72], auto-regressive [*e.g.,* 32, 35, 108, 109], skip-gram language model [110], mask prediction [33, 36] or amino acid contrastive learning objectives [111], similarity metric learning [112, 113]. Although using different architectures (*e.g.,* LSTMs, transformers, graph networks), different pre-trained datasets (*e.g.,* UniRef100 [114], BFD [115], Pfam [116]), and different objectives, these models convert input sequences into per amino acid representations. Most recently, thanks to AlphaFoldV2 and its folded dataset [30], ESM-IF [38] proposes to learn the 3D model with inverse folding, which predicts sequences from backbone 3D structures. The input is 3D coordinate information for each residues backbone N, CA, and C atoms (present in every amino acid). Passing the input through the ESM-IF permutation invariant model, the model outputs a probability distribution for each amino acid at each residue position.

Sequence-based Pretraining Methods. ESM-1b [28], MSA Transformer [34], and [33] are several popular pre-trained transformer protein language models, which regard one amino acid as a token. [72] trains an antibody language model with antibody data. These works have been applied to simulation experiments. For example, [117] performs ESM-guided affinity maturation of seven diverse antibodies, screening 20 or fewer variants of each antibody across only two rounds of evolution. [72] demonstrates that their model can be applied to generate synthetic libraries that may accelerate the discovery of therapeutic antibody candidates in real experiments.

Sparse and Low-rank Subspace Fine-tuning. *Sparsity-aware fine-tuning* is typically leveraged towards the goal of parameter-efficiency [118, 119, 120, 121], which only tunes a few of model weights and keep the rest unchanged. [122] embeds the sparsity into fine-tuning with a differentiable approximation of ℓ_0 regularization. [65] introduces sparse tuning patterns via classical pruning methods [123]. [124] pre-defines the shape of sparsity patterns and learns their combination. Another alternative solution for efficient fine-tuning is to constrain weight updates within a *low-rank subspace* [52, 65, 125]. Numerous literature [55, 126, 127, 128] point out that the intrinsic low-rank dimensionality of trained over-parameterized models. [129, 52] focus on imposing explicit low-rank structures when transferring pre-trained models to diverse downstream tasks, leading to considerable parameter efficiency. Recently, several pioneering studies [65, 124, 130] consider combining sparsity

and low-rank decomposition for improved efficient training. However, we investigate these two structural priors from a distinctive perspective, *i.e.,* their implicit regularization effects against data scarcity and domain shifts.

B More Method Details for SAP

During the pretraining, the language model only learns from context-token pairs, instead of learning a global representation for one sentence (or one protein). Once we target the global representation of one sentence (or one protein), we should learn from a loss function that directly compares one sentence and another [37]. Similar to the sentence representations of BERT [105], we notice that the protein representations learned by sequence models are not uniformly distributed in the latent space [37]. Therefore, we adopt contrastive learning in the protein embedding space to distribute the representations more uniformly and let the 3D model inject information into the sequence model. We refer readers to [37, 131] for details about motivations.

Momentum Encoder to Construct Negative Examples. Following [42], the momentum model $\mathcal{F}^{\text{mom}}$ is a slowly updated model with update rule: $\mathcal{F}^{\text{mom}} \leftarrow \alpha\mathcal{F}^{\text{mom}} + (1-\alpha)\mathcal{F}$, where $\alpha \in [0, 1)$ is the momentum coefficient. During the optimization of Equation 1, to save the computation budget for negative examples, we adopt the memory bank [42] to construct enough negative examples. The Neg set is taken to be a queue of $\mathcal{F}'(\boldsymbol{x}_j)$ representations from previous mini-batches.

C More Implementation Details

Training Details . *Baselines.* 3D GCN [63] is trained for 20 epochs, with an initial learning rate of 1×10^{-4} that decays by 0.1 at the 10th epoch. For TAPE [64], we train it for 4 epochs, with an initial learning rate of 1×10^{-4} and a linear decay schedule. As for ESM-1B, we follow [28] and only train a linear classification head on the top of ESM-1B backbone. The head tuning consists of 4 epochs with an initial learning rate of 2×10^{-2} and an OneCycle [132] decay scheduler. A training batch size of 4 is used across all experiments. Since we start tuning from pre-trained models [64, 28], the performance of TAPE and ESM-1B are usually saturated after $2 \sim 3$ epochs.

▷ *SAP.* We use AlphaFoldDB [30] for SAP protein pre-training. We filter the data with sequence length and data quality and finally get 270K data. ESM-1B [32] and ESM-IF [38] backbone are used to process the sequence and 3D coordinate inputs, and an average pooling layer is applied to the final-layer token representations of ESM models and get protein embeddings. A momentum encoder with $\tau = 1.0$, momentum encoder coefficient $\alpha = 0.9999$ and memory bank of size $65,536$ is used and the model is trained for 4 epochs, with AdamW optimizer [133], weight decay 10^{-12}, batch size 512 and an initial learning rate 10^{-6} decayed with OneCycle [132] decay scheduler.

▷ *FST.* For the proposed FST, we choose an initial learning rate of 1×10^{-2} for the linear classification head, and an initial learning rate of 1×10^{-3} for training the low-rank and sparse components in ESM-1B. Other training configurations inherit the same ones from tuning ESM-1B. As for the hyperparameters of rank r and the number of non-zero elements $|\Omega|$ in FST, we perform screenings on $r \in \{4, 8, 16\}$ and $|\Omega| \in \{16, 32, 64, 128\}$, where we choose $(r, |\Omega|) = (4, 64)$ on `HP-S²C2/C5` and $(r, |\Omega|) = (8, 64)$ on `HP-S` and `HP-SC2`. Meantime, we adopt a one-step gradient ascent with a step size of 1×10^{-5} to generate worst-case feature augmentations, and apply them to the last two layers of ESM-1B, as suggested in [22].

Computing resources. Experiments use `Tesla V100-SXM2-32GB` GPUs as computing resources. Each experiment can be run with a single `V100` GPU.

C.1 Experimental Details about Protein Editing

Search Space. Based our fine-tuned classifiers, we generate 20 possible single point mutations at each position (i) in a natural protein sequence $p^{(i)}$ and then evaluate whether the resulted $\hat{p}_k^{(i)}$ ($k \in \{1, \cdots, 20\}$) successfully leads to improved thermostability. The corresponding successful rate is used as our evaluation metric.

Optimization Method. We adopt a recently-proposed well-known method, PWWS [67]. PWWS replaces tokens base on a ranking score $H(x, x_i^*) = \phi(S(x))_i \Delta P_i^*$, where x is the input sequence,

Table A5: We report the correlation coefficient between Rosetta $\Delta\Delta$G and FireProtDB ΔT$_m$, $\Delta\Delta$G.

FireProtDB	Rosetta	Data Size	Spearman (%)	Pearson (%)
$\Delta\Delta$G	with backbone relaxation $\Delta\Delta$G	3,399	44.93	21.36
$\Delta\Delta$G	w/o. backbone relaxation $\Delta\Delta$G	3,248	28.59	3.31
ΔT$_m$	with backbone relaxation $\Delta\Delta$G	981	-35.91	-3.98
ΔT$_m$	w/o. backbone relaxation $\Delta\Delta$G	1,018	-13.54	0.26

x_i^* stands for replace the amino acid at index i, ΔP_i^* denotes the change in classification probability after replacing with x_i^*, and $\phi(S(x))_i$ is the saliency score.

Dataset Descriptions. Previously, numbers of benchmarks have been proposed for thermostability or $\Delta\Delta$G prediction. Some of them using Rosetta [134] predicted score, [135] and others create datasets, *e.g.*, ProThermDB [136] based on experiment records. The curation of large high-quality thermostability datasets is still ongoing, and protein stability experiments are time-consuming and expensive [137]. We set up our benchmark using FireProtDB [137], which is a superset of published experiments records and is the most up-to-date dataset to our knowledge. We download the latest version of FireProtDB, clean the duplicates, and extract the ΔTm values. We remain all the values with $\|\Delta Tm\| \geq 1$ to make sure that the temperature change is large enough and not random noise. Finally, we get 961 data instances with $\|\Delta Tm\| > 1$.

Benchmark Quality. Rosetta is an academic framework for computational modeling and analysis of protein structures. We utilize the Rosetta Cartesian $\Delta\Delta$G application [95] to assess how well established biophysical computational tools can recapitulate the thermostability dataset - FireProtDB. The Cartesian $\Delta\Delta$G application calculates the change in folding energy of a mutation. First, we relax the crystal structures of the proteins in FireProtDB with an unconstrained FastRelax as recommended in [138], which allows the backbone and side-chain atoms to move slightly to be better accommodated into the chosen Rosetta score, "ref2015_cart". Cartesian $\Delta\Delta$G then mutates the residues as specified in the database, packs side-chain conformations, and does gradient-based minimization of the atomic coordinates. Using the values from the Rosetta score, we can then calculate the $\Delta\Delta$G of mutation. Any mutations that failed during the calculation for any reason were discarded.

We calculate the correlation between simulation $\Delta\Delta$G and experimental $\Delta\Delta$G in Table A5. Allowing backbone relaxation during structure generation improves the Spearman $\Delta\Delta$G to 44.93 from 28.59 and the Spearman ΔT$_m$ to -35.91 from -13.54. Compare with our results in Table 2, we observe that our proposal offers better candidate mutations in terms of the spearman correlation coefficient.

Training Settings. For zero-shot transfer, we directly apply the models trained on `HP-S` to $\hat{\mathcal{P}}$. For fine-tuning, we train all the models with AdamW optimizer, 5×10^{-2} weight decay, and the OneCycle learning rate schedule. TAPE models are trained with batch size 8, learning rate 10^{-3} and 10 epochs, while ESM models are trained with batch size 16, learning rate 10^{-3} and 20 epochs. For our editting, we use models trained on $\mathcal{P}$. When fine-tuning ESM models, we re-initialize the final linear head.

D More Experimental Results

Table A6: Head Tuning versus Full Tuning of ESM-1B on thermostability classification and regression. 95% confidence interval are computed via the 10-fold evaluation on `HP-S`^{2}`C2/C5` and 3 replicates on `HP-S`.

ESM-1B	`HP-S`^{2}`C2`		`HP-S`^{2}`C5`		`HP-S`	
	Accuracy	Spearman	Accuracy	Spearman	Accuracy	Spearman
Head Tuning	**91.19**±0.47	**0.890**±0.018	**83.26**±1.54	**0.712**±0.043	**69.50**±0.16	**0.809**±0.001
Full Tuning	88.75±1.29	0.797±0.029	77.97±2.46	0.503±0.058	65.88±0.89	0.615±0.050

Head Tuning versus Full Tuning of ESM-1B. In Table A6, only tuning the head of ESM-1B consistently achieves better performance than tuning the whole ESM-1B across all three datasets. A potential explanation is that the fully tuned ESM-1B tends to overfit due to the relatively small size of `HotProtein` (182K), where the pre-training dataset has around 22M protein sequences.

Table A7: Hyperparameter tuning of the Mixup augmentation. ESM-1B equipped with our proposals is adopted with HP-S^{2}C2 and HP-S^{2}C5.

Settings/Acc. (%)	HP-S^{2}C2	HP-S^{2}C5
$\lambda \sim \mathcal{B}(0.2, 0.1)$	0.8547	0.8638
$\lambda \sim \mathcal{B}(0.1, 0.2)$	0.8600	0.8527
$\lambda \sim \mathcal{B}(0.2, 0.5)$	0.8675	0.8596

Table A8: Evaluation results of our proposals with ESM-1B on the Meltome Atlas benchmark.

Methods	Spearman	Pearson
ESM-1B + Ours	**0.4560**	**0.6866**
ESM-1B	0.3874	0.5331
TAPE	0.3076	0.3132

Hyperparameter Tuning. In general, we find that our method is not very sensitive to hyper-parameter tuning to yield good results. Specifically: (1) Many hyper-parameters were left at default values. For example, for SAP, we used the hyper-parameters from [42] without optimizing. (2) For the other hyper-parameters (e.g. batch size, learning rate), we used standard 10-fold cross-validation for selection. (3) Our algorithm is not sensitive to the choice of hyper-parameters, as shown in Table 2, our FST achieves considerable improvements for various hyper-parameter choices. (4) We provide more ablations for Mixup augmentation in Table A7, where λ is driven from beta distributions $\mathcal{B}$.

Our Proposal on Other Benchmarks. To demonstrate the generalization ability of our methods, we further perform the zero-shot transferring on Meltome Atlas (we train the model on HotProtein and directly evaluate the model's performance on 8K Meltome Atlas data). We notice that our method still achieves better results in Table A8.

Table A9: Comparison with more baselines of MLP and ESM-1B without any pre-training.

Settings/Acc. (%)	HP-S^{2}C2	HP-S^{2}C5
ESM-1B w.o. pre-training	0.7808	0.6652
ESM-1B w. pre-training	**0.9119**	**0.8326**
MLP	0.6931	0.6378

Table A10: More approaches to inject the structure information with ESM-1B.

Settings/Acc. (%)	HP-S^{2}C2	HP-S^{2}C5
Ours	**0.9236**	**0.8625**
Add	0.8873	0.8093
Concat	0.9001	0.8186

More Baselines: Small Neural Networks and ESM-1B without Pre-training. We conduct extra experiments with 1) ESM-1B without the pretraining weights (with 2 times larger number of training iterations than a MLP's) and 2) a small neural network, 3-layer MLP (256 Dimensional Embedding Layer $\rightarrow$ 3 Layer MLP $\rightarrow$ Average Pool) in Table A9. These two approaches achieve much worse results than ESM-1B baseline.

Other Approaches to Inject Structure Information. As shown in Table A10, we notice that directly concatenating or adding the final-layer representations of ESM-IF (structure information) and ESM-1B (sequence information) comes to slightly worse results.

More Comparisons: End-to-end or Freezing Some Layers Tuning. We provide additional results with ESM-1B on HP-S^{2}C2 and HP-S^{2}C5. "First 1 / 3 Layers" indicates the layers close to the input. To achieve better results, we tune the backbone learning rate with 10-fold cross-validation for these numbers. Table A11 tells us only tuning the head leads to superior performance.

Evaluation on Balanced Test Sets. We perform extra evaluations on manually balanced test sets. Results are summarized in Table A12, where our methods consistently show superior performance.

Table A11: Comparison with the end-to-end and partially frozen tuning with ESM-1B on HP-S^{2}C2 and HP-S^{2}C5.

Settings/Acc. (%)	HP-S^{2}C2	HP-S^{2}C5
End-to-End	0.8875	0.7797
Freeze First 1 / 3 Layers	0.8927	0.8018
Freeze First 2 / 3 Layers	0.9011	0.8195
Tuning Heads	0.9119	0.8326
Tuning Heads + Ours	**0.9236**	**0.8625**

Table A12: Evaluation on manually balanced datasets.

Settings/Acc. (%)	HP-S^{2}C2	HP-S^{2}C5	HP-S
3D GCN	0.7633	0.5004	-
TAPE	0.8117	0.5535	0.6370
ESM-IF1	0.7752	0.6032	-
ESM-1B	0.8605	0.8029	0.6980
ESM-1B + Ours	**0.9308**	**0.8294**	**0.7517**

Ablation on the Feature Aggregation Methods. In Table A13, we conduct ablation studies on the feature aggregation methods with ESM-1B on HP-S^{2}C2 and HP-S^{2}C5, including average pooling, max pooling, and no pooling. Results are summarized in the below table. We observe that the average pooling outperforms other aggregation options.

Comparison to ℓ_2 (Ridge Regression) and ℓ_1 (Lasso) Regularization. We conduct comparisons to ℓ_2 (ridge regression) and ℓ_1 (Lasso) regularization with ESM-1B on HP-S^{2}C5. Results are in Table A14. We observe that additional regularizers may lead to performance degradations.

Table A13: Ablation on the feature aggregation methods with ESM-1B on HP-S^{2}C2 and HP-S^{2}C5.

Settings/Acc. (%)	HP-S^{2}C2	HP-S^{2}C5
Average Pooling	**0.9119**	**0.8326**
Max Pooling	0.8819	0.3750
No Pooling	0.9084	0.7808

Table A14: Comparisons with the ℓ_1 and ℓ_2 regularized tuning for ESM-1B on HP-S^{2}C5.

Settings/Acc. (%)	HP-S^{2}C5
ESM-1B	**0.8326**
ESM-1B + ℓ_2 (Ridge Regression)	0.7375
ESM-1B + ℓ_1 (Lasso)	0.3808

E More Dataset Details

Data Collection and Process. All the data in the NCBI bioproject is accumulated from all scientists who publish data that makes it into the NCBI database. Thus, there is a lot of duplication and variation in data entry. Due to the vast number of organisms and multiple strains for organisms, we removed duplicates by taking the first observation in the NCBI bioproject that had a consistent quantitative optimal growth temperature and temperature classification.

We remove all sequences that are greater than 1500 amino acids, since most proteins of interest for engineering fall within this range and it greatly simplifies clustering. During the clustering, we cluster across all organisms in the same temperature bin in order to remove redundancy across organisms and kept representative sequences from each cluster in order to not bias sampling to a specific organism.

There are numerous organism-specific idiosyncrasies present (such as sequence homology between organisms due to their evolutionary relationship) in each organism proteome that has nothing to do with thermostability and instead their unique adaptation to their environment. However, there is no clear-cut way to identify these features/sequences and remove them. Therefore, we expect there to be numerous organism-specific idiosyncrasies present (such as sequence homology between organisms due to their evolutionary relationship) in each organism proteome that has nothing to do with thermostability and instead their unique adaptation to their environment. In practice, we do not further filter the data to reduce evolutionary differences. Instead, we cluster proteins based on sequence similarity across all organisms within a temperature bin and select the representative sequence of each cluster to avoid biasing our dataset to a particular organism.

Domain Shift. For domain shifts, it includes two perspectives: ❶ *Between HotProtein and FireprotDB*. Annotating a proteome with the optimal growth temperature of the organism provides us a lower bound soft label for training on nearly 200K sequences (i.e., HotProtein). However, we expect to observe a domain shift when fine-tuning on experimentally curated stability datasets (i.e., FireprotDB) since these labels accurately represent a protein variant thermodynamic properties, unlike our coarse, optimal growth temperature label. We use the classic tool, i.e., **cd-hit** [6], to compute the protein sequence similarity (1) between FireprotDB and Hotprotein; (2) within FireprotDB. We find the similarity between FireportDB and Hotprotein (0.1928) is much lower than the one within FireprotDB (0.2504). It suggests the substantial domain shifts, echoed with our paper's descriptions. ❷ *Within HotProtein*. HotProtein covers 230 in different species where proteins have different functionality and structures. For example, we expect to observe domain shifts between proteins sampled from the eukaryotic species (primarily unicellular fungal) and the prokaryotic organisms.

Data Scarcity. For the data scarcity, we measure our framework on two kinds of data-limited scenarios: (1) We have evaluated our proposals on subsampled HP-S2C2 and HP-S2C5, and demonstrated their effectiveness. HP-S2C2 and HP-S2C5 only have around 2K sequences ($\frac{2}{183}$ of the whole HotProtein dataset). (2) We further examine our approaches on FireprotDB which is a manually

[6] http://weizhong-lab.ucsd.edu/cd-hit/

curated database of the protein stability data for single-point mutants. It only contains 0.2K natural protein amino acid sequences which serve as the training set in our case. Table 4 shows that our methods lead to consistent performance improvements.

An Organism Lives is Correlates with the Thermostability of the Protein Sequences of that Organism. The Meltome Atlas provides thermal proteome profiling (TPP) for 13 organisms [20]. Although TPP is not the melting temperature of the protein since it is not measured with purified proteins and requires the protein to become insoluble upon denaturation (proteins can remain soluble even after denaturation), it provides empirical evidence that for mesophilic, and thermophilic prokaryotic organisms, the optimal growth temperature of an organism is very close to the lower bound of the thermal stability of that organism. This is not the case for eukaryotic and psychrophilic prokaryotes where the Meltome Atlas shows a >10C gap between the optimal growth temperature and the lower bound of protein stability in these organisms. Thus, given the available proteome experimental data, we assume that using the optimal growth condition for prokaryotic mesophiles and thermophiles provides an accurate lower bound melting temperature for their respective proteome. Furthermore, although our method utilizes the lower bound of a proteome's thermal stability as a label, we demonstrate that it provides a sufficient learning signal to improve performance on the manually-curated experimental dataset: FireProtDB.

F Discussion of Broader Impact

This research aims to predict the thermal stability of proteins from the sequence and structural data and predict thermostabilizing mutation designs. Improving a protein's robustness to thermal challenges can often be the deciding factor for the commercialization of a biocatalyst. Furthermore, the ability to rapidly thermostabilize a protein will open the door for the engineering of a broader range of biotechnologically relevant enzymes and therapeutics, which can have a profound impact on the chemical, agricultural, food, and pharmaceutical industries. It is well documented that functional residues tend to be destabilizing residues and engineering function into a protein can quickly destabilize the protein. Thus, it is common to first improve a protein's stability prior to introducing destabilizing mutations. Our work attempts to accelerate the initial stabilization of a target protein to enable downstream functional protein engineering.