Preferential Bayesian Optimisation for Protein Design with Ranking-Based Fitness Predictors

Alex Hawkins-Hooker[1,2] * **Paul Duckworth**[1] **Oliver Bent**[1] †

[1]InstaDeep Ltd [2]University College London

Abstract

Ranking-based loss functions have recently been shown to improve the quality of predictions of fitness landscapes for both standard supervised deep learning models and fine-tuned protein language models. We consider the implications of this finding for protein design with Bayesian optimisation. We investigate uncertainty quantification techniques applicable to protein language models fine-tuned with ranking losses, and show that they offer competitive calibration to CNN ensembles while demonstrating superior predictive performance. Finally, we demonstrate how uncertainty-aware ranking-based models can be exploited for protein design within the framework of preferential Bayesian optimisation.

1 Introduction

Building accurate models of protein sequence-function mappings from experimental datasets is fundamental to data-driven protein engineering. While the learning of fitness predictors has most often been formulated as a regression problem, recent works [2, 15] have reported superior results when training models using ranking losses, where the continuous fitness values are converted into a set of binary pairwise comparisons between sequences. This formulation has been found to be particularly well-suited to fine-tuning protein language models (PLMs) in the low data setting often encountered in protein design [15]. These observations motivate the study of such ranking-based predictors as surrogate models in model-guided design approaches such as Bayesian optimisation (BO). However, using ranking-based surrogates raises two challenges. First, for surrogate models to be useful for BO, they must not only be accurate but also provide well-calibrated uncertainty estimates [7, 8], a consideration which has not been addressed in prior work on PLM fine-tuning. Second, models trained using ranking-based losses do not provide estimates of fitness values, preventing their straightforward use as surrogates in combination with standard BO acquisition functions like expected improvement.

In this work we explore the suitability of ranking-based fitness prediction models as surrogate models for BO. We propose ensemble-based uncertainty quantification strategies suitable for application to protein language models fine-tuned with ranking losses, and show that they lead to strong performance on a set of few-shot protein fitness prediction tasks. We further show that by using acquisition functions from the literature on preferential Bayesian optimisation, the accuracy of ranking-based predictors can be harnessed to accelerate black-box protein design.

*Work done during an internship at InstaDeep Ltd

†Corresponding author: o.bent@instadeep.com

Machine Learning for Structural Biology Workshop, NeurIPS 2023.

2 Methods

2.1 Ranking-based training of fitness landscape predictors

While mean squared error (MSE) has been widely used as a loss function in training sequence-based predictive models of fitness landscapes, two recent works have advocated the use of ranking-based loss functions [2, 15]. In particular, they suggest parameterising a Bradley-Terry model with a learned function of the sequence. The Bradley-Terry model represents the probability that a given sequence x_i has higher fitness y than another sequence x_j by parameterising a binary classifier via the difference in scores of each sequence under a learned scoring function $s_\theta(x)$:

$$p(y(x_i) > y(x_j)) = \sigma(\beta(s_\theta(x_i) - s_\theta(x_j))) \,, \tag{1}$$

where β is an inverse temperature parameter and σ is the logistic sigmoid function. The parameters θ of the scoring function are optimised by minimising the binary cross entropy loss across all pairwise comparisons between sequences. In this way, fitness prediction for a dataset of size N is converted from a regression problem with N labels into a binary classification problem with $N \times N$ labels.

Fine-tuning pre-trained protein language models with a ranking loss We focus on autoregressive protein language models, since they define a natural scoring function for complete sequences via their (per amino acid) log-likelihood:

$$s_\theta(x) = \frac{1}{L} \sum_{i=1}^{L} \log p(x_i | x_{<i}) \,. \tag{2}$$

We follow [15] in using this log-likelihood-based scoring function to parameterise the Bradley-Terry model in Equation 1, and fine-tuning all parameters. Since the log-likelihoods of autoregressive protein language models are strong zero-shot predictors of the fitness effects of mutations [21, 22], the difference in log-likelihoods can already produce an effective pairwise classifier at initialisation.

2.2 Uncertainty quantification for fine-tuned protein language models

Ensembles of CNNs initialized from different random seeds have become a de-facto standard for applications that require uncertainty estimation, for example when training a surrogate model for BO [7, 8, 23]. However, simple ensembles of fine-tuned PLMs are likely to underestimate uncertainty since each member of the ensemble is initialised to the same pre-trained weights. We therefore evaluate two additional ensembling strategies for fine-tuned PLMs:

- **Bagging:** each ensemble member is fine-tuned on a randomly sampled subset of training data. We use 66% of the training data to train each ensemble member, finding this provided a better tradeoff between uncertainty estimation and performance than other tested values (50%, 75%).
- **Conditioning set ensembles (cond.):** we propose to exploit the direction invariance commonly used in training autoregressive protein language models [22, 21] to create ensembles of scoring functions. For each ensemble member, we sample a random residue index on which to split each protein into two segments. We score the amino acids in the first segment in reverse, conditioned on the second segment, and score the amino acids in the second segment in the forwards direction, conditioned on the amino acids in the first segment (Appendix A.3).

2.3 Preferential Bayesian optimisation

Preferential Bayesian optimisation [6] is a family of methods for finding the global optimum of a function $y(x)$ which can only be queried indirectly via binary-valued pairwise comparisons $x_1 \succ x_2$ representing which of two inputs has the higher (unseen) function value ($x_1 \succ x_2 := y(x_1) > y(x_2)$). The central idea is to introduce a surrogate model representing a distribution over the values of a latent function f which induces the binary observations via a probabilistic output model such as the Bradley-Terry model. Importantly, the lack of an explicit model of the observations y in this formulation means that standard acquisition functions such as expected improvement cannot be used without modification. Further background on preferential BO is provided in Appendix A.6.

Using ranking-based fitness predictors as preferential surrogates In typical protein design settings we *are* able to directly query values of the function we want to optimise, for example by performing a wet-lab experiment. However, when training the ranking-based fitness predictors introduced in Section 2.1 on the results of such experiments, we convert the space of observations associated with the set of fitness measurements from a space of scalar values to a space of pairwise comparisons. We show in Section 3 that re-framing the problem in this way is empirically well-motivated, often leading to improved fitness prediction performance. While ranking-based models do not produce direct predictions of y, they do introduce a latent scoring function $f(x) \equiv s_\theta(x)$. In order to exploit the performance benefits of these ranking models, we therefore propose to adopt the framework of preferential BO, by using ensembles of ranking-based predictors to represent uncertainty in this latent function (Appendix A.5).

Preferential acquisition functions We assume we are operating in a batch setting in which q sequences are selected for evaluation in each round. A complete set of pairwise comparisons between batch members can be inferred from the returned values. Siivola et al. [27] consider appropriate choices of acquisition function for preferential BO in the complete-batch setting, recommending the use of Thompson sampling for problems with high dimensionality or large batch sizes. Batches of points are selected using Thompson sampling (TS) by sampling q realisations of the posterior latent function from $p(f|\mathcal{D})$, and returning the locations of the maxima of these function realisations. We use a version of TS adapted to ensembles (Appendix A.7) as our main acquisition strategy, but also discuss and evaluate alternative preferential acquisition functions in Appendix A.9.

Local acquisition function optimisation The high-dimensional, discrete nature of protein sequences poses a challenge for BO methods, which are known to suffer from the curse of dimensionality [29]. Recent work has demonstrated that local variants of BO can help avoid the over-exploration that otherwise occurs in high dimensions [5, 29]. The main idea is to optimise the acquisition function only over a local neighbourhood of the current point [5, 18, 20]. We therefore construct batches of acquisitions by optimising acquisition functions within a neighbourhood containing sequences having up to 3 amino acid substitutions relative to the current sequence, similar to [8]. Acquisition function optimisation is performed using a stochastic hill climbing strategy described in Appendix A.8.

3 Fitness landscape prediction

Datasets We evaluate few-shot fitness predictions on three fitness landscapes: the GB1 and AAV landscapes from FLIP [4], and the GFP landscape of [26] (Appendix A.2). For the latter we create our own FLIP-style splits designed to assess biologically relevant forms of generalisation. To test models in the low data setting, we sample 128 training sequences at random from each training split. We select two splits on which to report results per landscape, covering a variety of split types across all landscapes. Where test splits are larger than 5000 sequences, we use a random sample of 5000 sequences to evaluate all models. For the AAV splits, we only provide CNN models with the 28 residue mutated region, and not the additional context from the wild-type sequence, following [2].

Fitness predictors We use ProGen2 (small) checkpoints [21] to instantiate pre-trained language model scoring functions $s_\theta(x)$. We then use these scoring functions to fine-tune fitness predictors as described in Section 2.1. We also report results when using the log-likelihood directly as a regression function and fine-tuning with MSE. As an alternative to fine-tuned PLMs, we also report results for CNN ensembles, which performed competitively in a recent benchmark of uncertainty quantification methods [7]. We trained CNN ensembles with both ranking loss and MSE.

Since ProGen2 results are affected by the choice of aggregation method when computing the scoring function (per token log-likelihood or sequence-level log-likelihood in Equation 2, equivalent to different choices of β in Equation 1), for each protein and each ensembling strategy we chose the aggregation method leading to the best Spearman correlation on the reported split.

Metrics To measure prediction quality we use Spearman correlation. For ranking-based methods the correlation is computed between the observed function values and the values of the latent scoring function $s_\theta(x)$. For these methods we also report the negative log-likelihood (equivalent to binary cross entropy) on test set pairwise comparisons as a proxy measure of the quality of the models'

Table 1: Spearman correlation on the splits. Ensembles (of size K) are below the dividing line. Single model results are averages across members of the corresponding ensemble.

	GB1		AAV		GFP	
	2-vs-rest	low-vs-high	1-vs-many	low-vs-high	1-vs-many	2-vs-many
CNN regression	0.148	0.080	0.382	-0.081	0.236	0.169
CNN ranking	0.480	0.065	0.508	0.025	0.214	0.161
ProGen2 regression	0.390	0.036	0.672	0.073	**0.598**	**0.554**
ProGen2 ranking	**0.654**	**0.162**	**0.732**	**0.073**	0.588	0.520
CNN regression (K=10)	0.201	0.045	0.553	-0.147	0.483	0.325
CNN ranking (K=10)	0.511	0.101	0.586	0.080	0.359	0.273
ProGen2 regression (K=7)	0.391	0.036	0.692	**0.127**	0.601	**0.555**
ProGen2 ranking (K=7)	**0.661**	0.197	**0.780**	0.118	0.611	0.533
ProGen2 ranking bagging (K=7)	0.656	**0.227**	0.755	0.055	0.601	0.486
ProGen2 ranking cond. (K=7)	**0.661**	0.207	0.774	0.091	**0.628**	0.541

uncertainty estimates. When computing this value for ensembles, classification probabilities are averaged, as we found this to be more effective than averaging logits before applying a sigmoid (Appendix A.5).

Results We report Spearman correlations for selected splits in Table 1, and test set negative log-likelihoods in Table 2. Several broad patterns emerge. First, we reproduce the finding that direct fine-tuning of protein language models is a powerful approach in the low-data regime [15], with fine-tuned ProGen2 models outperforming CNN ensembles across almost all splits. Second, we offer further evidence that ranking losses often, but not always, outperform regression losses, both for CNNs (extending the results of [2] to a lower-data setting) and fine-tuned PLMs. Direct regression-based fine-tuning of ProGen2 likelihoods also often surpasses CNNs, and sometimes outperforms ranking-based fine-tuning. As expected, ensembling strategies lead to further improvements, with ProGen2 ensembles achieving the highest performance in every split, although the relative performance of different ensembling methods is less clear.

Comparing Table 1 and Table 2, ensembling approaches often lead to substantial reductions in test set negative log-likelihood compared to single models, indicating more accurate quantification of uncertainty. Methods which produce more diverse ensembles (bagging and conditioning set ensembles) tend to have much larger effects than simply training with different random seeds, highlighting the importance of the choice of ensembling strategy.

Table 2: Calibration of pairwise classifications measured via negative log-likelihood. Ensembles (of size K) are below the dividing line. Single model results are averages across members of the corresponding ensemble.

	GB1		AAV		GFP	
	2-vs-rest	low-vs-high	1-vs-many	low-vs-high	1-vs-many	2-vs-many
CNN ranking	**1.22**	**0.931**	0.742	1.43	**0.714**	**0.813**
ProGen2 ranking	1.30	0.960	**0.486**	**0.763**	3.34	3.45
CNN ranking (K=10)	0.842	0.738	0.556	0.758	**0.654**	**0.668**
ProGen2 ranking (K=7)	1.04	0.793	**0.437**	0.706	2.01	2.28
ProGen2 ranking bagging (K=7)	0.824	0.770	0.462	**0.692**	0.927	0.947
ProGen2 ranking cond. (K=7)	**0.598**	**0.741**	**0.437**	0.704	0.977	0.950

4 Fitness landscape optimisation

To test the effectiveness of ranking-based predictive strategies for design applications, we use them as surrogate models in a Bayesian optimisation setting. Following previous work, we task BO

strategies with finding sequences which optimise the predictions of a black box fitness oracle, given a small subset of the landscape as training data.

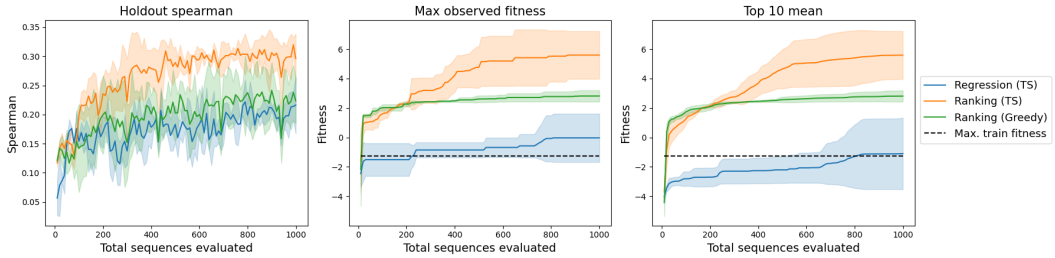

Figure 1: Comparison of BO strategies for optimising AAV fitness oracle. Panel details in text.

Task To compare optimisation strategies, we assess their success in maximising the fitness of the AAV capsid protein VP1, as estimated by a fitness oracle trained on the extensive (roughly 284,000 mutants) experimental fitness landscape generated by Bryant et al. [3]. The oracle is a fine-tuned TAPE model that was previously used to evaluate the PEX optimiser [23]. We allow optimisation routines to propose arbitrary combinations of substitutions within the 28 residue region which was mutated to generate the empirical landscape, and pass the resulting 28 residue sequences to the oracle. Before the optimisation process starts, surrogate models are trained on 96 low-fitness sequences from the empirical landscape (we use a fitness threshold of -1.2 following [22]).

Optimisation strategies We compare a preferential BO strategy using an ensemble of ranking CNNs to a standard BO approach using an ensemble of regression CNNs. In each case, we use ensembles of size 10 and propose batches of 10 sequences to be evaluated by the oracle in each round using Thompson sampling. All methods optimise the acquisition function within a local 3-mutation neighbourhood of the highest-scoring sequence from the previous batch (or from the training set, if no batches have been acquired). Acquisition function optimisation over these local neighbourhoods is performed using a stochastic hill-climbing strategy described in Appendix A.8. As an uncertainty-free baseline, we include a 'greedy' strategy, in which stochastic hill climbing is used to select sequences in the local neighbourhood having the highest average predicted scores under the ensemble.

Results In Figure 1, we plot the maximum of the fitness values returned by the oracle after n total queries (centre), and the average of the top 10 values returned after n queries (right). To show the evolution of surrogate model performance, we also plot the Spearman correlation of the surrogate model on a fixed heldout set of 512 sequences (left). Whereas BO using the regression-based ensemble as a surrogate (blue line) achieves limited progress during optimisation, using a more accurate ranking-based ensemble as a surrogate (orange) allows the preferential BO strategy to optimise fitness far beyond the range of values in the training set. Thompson sampling's automatic balancing of exploration and exploitation leads to significantly better performance than a purely exploitative greedy strategy (green), confirming the value of explicit uncertainty estimates.

5 Discussion

Achieving sample-efficient optimisation of protein fitness landscapes requires the development of accurate fitness predictors as well as of design algorithms that exploit these predictions to guide the choice of sequences to test. We demonstrated that the previously reported differences in fitness prediction capacity between models trained with regression losses and models trained with ranking-based losses can lead to significant differences in performance between BO algorithms using these models as surrogates. We hope that this motivates future work on designing black-box optimisation algorithms explicitly tailored towards making use of the most effective few-shot prediction strategies.

References

[1] Surojit Biswas, Grigory Khimulya, Ethan C. Alley, Kevin M. Esvelt, and George M. Church. "Low-N protein engineering with data-efficient deep learning". *Nature Methods* 18.4 (2021).

[2] David H. Brookes, Jakub Otwinowski, and Sam Sinai. "Contrastive losses as generalized models of global epistasis". *arXiv:2305.03136 [q-bio.PE]* (2023).

[3] Drew H. Bryant, Ali Bashir, Sam Sinai, Nina K. Jain, Pierce J. Ogden, Patrick F. Riley, George M. Church, Lucy J. Colwell, and Eric D. Kelsic. "Deep diversification of an AAV capsid protein by machine learning". *Nature Biotechnology* 39.6 (2021).

[4] Christian Dallago, Jody Mou, Kadina E. Johnston, Bruce J. Wittmann, Nicholas Bhattacharya, Samuel Goldman, Ali Madani, and Kevin K. Yang. "FLIP: Benchmark tasks in fitness landscape inference for proteins". *bioRxiv 2021.11.09.467890* (2022).

[5] David Eriksson, Michael Pearce, Jacob Gardner, Ryan D Turner, and Matthias Poloczek. "Scalable Global Optimization via Local Bayesian Optimization". *Advances in Neural Information Processing Systems* (2019).

[6] Javier González, Zhenwen Dai, Andreas Damianou, and Neil D. Lawrence. "Preferential Bayesian Optimization". *Proceedings of the 34th International Conference on Machine Learning* (2017).

[7] Kevin P. Greenman, Ava P. Amini, and Kevin K. Yang. "Benchmarking Uncertainty Quantification for Protein Engineering". *bioRxiv 2023.04.17.536962* (2023).

[8] Nate Gruver, Samuel Stanton, Polina Kirichenko, Marc Finzi, Phillip Maffettone, Vivek Myers, Emily Delaney, Peyton Greenside, and Andrew Gordon Wilson. "Effective Surrogate Models for Protein Design with Bayesian Optimization". *ICML Workshop on Machine Learning in Computational Biology* (2021).

[9] Brian L. Hie, Varun R. Shanker, Duo Xu, Theodora U. J. Bruun, Payton A. Weidenbacher, Shaogeng Tang, Wesley Wu, John E. Pak, and Peter S. Kim. "Efficient evolution of human antibodies from general protein language models". *Nature Biotechnology* (2023).

[10] Brian L. Hie and Kevin K. Yang. "Adaptive machine learning for protein engineering". *Current Opinion in Structural Biology* 72 (2022).

[11] Chloe Hsu, Hunter Nisonoff, Clara Fannjiang, and Jennifer Listgarten. "Learning protein fitness models from evolutionary and assay-labeled data". *Nature Biotechnology* 40.7 (2022).

[12] Mingyang Hu, Fajie Yuan, Kevin Yang, Fusong Ju, Jin Su, Hui Wang, Fei Yang, and Qiuyang Ding. "Exploring evolution-aware & -free protein language models as protein function predictors". *Advances in Neural Information Processing Systems* (2022).

[13] Moksh Jain, Emmanuel Bengio, Alex Hernandez-Garcia, Jarrid Rector-Brooks, Bonaventure F. P. Dossou, Chanakya Ajit Ekbote, Jie Fu, Tianyu Zhang, Michael Kilgour, Dinghuai Zhang, Lena Simine, Payel Das, and Yoshua Bengio. "Biological Sequence Design with GFlowNets". *Proceedings of the 39th International Conference on Machine Learning* (2022).

[14] Asif Khan, Alexander I. Cowen-Rivers, Antoine Grosnit, Derrick-Goh-Xin Deik, Philippe A. Robert, Victor Greiff, Eva Smorodina, Puneet Rawat, Rahmad Akbar, Kamil Dreczkowski, Rasul Tutunov, Dany Bou-Ammar, Jun Wang, Amos Storkey, and Haitham Bou-Ammar. "Toward real-world automated antibody design with combinatorial Bayesian optimization". *Cell Reports Methods* 3.1 (2023).

[15] Ben Krause, Nikhil Naik, Wenhao Liu, and Ali Madani. "Don't throw away that linear head: Few-shot protein fitness prediction with generative models". *OpenReview* (2021).

[16] Ben Krause, Subu Subramanian, Tom Yuan, Marisa Yang, Aaron Sato, and Nikhil Naik. "Improving Antibody Affinity Using Laboratory Data with Language Model Guided Design". *bioRxiv 2023.09.13.557505* (2023).

[17] Lin Li, Esther Gupta, John Spaeth, Leslie Shing, Rafael Jaimes, Emily Engelhart, Randolph Lopez, Rajmonda S. Caceres, Tristan Bepler, and Matthew E. Walsh. "Machine learning optimization of candidate antibody yields highly diverse sub-nanomolar affinity antibody libraries". *Nature Communications* 14.1 (2023).

[18] Natalie Maus, Haydn Jones, Juston Moore, Matt J. Kusner, John Bradshaw, and Jacob Gardner. "Local Latent Space Bayesian Optimization over Structured Inputs". *Advances in Neural Information Processing Systems* (2022).

[19] Joshua Meier, Roshan Rao, Robert Verkuil, Jason Liu, Tom Sercu, and Alex Rives. "Language models enable zero-shot prediction of the effects of mutations on protein function". *Advances in Neural Information Processing Systems* (2021).

[20] Quan Nguyen, Kaiwen Wu, Jacob Gardner, and Roman Garnett. "Local Bayesian optimization via maximizing probability of descent". *Advances in Neural Information Processing Systems* 35 (2022).

[21] Erik Nijkamp, Jeffrey Ruffolo, Eli N. Weinstein, Nikhil Naik, and Ali Madani. "ProGen2: Exploring the Boundaries of Protein Language Models". *arXiv:2206.13517 [cs.LG]* (2022).

[22] Pascal Notin, Mafalda Dias, Jonathan Frazer, Javier Marchena Hurtado, Aidan N. Gomez, Debora Marks, and Yarin Gal. "Tranception: Protein Fitness Prediction with Autoregressive Transformers and Inference-time Retrieval". *Proceedings of the 39th International Conference on Machine Learning* (2022).

[23] Zhizhou Ren, Jiahan Li, Fan Ding, Yuan Zhou, Jianzhu Ma, and Jian Peng. "Proximal Exploration for Model-guided Protein Sequence Design". *Proceedings of the 39th International Conference on Machine Learning* (2022).

[24] Adam J. Riesselman, John B. Ingraham, and Debora S. Marks. "Deep generative models of genetic variation capture the effects of mutations". *Nature Methods* 15.10 (2018).

[25] 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". *Proceedings of the National Academy of Sciences* 118.15 (2021).

[26] Karen S. Sarkisyan, Dmitry A. Bolotin, Margarita V. Meer, Dinara R. Usmanova, Alexander S. Mishin, George V. Sharonov, Dmitry N. Ivankov, Nina G. Bozhanova, Mikhail S. Baranov, Onuralp Soylemez, Natalya S. Bogatyreva, Peter K. Vlasov, Evgeny S. Egorov, Maria D. Logacheva, Alexey S. Kondrashov, Dmitry M. Chudakov, Ekaterina V. Putintseva, Ilgar Z. Mamedov, Dan S. Tawfik, Konstantin A. Lukyanov, and Fyodor A. Kondrashov. "Local fitness landscape of the green fluorescent protein". *Nature* 533.7603 (2016).

[27] Eero Siivola, Akash Kumar Dhaka, Michael Riis Andersen, Javier Gonzalez, Pablo Garcia Moreno, and Aki Vehtari. "Preferential Batch Bayesian Optimization". *2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP)* (2021).

[28] Samuel Stanton, Wesley Maddox, Nate Gruver, Phillip Maffettone, Emily Delaney, Peyton Greenside, and Andrew Gordon Wilson. "Accelerating Bayesian Optimization for Biological Sequence Design with Denoising Autoencoders". *Proceedings of the 39th International Conference on Machine Learning* (2022).

[29] Kaiwen Wu, Kyurae Kim, Roman Garnett, and Jacob R. Gardner. "The Behavior and Convergence of Local Bayesian Optimization". *arXiv:2305.15572 [cs.LG]* (2023).

[30] Nicholas C Wu, Lei Dai, C Anders Olson, James O Lloyd-Smith, and Ren Sun. "Adaptation in protein fitness landscapes is facilitated by indirect paths". *eLife* 5 (2016).

[31] Ziyue Yang, Katarina A. Milas, and Andrew D. White. "Now What Sequence? Pre-trained Ensembles for Bayesian Optimization of Protein Sequences". *bioRxiv 2022.08.05.502972* (2022).

A Appendix

A.1 Related work

Zero and few-shot fitness prediction: Generative sequence models trained either on individual families or across all of natural protein space have been established as strong zero-shot predictors of protein fitness, with autoregressive language models currently the most-effective general approaches [24, 19, 22]. Few-shot fitness prediction has been studied using strategies that include combining zero-shot predictors with supervised models [11], as well as directly fine-tuning protein language models [15, 25], or using language model embeddings as input to task-specific predictors [4, 12, 7].

Exploiting uncertainty for protein engineering: Several works have proposed to use variants of BO for designing biological sequences, including proteins [8, 13, 14, 28, 31], with a range of uncertainty-driven design approaches reviewed in [10]. Most relevant to the present work, Gruver et al. [8] study various choices of surrogate model for protein design with BO, finding CNN ensembles to be particularly robust to the kinds of distribution shift encountered during online design. More recently, Greenman, Amini, and Yang [7] studied a range of uncertainty quantification strategies applied to models trained either directly on sequences or on frozen language model embeddings.

A number of works have also attempted to combine uncertainty estimates with protein language models to solve specific design tasks, in settings ranging from zero [9] or few-shot design [1, 16] to single-round design given large training sets of sequence-fitness pairs [17].

While our work shares motivation with these prior works, our main interest is in studying the applicability of the currently most promising fitness prediction approaches (i.e. protein language model fine-tuning and ranking-based losses) to multi-round protein design tasks in a controlled setting.

A.2 Datasets

We evaluate on experimental protein fitness landscapes generated using Deep Mutational Scanning (DMS) techniques, and studied in prior works on machine learning for fitness prediction. These datasets consist of sets of protein sequences together with experimentally determined continuous 'fitness' values. The biological significance of these values depends on the biological assay used to generate fitness measurements. We briefly summarise the characteristics of the three datasets below.

GB1 We use GB1 splits from FLIP [4]. These provide biologically relevant splits of the original GB1 dataset [30], which contains a combinatorially complete set of measurements of the fitness values of a set of sequences derived by mutating the B1 domain of protein G at 4 positions. The fitness values are generated by a binding assay. We report performance on the 2-vs-rest split, in which the training set consists of sequences having 2 or fewer mutations and the test set consists of sequences with 3 or 4 mutations, and on the low-vs-high split, in which the training set contains variants with fitnesses lower than wild-type and the test set contains variants with fitnesses above wild-type.

AAV We also use FLIP splits for the AAV dataset. The original dataset [3] contains approximately $284,000$ variants of the adeno-associated virus 2 (AAV2) VP1 capsid protein with associated fitness values. The variants are mutated in a common 28-residue region, and the fitness value measures the viral packaging success. Both substitutions and insertions are included in the dataset, which contains significant mutational diversity within the mutated region (average Levenshtein distance to WT is 12.5). The FLIP 1-vs-many split contains a subset of single mutants as the training set and a subset of higher-order mutants as the test set, while the low-vs-high split contains mutants having fitness values lower than the fitness value observed for the wild-type in the training set, and mutants having fitness values higher than the WT in the test set.

GFP The GFP fitness landscape reported by [26] has been widely studied in prior works on machine learning for protein fitness prediction and design. We downloaded the processed dataset from ProteinGym, consisting of approximately 52,000 sequences containing up to 15 mutations. We constructed our own FLIP-style splits: 1-vs-many (single mutant training set, higher-order test set) and 2-vs-many (single and double mutants in training set, three plus in test set).

A.3 Conditioning set ensembles

We exploit the ability of ProGen to score sequences either forwards (i.e. left-to-right) or in reverse. We generate ensembles of scoring functions differing in which residues are scored in which direction. For each ensemble member, we sample a random residue index b on which to split each protein into two segments. The first segment is scored in reverse, conditioned on the second segment, and the second segment is scored forwards, conditioned on the first segment. The conditional likelihoods of the two segments are then added to form a pseudo-likelihood scoring function.

$$s_\theta(x) = \frac{1}{L}\left(\sum_{i=1}^{b} \log p(x_i|x_{>i}) + \sum_{i=b+1}^{L} \log p(x_i|x_{<i})\right),\ b \sim \text{Uniform}(10, L-10) \tag{3}$$

The k^{th} ensemble member is constructed by sampling b_k, and then fine-tuning using the resulting scoring function for all sequences. In this way, each ensemble member varies in the set of conditioning variables for each residue.

A.4 Hyperparameter details

Fitness Prediction We used the CNN architecture from [23]. CNN models were trained for 100 epochs with a learning rate of 1e-3 and a batch size of 128. Fine-tuning of ProGen2 models was performed for 25 epochs with a learning rate of 1e-5, and a batch size of 32. All 32×32 comparisons in a batch were used to evaluate the binary cross entropy loss. Where batches of size 32 did not fit in memory (due to longer input sequences), we used gradient accumulation to maintain an effective batch size of 32.

When reporting results we used end-of-epoch checkpoints with the best performance on a sets of 128 sequences sampled and removed from the test set, constituting an upper bound on the performance that could be achieved using a validation set. The same sets of sequences were used to determine the best epoch for all models.

Bayesian optimisation We use the same CNN architecture as before. Ensembles of 10 models were trained for 50 epochs each. We retrained the ensembles from new random initialisations each time the dataset was updated with newly acquired observations. Fitness landscape optimisation was run using three initial random seeds to control stochasticity in model initialisation and training and acquisition strategies. The random seeds do not affect the choice of training sequences. Plots show mean and standard deviations across runs.

A.5 Evaluating ranking ensembles

Given a set of parameters θ_i corresponding to the members of an ensemble, we evaluate the predictions by computing the average score for each sequence x: $s_{\text{ens}}(x) = \frac{1}{K}\sum_i s_{\theta_i}(x)$, and comparing these scores to the fitness values y via the Spearman correlation.

When computing negative log-likelihood on the test sets, we average the probabilities $p(y(x_1) > y(x_2)|x_1, x_2)$ rather than applying the sigmoid to the difference in average scores, as we found this worked slightly better.

A.6 Surrogate models in preferential Bayesian optimisation

We assume the existence of a dataset of binary-valued pairwise comparisons, representing the ordering of pairs of sequences under the function to be optimised. We then posit a latent function f, which is in turn assumed to generate the binary observations via a likelihood $p(x_1 \succ x_2|f(x_1), f(x_2))$. In case this likelihood is the Bradley-Terry model above, then given a set of observed pairwise comparisons $\mathcal{D}$, the predictive distribution for the comparison between points x_1 and x_2 is:

$$p(x_1 \succ x_2|\mathcal{D}, x_1, x_2) = \int p(f|\mathcal{D})\sigma(f(x_1) - f(x_2))df \tag{4}$$

Given a set of pairwise observations, a posterior distribution over latent functions is formed, and the uncertainty in the location of the optimum captured in this distribution is used to guide the acquisition of new points to evaluate, as in standard BO.

A.7 Thompson sampling

In the context of batch Bayesian optimisation, Thompson sampling (TS) provides a commonly used alternative to traditional acquisition functions [5]. Given a posterior distribution over functions $p(f|\mathcal{D})$ and a batch size q, Thompson sampling generates a batch of query points by drawing q

function realisations from the posterior, and returning the query points $x^{(i)}$ having the maximum values under each sampled function:

$$x^{(i)} = \text{argmax} f^{(i)}(x),\ f^{(i)} \sim p(f|\mathcal{D}),\ i \in \{1, ...q\} \tag{5}$$

Maximising different draws from the posterior automatically generates diversity within the query points in a batch, thereby offering a straightforward and readily parallelisable alternative to more complicated batch generalisations of standard acquisition functions. Thompson sampling is also naturally extended to the preferential setting by selecting as query points the maxima of functions drawn from a distribution over *latent* functions f [27].

In our setting, we replace sampling q realisations from the posterior with sampling (without replacement) q members from the ensemble. Thus the i^{th} query point in a batch is found by maximising the function represented by the i^{th} ensemble member:

$$x^{(i)} = \text{argmax} s_{\theta_i}(x), \tag{6}$$

Equation 6 thus requires optimisation of individual ensemble members. In practice, we optimise within a small mutational radius using the stochastic hill climbing strategy described in Section A.8.

A.8 Acquisition function optimisation

We perform acquisition function optimisation using a stochastic hill climbing optimisation strategy. We perform r hill climbing steps, where r is the size of the local neighbourhood over which the optimisation is being performed (for all experiments here, $r = 3$). In each step 100 random mutations to the current sequence are sampled, and the one having the highest acquisition score is retained and incorporated into the current sequence. To better approximate the true local maximum, for each sequence to be acquired we generate a set of candidates by repeating this process 10 times, and select the candidate having the highest acquisition score among these 10 sequences (similar to the common practice of using random restarts in acquisition function optimisation). When acquiring a batch of acquisitions we explicitly prohibit duplicates by preventing mutations that would lead to duplicate sequences from being sampled during the hill climbing procedure.

A.9 Alternative preferential acquisition functions

Much of the literature on preferential BO has been concerned with proposing and evaluating modifications to standard acquisition functions suitable for use with preference-based surrogates. Here we consider as representative examples preferential extensions of the standard improvement-based acquisition functions, expected improvement (EI), and probability of improvement (PI). The crucial difference is that the absence of an explicit model of the observed function values means that improvement must instead be defined with reference to the latent function values modelled by the preferential surrogate. Thus, to generalise expected improvement, we compute improvement relative to the *predicted latent function value* at the point x^* having the best underlying value $y(x^*)$ to date:

$$I(x, x^*) = \max(0, f(x) - f(x^*)) \,, \tag{7}$$

where f is the latent function introduced in Appendix A.6. The expected improvement is then simply the expectation under the posterior over latent functions:

$$\text{EI}(x, \mathcal{D}) = \int I(x, x^*|f) p(f|\mathcal{D}) df \,. \tag{8}$$

To define probability of improvement, we exploit the fact that Equation 4 directly defines the required probability by substituting x^* for x_2:

$$\text{PI}(x, \mathcal{D}) := p(y(x) > y(x^*)|\mathcal{D}) = \int p(f|\mathcal{D}) \sigma(f(x) - f(x^*)) df \,. \tag{9}$$

To implement these acquisition functions with our ranking ensembles, we replace expectations under the posterior with averages under the ensemble. The acquisition functions are then evaluated as:

$$\mathrm{EI}(x, \mathcal{D}) = \frac{1}{K} \sum_{i=1}^{K} \max(0, s_{\theta_i}(x) - s_{\theta_i}(x^*)) \tag{10}$$

$$\mathrm{PI}(x, \mathcal{D}) = \frac{1}{K} \sum_{i=1}^{K} \sigma(s_{\theta_i}(x) - s_{\theta_i}(x^*)) \tag{11}$$

$$\tag{12}$$

We compare these acquisition functions to Thompson sampling and greedy batch construction on the AAV landscape optimisation task introduced in Section 4. To construct batches of acquisitions using the preferential acquisition functions, we perform stochastic hill climbing on the acquisition function values as described in Appendix A.8, explicitly preventing duplicate acquisitions within batches.

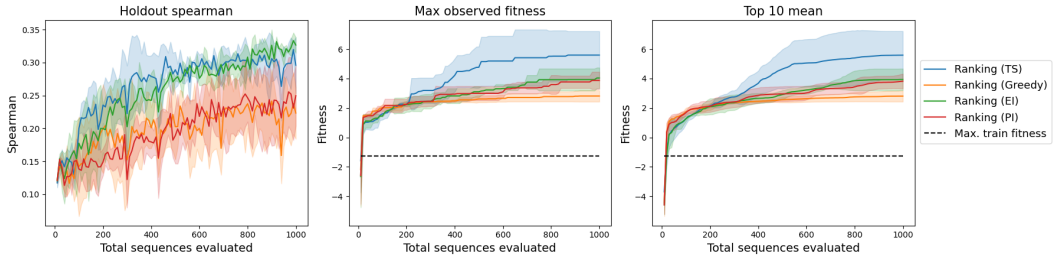

Figure 2: Alternative preferential BO batch acquisition strategies compared to Thompson sampling and greedy batch construction on the AAV fitness optimisation task.

Figure 3 shows that optimisation performance when using preferential EI or preferential PI is roughly equivalent. Though these acquisition strategies lead to better results than the purely exploitative greedy strategy, Thompson sampling remains the most effective strategy. Whereas Thompson sampling naturally encourages diversity in the members of a batch, constructing a batch by (approximately) finding the q sequences which individually have the highest pointwise EI or PI may lead to similar sets of acquisitions. To mitigate this problem, it is possible to derive extensions to EI or PI that explicitly estimate the overall improvement after acquiring a batch of sequences: indeed one such extension adapted to the preferential setting was previously studied by Siivola et al. [27]. However, these batch acquisition functions can be challenging both to compute and to optimise, especially when not using Gaussian process surrogates, and we leave further exploration of them to future work.

A.10 Effect of training dataset

The optimisation results in Section 4 were obtained using a randomly sampled set of known low-scoring sequences as a training set, thereby testing the extent to which optimisation methods are capable of improving initially unpromising sequences. As a somewhat more realistic scenario, we also tested optimisers when training on a set of point mutants of the wild-type sequence. We randomly sampled 96 single mutants for this purpose. Results are shown in Figure 3, and show similar relative performances, with optimisers using ranking-based surrogates again outperforming optimisers using regression-based surrogates, and the preferential BO configuration (ranking surrogate with Thompson sampling batch acquisition) clearly performing the best.

We also assessed the impact of re-scoring sequences in the initial training dataset with the oracle rather than using the experimental fitness values directly. Figure 4 shows that the relative performance of the ranking-based method and the regression-based method is not significantly affected by this change, which reflects the fact that the oracle was trained on the experimental values, and therefore the oracle predictions show very high correlation with the true fitness values.

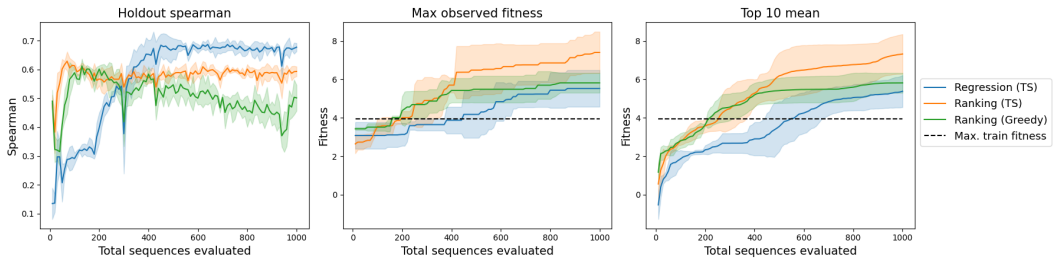

Figure 3: Comparison of BO strategies for optimising AAV fitness oracle, given a random set of point mutations of the wild-type as a training dataset

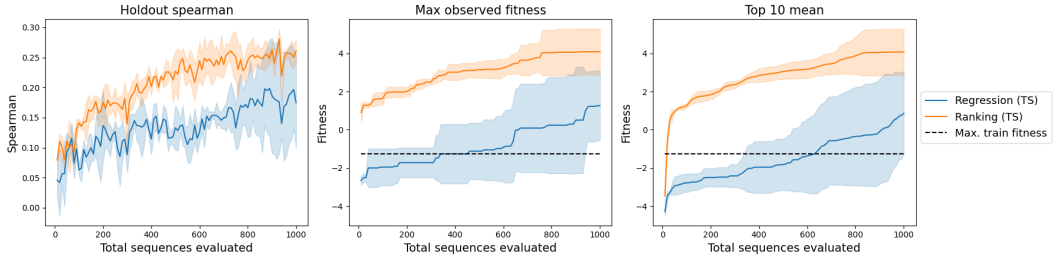

Figure 4: Re-scoring the initial dataset using the oracle does not significantly impact relative performance.