ARTICLES
https://doi.org/10.1038/s42256-022-00532-1
nature machine intelligence

Transformer-based protein generation with regularized latent space optimization

Egbert Castro[1], Abhinav Godavarthi[2], Julian Rubinfien[3], Kevin Givechian[4], Dhananjay Bhaskar[4] and Smita Krishnaswamy[1,2,4,5] ✉

The development of powerful natural language models has improved the ability to learn meaningful representations of protein sequences. In addition, advances in high-throughput mutagenesis, directed evolution and next-generation sequencing have allowed for the accumulation of large amounts of labelled fitness data. Leveraging these two trends, we introduce Regularized Latent Space Optimization (ReLSO), a deep transformer-based autoencoder, which features a highly structured latent space that is trained to jointly generate sequences as well as predict fitness. Through regularized prediction heads, ReLSO introduces a powerful protein sequence encoder and a novel approach for efficient fitness landscape traversal. Using ReLSO, we explicitly model the sequence–function landscape of large labelled datasets and generate new molecules by optimizing within the latent space using gradient-based methods. We evaluate this approach on several publicly available protein datasets, including variant sets of anti-ranibizumab and green fluorescent protein. We observe a greater sequence optimization efficiency (increase in fitness per optimization step) using ReLSO compared with other approaches, where ReLSO more robustly generates high-fitness sequences. Furthermore, the attention-based relationships learned by the jointly trained ReLSO models provide a potential avenue towards sequence-level fitness attribution information.

The primary challenge in sequence-based protein design is the vast space of possible sequences. A small protein of 30 residues (mean length in eukaryotes ≈ 472, ref. [1]) translates into a total search space of 10^{38}—far beyond the reach of modern high-throughput screening technologies. This obstacle is further exacerbated by epistasis (higher-order interactions between amino acids at distant residues in the sequence), which makes it difficult to predict the effect of small changes in the sequence on its properties[2]. Together, this motivates the need for approaches that can better leverage sequence–function relationships, often described using fitness landscapes[3], to more efficiently generate protein sequences with desired properties. To address this problem, we propose a data-driven deep generative approach called Regularized Latent Space Optimization (ReLSO). ReLSO leverages the greater abundance of labelled data arising from recent improvements in library generation and phenotypic screening technologies to learn a highly structured latent space of joint sequence and structure information. Further, we introduce novel regularizations to the latent space in ReLSO such that molecules can be optimized and redesigned directly in the latent space using gradient ascent on the fitness function.

Although the fitness of a protein (we use this term in general to refer to some quantifiable level of functionality that an amino-acid sequence possesses: for example, binding affinity, fluorescence, catalysis and stability) is more directly a consequence of its folded, three-dimensional structure rather than strictly its amino-acid sequence, it is often preferable to connect fitness directly to sequence since structural information may not always be available. Indeed, when generating a library of variants for therapeutic discovery or synthetic biology, either through a designed, combinatorial approach or by random mutagenesis, it is cost prohibitive to solve for the structure of each of the typically 10^3–10^9 variants produced.

Here we observe that protein design is fundamentally a search problem in a complex and vast space of amino-acid sequences. For most biologically relevant proteins, sequence length can range from a few tens to several thousands of residues[1]. Since each position of an N-length sequence may contain one of 20 possible amino acids, the resulting combinatorial space ($\approx 20^N$ sequences) is often too large to search exhaustively. Notably, this problem arises with the consideration of just canonical amino acids, notwithstanding the growing number of non-canonical alternatives[4]. A major consequence of the scale of this search space is that most publicly available datasets, although high throughout in their scale, capture only a small fraction of possible sequence space and thus the vast majority of possible variants are left unexplored.

To navigate the sequence space, an iterative search procedure called directed evolution[5] is often applied, where batches of randomized sequences are generated and screened for a function or property of interest. The best sequences are then carried over to the next round of library generation and selection. Effectively, this searches sequence space using a hill-climbing approach, and as a consequence is susceptible to local maxima that may obscure the discovery of better sequences. Other approaches to protein design include structure-based design[6,7], where ideal structures are chosen a priori and the task is to fit a sequence to the design. Recently, several promising approaches have emerged incorporating deep learning into the design[8,9], search[10,11] and optimization[12] of proteins. However, these methods are typically used for in silico screening by training a model to predict fitness scores directly from the input amino-acid sequences. Recent approaches have also utilized reinforcement learning to optimize sequences[13]. Although these approaches are valuable for reducing the experimental screening burden by proposing promising sequences, the challenge of navigating the sequence space remains unaddressed.

[1]Computational Biology and Bioinformatics Program, Yale University, New Haven, CT, USA. [2]Department of Applied Mathematics, Yale University, New Haven, CT, USA. [3]Department of Mathematics, Yale University, New Haven, CT, USA. [4]Department of Genetics, Yale University, New Haven, CT, USA. [5]Department of Computer Science, Yale University, New Haven, CT, USA. ✉e-mail: smita.krishnaswamy@yale.edu

An alternative to working in the sequence space is to learn a low-dimensional, semantically rich representation of peptides and proteins. These latent representations collectively form the latent space, which is easier to navigate. With this approach, a therapeutic candidate can be optimized using its latent representation, in a procedure called latent space optimization.

Here we propose ReLSO, a deep transformer-based approach to protein design, which combines the powerful encoding ability of a transformer model with a bottleneck that produces information-rich, low-dimensional latent representations. The latent space in ReLSO, besides being low dimensional, is regularized to be (1) smooth with respect to structure and fitness by way of fitness prediction from the latent space, (2) continuous and interpolatable between training data points and (3) pseudoconvex on the basis of negative sampling outside the data. This highly designed latent space enables optimization directly in latent space using gradient ascent on the fitness and converges to an optimum that can then be decoded back into the sequence space.

Key contributions of ReLSO include the following.

- The novel use of a transformer-based encoder with an autoencoder-type bottleneck for rich and interpretable encodings of protein sequences.
- A latent space that is organized by sequence–function relationships, which ameliorates difficulties in optimization due to combinatorial explosion.
- A convex latent space that is reshaped using norm-based negative sampling to induce a natural boundary and stopping criterion for gradient-based optimization.
- An interpolation-based regularization that enforces gradual changes in decoded sequence space when traversing though latent space. This allows for a more dense sampling of the underlying sequence manifold on which the training data lies.
- A gradient-ascent algorithm for generating new sequences from the latent space.

We evaluate ReLSO on several publicly available protein datasets, including variant sets of anti-ranibizumab and green fluorescent protein (GFP). We view this domain first through a protein representation learning perspective, where we compare popular representations of proteins. We observe that ReLSO learns a more organized, smoother representation relative to other approaches. Next we examine the optimization ability of ReLSO on several protein design tasks. Compared with other sequence-based optimization approaches, ReLSO shows greater optimization efficiency (increase in fitness per optimization step) using its fitness-directed traversal of latent space. This optimization efficiency allows ReLSO to more robustly generate high-fitness sequences. Finally, the attention-based relationships learned by the jointly trained ReLSO models provide a potential avenue towards sequence-level fitness attribution information.

Results

The ReLSO architecture and regularizations. The ReLSO architecture is designed to jointly generate protein sequences as well as predict fitness from latent representations. The model is trained using a multitask loss formulation, which organizes the latent space by structure (input sequence) and function simultaneously, thus simplifying the task of finding sequences of high fitness from a search problem in a high-dimensional, discrete space to a much more amenable optimization problem in a low-dimensional, continuous space. ReLSO leverages a transformer encoder to learn the mapping of sequences to latent space and utilizes gradient-based optimization to systematically and efficiently move through latent space towards regions of high fitness. A norm-based negative sampling penalty is used to reshape the latent fitness landscape to be pseudoconvex. This has the dual benefit of further easing the optimization challenge as well as creating an implicit trust boundary. ReLSO makes innovative use of an interpolation regularization that enforces smoothness with respect to sequence, whereby small perturbations to a latent representation correspond to minor changes in the reconstructed sequence. This is especially relevant to protein design, as it allows for a dense sampling of the latent space for diverse protein sequence generation while retaining properties of interest.

Transformer-based encoder with dimensionality reduction. ReLSO employs a transformer-based encoder to learn the mapping from a sequence, x, to its latent representation, z (Fig. 1a). While other encoding methods that rely on convolutional and recurrent architectures have demonstrated success in this domain[14,15], we chose to use a transformer for several key reasons. First, the inductive bias of the transformer architecture matches the prevailing understanding that protein function is a consequence of pairwise interactions between residues (for example, catalytic triads of proteases). Indeed the transformer architecture has shown promise in several tasks relying on protein sequence for prediction[16–19]. Second, the transformer's attention-based encoding scheme provides for interpretability by analysis of learned attention weights[18,20]. Finally, transformers have demonstrated advantages in representing long sequences, as a consequence of viewing the entire sequence during the forward pass, thus avoiding the limitations inherent to encoders based on recurrent neural networks[21]. Finally, we refer readers to ref. [22] for a more in-depth discussion of protein sequence representation learning approaches.

The encoder network in ReLSO, f_θ, transforms input proteins to a token-level representation where each amino acid in the sequence is replaced by a positional encoding of fixed length. This representation is then compressed to a coarse, sequence-level, representation using an attention-based pooling mechanism, which computes the convex sum of positional encodings. This approach is preferred over mean or max pooling since it is able to weight the importance of positions in the sequence without incurring the computational cost of more complex strategies using recurrent-based pooling. In contrast to other transformer encoders, in ReLSO the dimensionality of this sequence-level representation is further reduced using a fully connected network (Fig. 1a). This amounts to passing the sequence information through an information bottleneck, resulting in an informative and low-dimensional z. Low dimensionality of the latent representation is important, since latent space grows exponentially with increasing dimension, making latent space optimization more challenging.

Jointly trained autoencoder (JT-AE). ReLSO incorporates two vital factors in protein design: (1) sequence and (2) fitness information. By jointly training an autoencoder with a prediction network, the original autoencoder architecture, comprised of an encoder f_θ and decoder g_θ, is supplemented with a network h_θ, which is tasked with predicting fitness from z. Here, x is an input sequence and y is the corresponding fitness measurement. The resulting objective function of this set-up takes the form

$$\mathcal{L} = ||g_\theta(f_\theta(x)) - x|| + ||h_\theta(f_\theta(x)) - y||,$$

which includes the reconstruction loss and the fitness prediction (regression) loss. At each backpropagation step, the encoder is updated with gradients from both losses and is therefore directed to encode information about sequence and fitness in z. Indeed, when the dimension of z is set to some low value, $d \ll N$, the encoder is forced to include only the most salient information about sequence and fitness and induces a connection between the two in z. This property was exploited in the biomolecular domain[23], where a

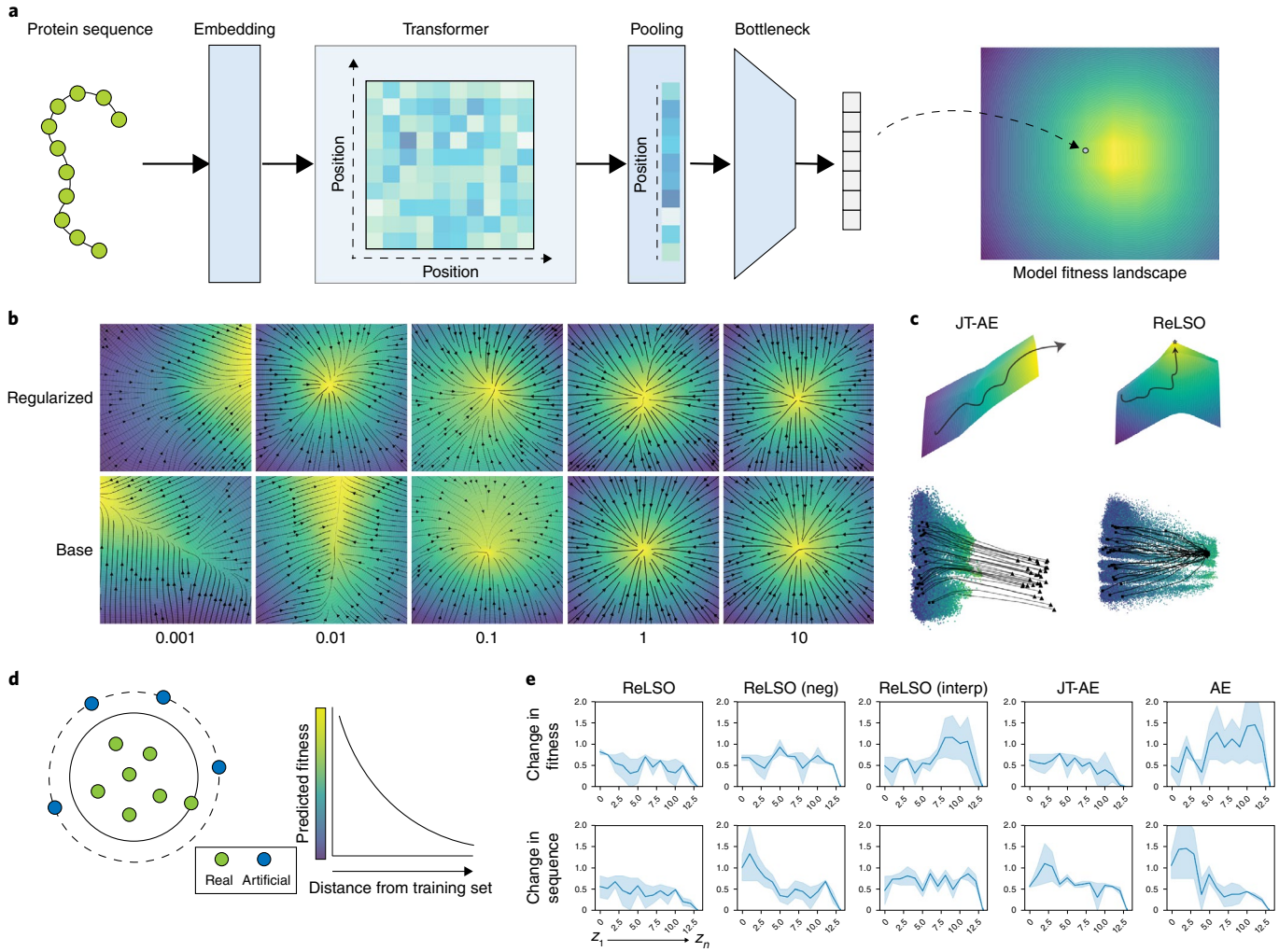

Fig. 1 | ReLSO maps sequences to a regularized model fitness landscape. a, To encode protein sequences, ReLSO uses a transformer-based encoder. The output of the transformer module is pooled using an attention-based pooling mechanism and then compressed further to produce the latent representation of an input sequence. The collection of latent points forms a model fitness landscape. **b**, ReLSO uses an auxiliary network to predict fitness from latent space and uses a norm-based negative sampling technique to enforce a pseudoconcave shape in the resulting latent space. The choice of the auxiliary network affects how sensitive the network is to the negative sampling loss. The plots in the top row were generated using a network penalized to learn smoother functions whereas the plots in the bottom row were produced by a conventional fully connected network. **c**, An inherent weakness of a naive joint-training approach, as in JT-AE, is that often a monotonic function is learned by the auxiliary network. Such a function lacks any stopping criterion when used for latent space optimization. To address this, we reshape the fitness function such that the global maxima lie in/near the training data. **d**, Negative sampling relies on a data augmentation strategy wherein artificial low-fitness points are generated on the periphery of latent space. **e**, We monitor how sequence and fitness change during latent space traversal using 100 sampled walks between pairs of distant points in latent space. The *x* axis denotes the step index along the path taken. The *y* axis displays the difference in the denoted property between the current step, z_i, and the final step, z_n, such that the difference becomes zero at the final step when $z_i = z_n$. The mean (line) and 95th percentile confidence interval (shaded region) are shown.

jointly trained variational autoencoder generated latent encodings organized by chemical properties. Here we leverage the same strategy to establish a strong correspondence between the protein sequence and its fitness, which is later utilized for generating novel sequences with desired fitness. Note that we refer to the model architecture trained with the reconstruction and fitness prediction losses as JT-AE. ReLSO refers to the complete model, which includes negative sampling and interpolation losses, described next.

Negative sampling for pseudoconvexity of latent space. A fundamental challenge in performing optimization in the latent space is that the optimization trajectory can stray far from the training data into regions where the prediction accuracy of the model deteriorates, producing untrustworthy results. Recent work has proposed techniques to define boundaries for model-based optimization by imposing constraints such as a sequence mutation radius[11] or by relying on model-derived likelihood values[9]. While mutation radii are straightforward to implement, the pronounced variability of fitness levels, even within the immediate mutational neighbourhood of a protein sequence, makes such global thresholds less than ideal. Additionally, a mutational radius constraint can be oversimplified, as high mutation count may potentiate functional benefit to the fitness of a protein (for example, the SARS-CoV-2 B.1.1.529 (Omicron) spike protein).

The fitness prediction head of the JT-AE provides directional information for latent space optimization. However, it does not impose any stopping criterion or any strong notion of a boundary or fitness optima. Furthermore, the addition of an auxiliary

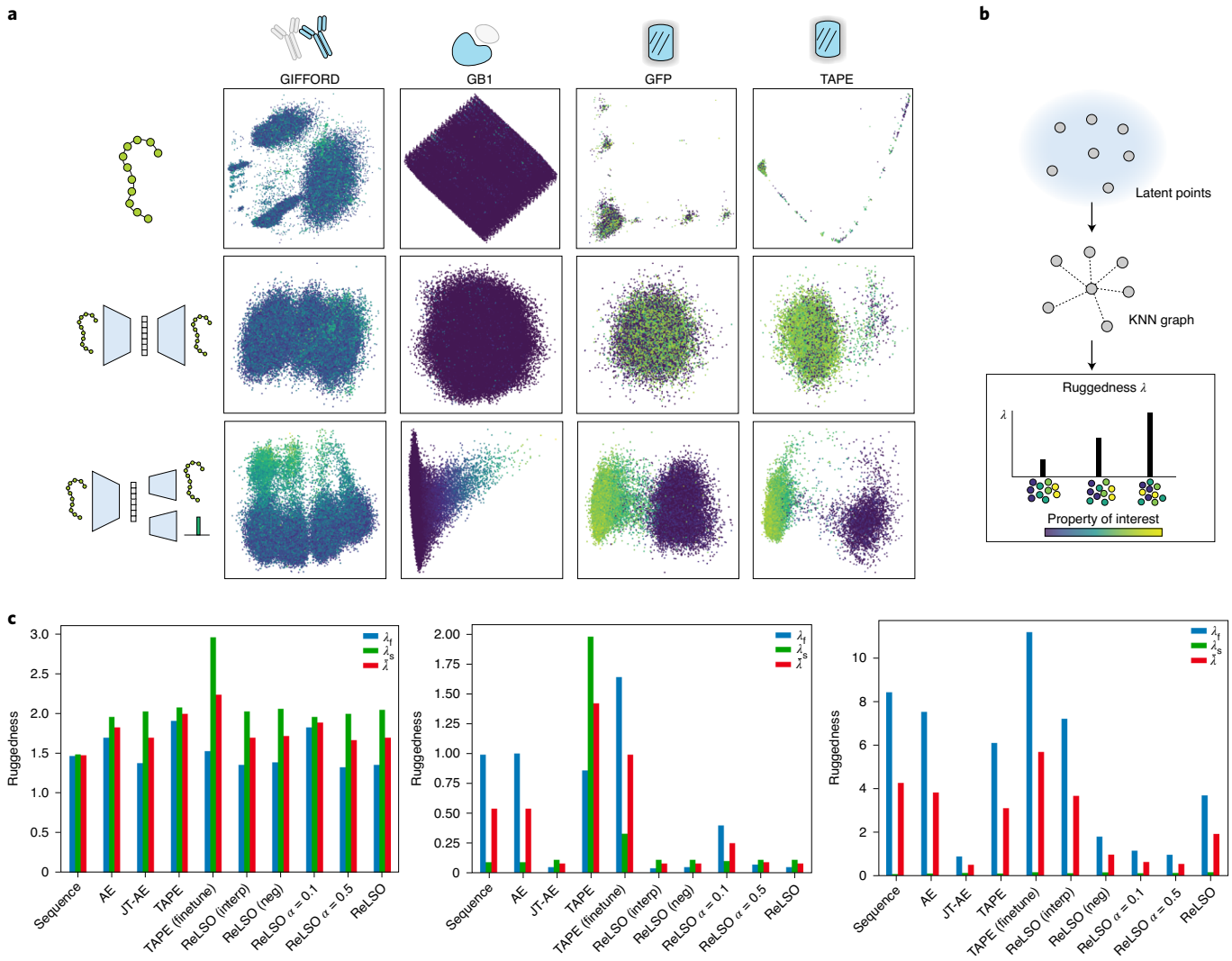

Fig. 2 | ReLSO learns smooth representations of protein sequences. a, We compare two main representation methods for protein sequences, namely amino-acid sequence, latent embeddings from an unsupervised autoencoder, and latent embeddings produced by JT-AE. The three protein sequence representation approaches are visualized using principal component analysis and each point is coloured according to its corresponding fitness value. Columns of the plot grid correspond to datasets used in this study. Across the four datasets, joint training produces a latent space organized by fitness and sequence information. **b**, Quantification of ruggedness is performed using a metric that measures neighbourhood-level variation of a property in latent space. **c**, We compare the ruggedness of representations with respect to differences in fitness, λ_f, and differences in sequence, λ_s.

attribute prediction task (for example, fitness prediction) to an autoencoder often results in unidirectional organization of the latent space by that attribute[23]. In such cases (Fig. 1d), following the gradient produces an optimization trajectory that extends far outside the training manifold with no natural stopping point. This may ultimately result in generated protein sequences that are 'unrealistic' with respect to other biochemical properties necessary for proper folding, such as stretches of homopolymers or amino-acid sequence motifs known to be deleterious to stability/solubility in solution.

To fully leverage the gradient signal provided by the fitness prediction head, h_θ, we introduce a bias in learned fitness function, $\hat{\phi}_z$, towards regions in the latent space near the training data. This is done using a data augmentation technique called norm-based negative sampling. Each z obtained from the training data is complemented with a set of negative samples, z_n. These negative examples are produced by sampling high-norm regions of the latent space surrounding real latent points (Fig. 1d). By assigning these artificial points, z_n, with low fitness and including them in the fitness prediction loss, $\hat{\phi}_z$ is reshaped in such a manner that there is a single fitness maximum located in or near the training data manifold. Using this regularization, an implicit trust region forms, thus providing a natural stopping criterion for latent space optimization. To better examine this phenomenon, we train ReLSO using a latent encoding dimension of two, $z \in \mathbb{R}^2$, and visualize the resulting, learned fitness function using h_θ. We observe that the negative sampling regularization induces a pseudoconcave shape wherein high-fitness latent points reside at the centre of this two-dimensional latent space (Fig. 1b). Increasing the strength of this regularization using a hyperparameter η further enforces this organization. We will refer to the JT-AE model augmented with this regularization as ReLSO (neg).

Interpolative sampling penalty for latent space continuity. To further improve the latent space of our jointly trained autoencoder for protein sequence optimization, we also introduce a penalty, which enforces smoother interpolation in latent space with respect to

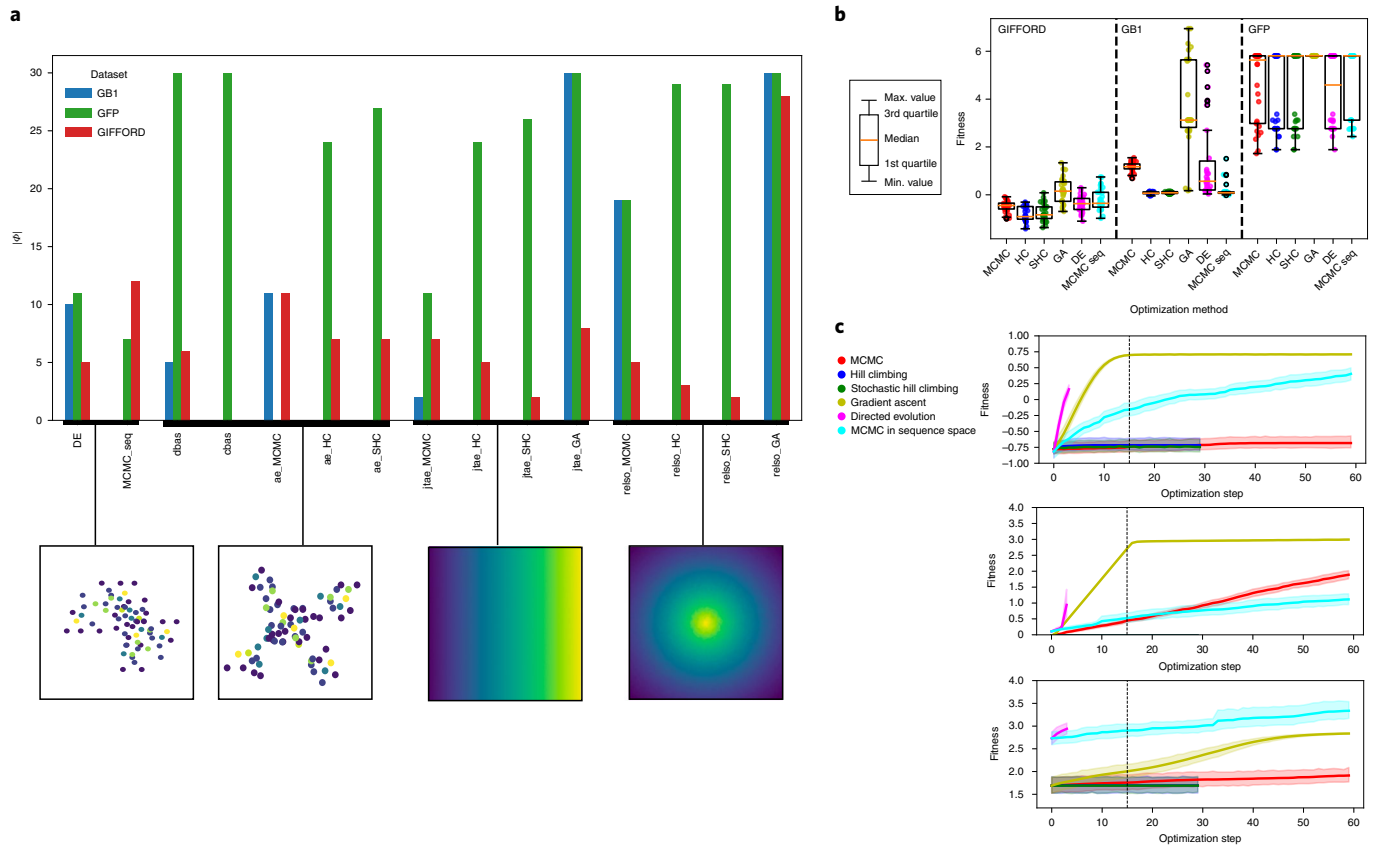

Fig. 3 | Comparison of methods for ML-based protein sequence optimization efficiency. a, For each method, a batch of low-fitness sequences, sampled from the held-out set, was chosen as a starting point. Ending optimized sequences that are predicted to be high fitness, determined by a threshold, are included in Φ. The optimization algorithms operate within a search space that is visualized below the plots. Left to right: sequence, AE, JT-AE and ReLSO. **b**, Ending fitness values of all 30 seeds are plotted, split by dataset. **c**, The trajectories of fitness values over optimization step display a wide spread of efficiency. As some methods label multiple sequences during a single optimization step (for example, hill climbing, directed evolution), some lines do not reach across the entire *x* axis. To highlight gradient ascent's quick convergence to high fitness, a vertical line is shown at the 15th optimization step in each plot.

sequence modifications. This is appealing for sequence-based protein design, as we would like to be able more densely sample latent space for both analysis of latent space optimization trajectories and enrichment of the areas of sequence space around high-fitness sequences.

We enforce gradual changes in the decoded sequence space during latent space traversal by the addition of an interpolation regularization term. In this term, a subset of the batch of latent points is used to compute a k-nearest-neighbour (KNN) graph using pairwise Euclidean distances. A set of new latent points z_p is then generated by interpolating between nearest neighbours. This new set of points z_p is passed through g_θ to produce a set of decoded sequences $\hat{x}_p$. We then penalize by the distance between two sequences in $\hat{x}$ and their interpolant $\hat{x}_p$. Formally, this penalty calculated element-wise by

$$\mathcal{L}_{\text{interp}} = \max(0, \frac{||\hat{x}_1 - \hat{x}_i|| + ||\hat{x}_2 - \hat{x}_i||}{2} - ||\hat{x}_1 - \hat{x}_2||)$$

where $\hat{x}_1$ and $\hat{x}_2$ are nearest neighbours in latent space and $\hat{x}_i$ is the decoded sequence of the interpolated latent point. We will refer to the JT-AE model augmented with only this regularization as ReLSO (interp). Finally the full model, with both negative sampling and interpolative sampling regularization, is referred to as ReLSO.

Latent space optimization and sequence generation using ReLSO. We leverage the highly structured latent space of ReLSO to perform protein sequence optimization on several publicly available datasets, with additional information on each in Supplementary Table 1. First, the latent space maintains a global organization not only to fitness (Fig. 2a) but also to sequence information (Fig. 2c). Next, the negative sampling and interpolative sampling regularizations induce a latent space with several properties that ease the protein sequence optimization task such as a pseudoconcave fitness function. Finally, traversal in the latent space of ReLSO results in gradual changes in both sequence and fitness (Fig. 1e). We quantify this smoothness through the use of a graph-based metric (Fig. 2b) derived from the graph signal processing literature and previously used in the biomolecular domain to study thermodynamic landscapes[24]. Combined, these properties form a search space amenable to optimization approaches.

To optimize protein sequences, we use gradient ascent, which allows for systematic and efficient modulation of fitness. First, a sequence, x, is encoded by f_θ to produce a latent encoding z. This process maps an input protein sequence to its point in the model's latent fitness landscape. Next, the gradient with respect to the predicted fitness for the latent point, $\nabla_z h_\theta$, is calculated. The determined gradient provides directional information towards the latent fitness maxima and is used to update the latent point.

$$z^{(t+1)} \leftarrow z^{(t)} + \epsilon \nabla_z h_\theta.$$

This iterative process requires two hyperparameters, step size, ϵ, and number of steps, K. At termination of the optimization loop, a final latent point z_f is produced. This point in latent space is then

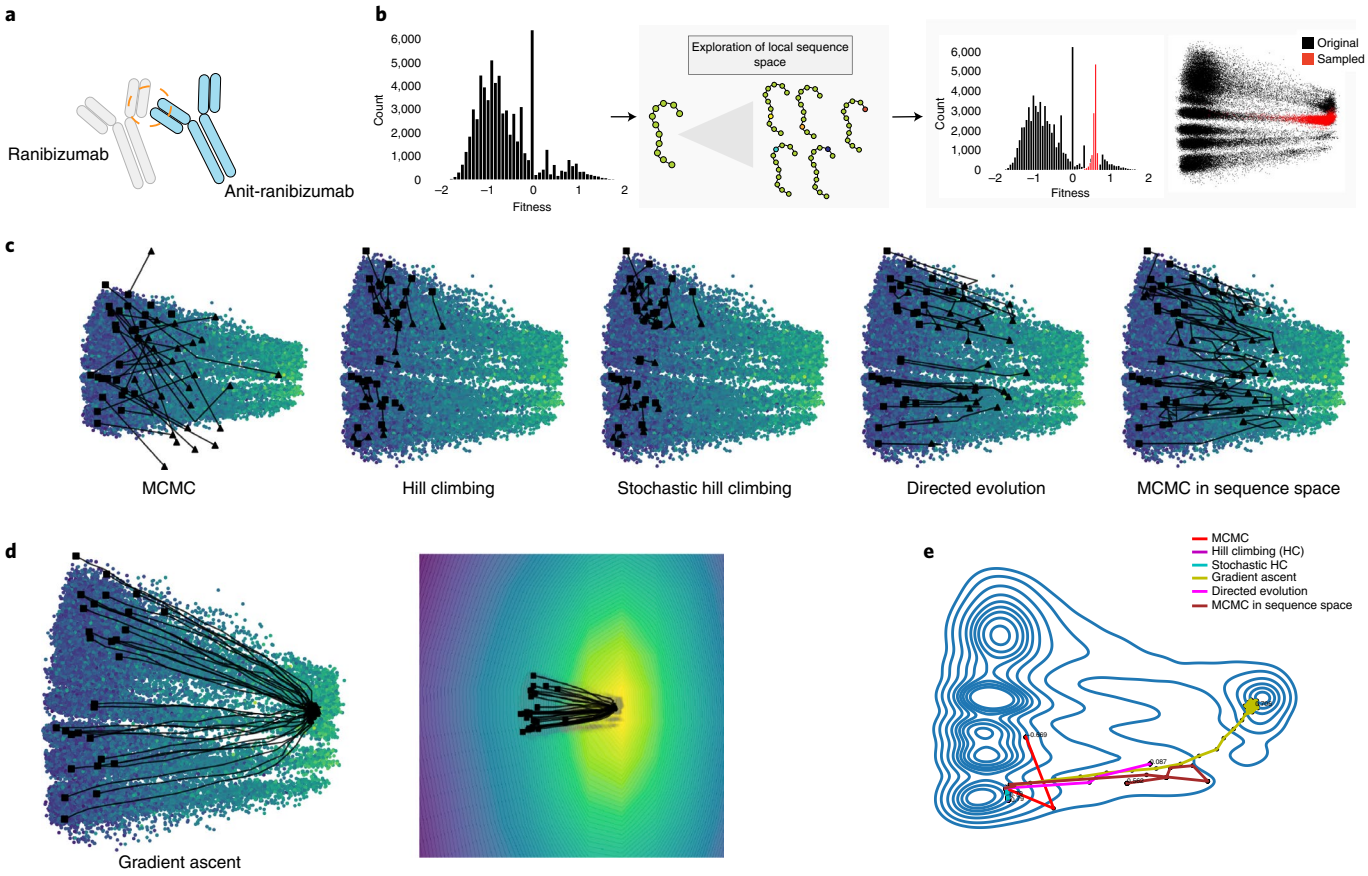

Fig. 4 | Protein sequence optimization of anti-ranibizumab antibodies. a, This task is focused on optimizing the highly variable CDR3 region of the anti-ranibizumab antibody. **b**, The tight coupling between Euclidean distance in latent space, sequence distance and fitness difference is examined. A set of sequences is generated from the local mutation space around a small set of high-fitness seed sequences present in the GIFFORD training set. These newly generated sequences are then encoded into the latent space of ReLSO and fitness values are predicted by the model. **c**, Visualization of optimization paths taken by gradient-free methods compared with those in this study. Each trajectory starts at a low-fitness sequence sampled from the held-out test set. **d**, Optimization paths taken by gradient ascent converge to a high-fitness region of latent space. This is a result of the underlying regularized fitness function learned by ReLSO. **e**, For the optimization of a single seed, paths taken by gradient-free methods show a less efficient progression to the data-driven global fitness maxima.

decoded to its corresponding sequence x_f using g_θ. The reliance on a gradient signal over other, more stochastic approaches allows for a more robust approach to sequence optimization that is less sensitive to starting points. If greater sequence diversity is desired, we found that injecting noise into the update step can increase the variation in sequences produced (not shown). Overall, this process is referred to as latent space optimization, whereby protein sequences are optimized in the latent space of a model rather than directly. A major benefit of using such an approach is the ability to learn a search space that ameliorates some of the challenges of optimizing protein sequences directly. In this work we learn a better search space for protein sequence optimization by heavily regularizing an autoencoder-like model such that the latent space maintains favourable properties while still being generative.

Comparison with other protein sequence optimization strategies. In recent years, many protein sequence optimization methods have emerged that rely on the use of a deep learning model. Some of these approaches use the model to perform an in silico screen of candidate sequences produced by iterative[10] or random[11] search. We view these methods as sequence-based search strategies, as the generation of new sequence candidates occurs at the sequence level. In contrast, methods such as refs. [8,9] generate new sequences by sampling the latent space of a generative model. In this study, we seek to leverage the gradient information present in h_θ to search for more fit protein sequences. We observed that, as a result of training to predict fitness from latent representation, the latent space became organized by fitness, as shown in Fig. 2a. To avoid several pathologies of optimization by gradient ascent in latent space, the regularizations included in ReLSO reshape this organization into a pseudoconcave shape (Fig. 1d), which eases the optimization challenge substantially.

As optimized sequences may possess hidden defects that present themselves in downstream analysis (for example, immunogenicity of antibodies), it is often desired to produce several promising candidates at the end of the optimization stage. We replicate this scenario by collecting high-fitness sequences in a set Φ whereby inclusion is restricted to sequences that are predicted to have a fitness value above some threshold. We evaluate the considered optimization methods by the cardinality of each method's Φ (Fig. 3a) and the ending fitness values (Fig. 3b). Additional results on the composition of sequences found in each method's Φ are included in Supplementary Table 2, where we report a set of metrics that describe the quality and quantity of optimized sequences generated. For each optimization approach, we began with a set of 30 seed sequences drawn from the bottom 25th percentile of each dataset's

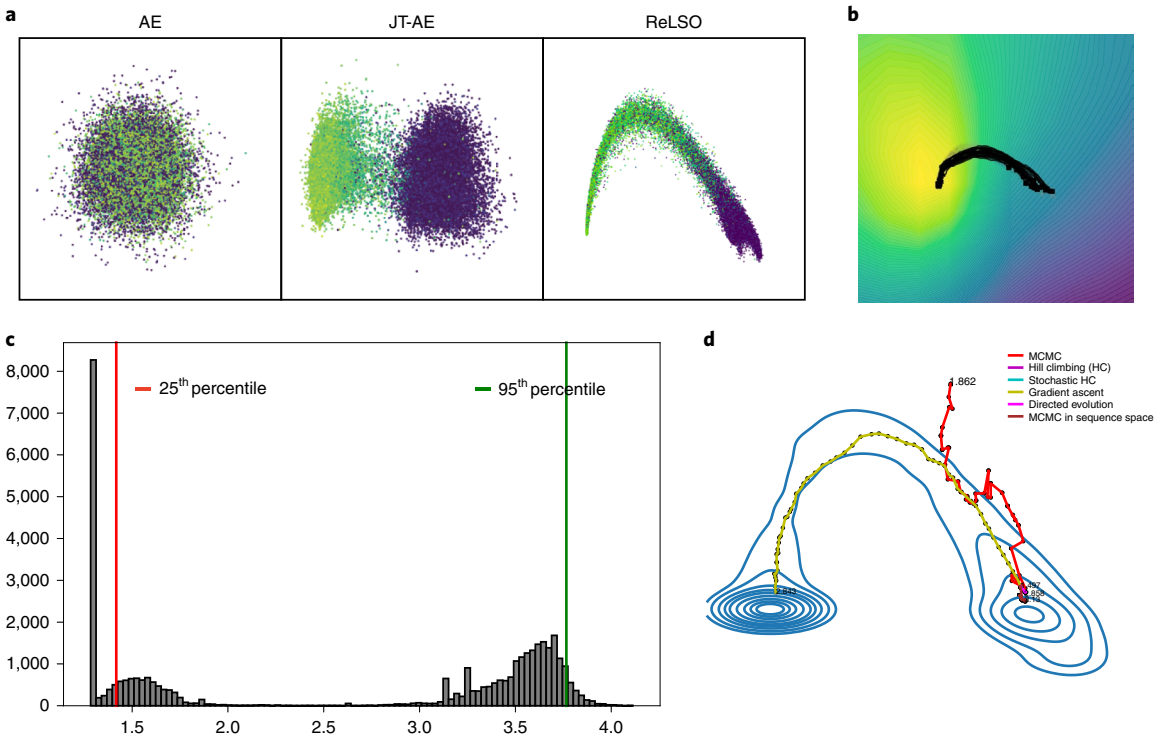

Fig. 5 | Optimization of fluorescence brightness of GFP. a, Visualization of the latent embeddings produced by AE, JT-AE and ReLSO models on the GFP dataset, coloured according to log fluorescence. **b**, The learned fitness function extracted from the auxiliary fitness prediction network of ReLSO. **c**, Bimodal distribution of log fluorescence values with a low- and a high-fluorescence mode. **d**, Optimization of a single, low-fluorescence seed sequence using gradient-free and gradient-based methods.

held-out test split. Next, each optimization approach is allowed an in silico evaluation budget of 60 evaluations, which corresponds to the number of sequences the method may observe and label. For methods such as directed evolution, which operate in a batchwise manner, the number of iterations is adjusted (Fig. 3c) to keep the total number of observed sequences constant across methods.

We find that ReLSO is able to produce a larger set of high-fitness sequences across the datasets with fewer optimization steps. This is observed in Fig. 3c, where gradient ascent is able to efficiently reach regions of high-fitness latent space and consequently generate high-fitness sequences with a fraction of the evaluation budget expended. With the narrow fitness landscape of GFP, we observe that most methods are able to reach high-fluorescence sequences; however, in GIFFORD and GB1, the performance gap between gradient ascent and other methods is more pronounced. Furthermore, in the case of GFP we observe that all optimized candidates of gradient ascent reside within the high-fluorescence mode of the dataset.

Optimization of anti-ranibizumab antibodies. A key high-throughput phage panning dataset used for antibody optimization and generation was obtained from ref. [15]. This dataset provided approximately 60,000 samples for sequence optimization of the third complementarity-determining region of the antibody heavy chain (CDR-H3) against the ranibizumab antibody. Success in this optimization task has implications in the context of modifying antibody binding to respective targets, thus further improving antibody–antigen binding, or generative antibody design with reduced antidrug antibody affinity. Together, this dataset motivated the task of localized manipulation of the CDR-H3 region of anti-ranibizumab sequences within the latent space of our model.

To subsequently explore the ability of the model to generalize to similar but non-identical sequences in the latent space, we generated 11,061 sequences using the top sequences within the GIFFORD dataset (enrichment binding affinity ≈ 1.5) (Fig. 4b). This filtered to 43 high-fitness 'seed' antibody sequences. Given the variability in physicochemical properties with respect to the potential of a single amino acid, we generated variants from each of these 43 sequences at one mutational amino-acid distance away from these 43 starting sequences. This allowed exploration of the local neighbourhood of the high-fitness latent space while also localizing to a particular window of the CDR3 domain.

Upon interpolation of the high-fitness sequences, each of 11,061 sequences was passed into the trained model for both binding affinity and embedding predictions. These embeddings and predicted fitness values were then examined (Fig. 4b). Interestingly, the generated sequences localized to a single high-fitness region of the latent space. Additionally, the generated sequence abundance appeared to be associated with a relatively high level of fitness as shown in Fig. 4b in line with the learned organization by fitness in latent space from the initial training data. These generative results suggest the potential implication of low mutational distances from regions of interest within the latent space of encoded sequences. Furthermore, latent space optimization followed by a more dense, local generative sampling of high-fitness variants may suggest an interesting avenue for sequence-space exploration and biotherapeutic optimization. Next, we examine the optimization trajectories taken by each optimization method across seed sequences. As previously described, each optimization begins with a low-fitness seed sequence and all methods have an equal in silico labelling budget (that is, calls to a fitness oracle). We observe that gradient-free methods, operating in either latent space or sequence space, all exhibit an inherent level of inefficiency in their search for higher fitness (Fig. 4c). For Markov chain Monte Carlo (MCMC) in latent space, this is compounded by the departure of optimization trajectories into sparsely populated

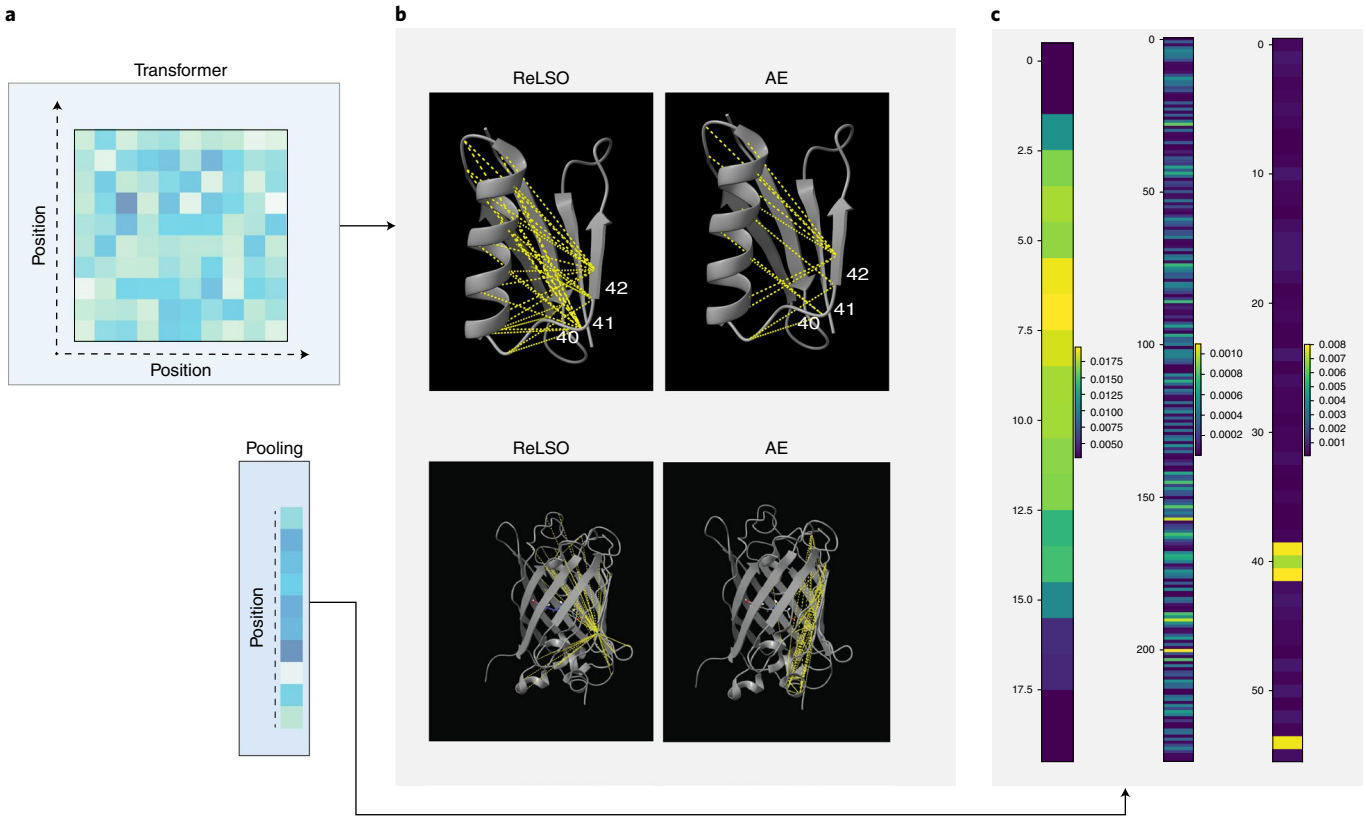

Fig. 6 | Leveraging attention relationships in ReLSO for fitness attribution. a, Attention weights can be extracted from the transformer module and pooling module in ReLSO's encoder. **b**, Here we visualize learned pairwise relationships defined by the attention maps of the transformer encoder. We compare the attention maps of the ReLSO model with those of the AE model. Top: GB1 protein structure [PDB:2J52] with overlaid attention. Bottom: GFP structure [PDB:2WUR] with overlaid attention. **c**, To generate a sequence-level representation of the protein sequence, an attention-based pooling step is used in ReLSO. The importance of each position as determined by this pooling mechanism is shown for GIFFORD, GFP and GB1.

regions of latent space. In contrast, the ability to leverage fitness gradients enables a more robust approach to sequence design whereby each optimization trajectory is able to reach high-fitness regions of latent space and be decoded to improved sequences (Fig. 4d).To highlight the difference in optimization efficiency between methods, we plot the optimization trajectories across several methods for a single seed in Fig. 4e.

GFP fitness landscape. In an effort to empirically describe epistasis in the fitness landscape of GFP, in ref. [25] random mutagenesis of the *Aequorea victoria* GFP was performed to produce 56,086 variants. The fluorescence of each variant was quantified, with multiple steps taken to ensure accurate estimates. The dataset produced from this study, which includes a mixture of variants with a mean number of 3.7 mutations per sequence, exhibited a narrow fitness landscape and a bimodal distribution of fluorescence values (Fig. 5c). The sequence–fitness relationship in the dataset presented an opportunity to evaluate the ReLSO model on a protein known to have epistatic relationships between residues and a challenging optimization landscape. We observe the ruggedness of the fitness landscape in the sequence representation and latent representation generated by the autoencoder (AE) model using principal component analysis (Fig. 2a,b). However, in the latent representations of JT-AE and ReLSO the bimodal distribution of fitness in the GFP fitness landscape is recapitulated (Fig. 5a). Investigation of the fitness function learned by the fitness prediction network of ReLSO, h_θ, reveals that the model has learned a smoothly changing function from low to high fluorescence and a global optimum in the high-fluorescence mode (Fig. 5b).

With the latent space organized by fitness, as demonstrated by both visualization of the latent coordinates and the learned fitness function, we conducted in silico sequence optimization with the same set-up as used on the GIFFORD dataset. First we sampled seed sequences from the held-out test set and selected sequences in the bottom quartile of observed fitness (log fluorescence ≤ 1.3), as visualized by the red vertical line in Fig. 5c. The threshold for high fitness and inclusion in Φ was set at the 95th percentile of log fluorescence (3.76) and is visualized with the green vertical line in Fig. 5c. Evaluation of optimization methods was then carried out and the results are presented in Fig. 3 and Supplementary Table 3. We observe a quick convergence to a high-fluorescence sequence by several methods, probably owing to the narrowness of the underlying fitness landscape[25]. However, knowledge of the fitness landscape still provides a reliable route for optimization, as not all seed sequences were able to be successfully optimized by methods other than gradient ascent (Fig. 5d). For sequences predicted to be high fluorescence by ReLSO, we observe a stabilizing change made to the sequence (mean ddG $= -1.3$ kcal mol^{-1}) as determined by Dynamut2[26]. ddG (or $\Delta\Delta$G, i.e. the change in the change in Gibbs free energy) measures the relative stability of a protein with respect to mutations in the amino acid sequence.

Interpretability. Encouraged by the success of refs. [18,20], we examined the attention weightings of the trained ReLSO model for possible localized sequence-fitness attribution. We hypothesized that, given the joint-training approach (Background and problem set-up) and the observed organization by fitness in the latent embeddings

(Fig. 2b), the learned attention information from the ReLSO model may also aid in elucidating fitness–sequence relationships. As the model displays a high level of performance on fitness prediction and sequence reconstruction (Supplementary Table 4), we posit that the model has captured salient and predictive aspects of protein sequences in its trained parameters. To examine this, we extract attention information from both the attention maps of the transformer layers (Fig. 6a) and the attention-based pooling operation. The former provides pairwise attention information while the latter provides positional attention information. To extract attention information, we randomly sample sequences from a dataset and pass them through the encoder and bottleneck modules of ReLSO. In each forward pass, the attention maps of the transformer model and the pooling weights of the bottleneck module are retained and aggregated. We then perform an averaging to reduce the level of noise. Additionally, a thresholding step is performed on the averaged attention maps to increase sparsity such that only the most salient relationships remain.

As previous approaches have focused on transformers trained in an unsupervised or self–supervised manner, we compare the attention information between AE and ReLSO (Fig. 6b). We observe several differences between the learned attention relationships of the ReLSO models and the AE models across the datasets. We view this divergence of attention relationships as a potential signal that the attention maps of transformers trained to predict fitness may also be probed to study sequence–fitness and structure–fitness relationships. Interestingly, the attention maps for GFP learned by ReLSO indicate the presence of potential epistatic interactions across a cylindrical pocket of the protein. Such long-range epistatic interactions have previously been identified[25] in GFP by experts in the field. Automated discovery of such interactions via the ReLSO architecture presents an exciting avenue for follow-up investigation. Next, we are able to observe that the variable residues in the GB1 dataset (39–41, 54) are highly weighted in the positional attention extracted from the ReLSO model, as shown in Fig. 6c. As these four positions are the only changing residues across sequences in the GB1 dataset, we confirm their importance for the prediction tasks with their attention maps as expected. In addition, the embeddings of the amino acids recapitulate their underlying biochemical properties in their organization (Supplementary Figs. 1 and 2). This result has been shared in previous work as a signal that the model has learned biologically meaningful information in its parameters[19].

Discussion

The ability to find better representations is vital to extracting insights from noisy, high-dimensional data within the fields of protein biology. Defined by their biochemical interactions, evolutionary selection pressures and function–stability tradeoffs, proteins are an increasingly important domain for the application of deep learning. More specifically, the field of biotherapeutic development has benefited considerably from the application of both linear and nonlinear models. Some of the very impactful models in this space have been largely supervised, but more recent work has proven the usefulness of leveraging unsupervised learning to pretrain predictive models to identify protein sequences with an enhanced property of interest.

Here we took an alternative path combining these two types of learning objective by instead taking a multitask learning approach. Through simultaneously optimizing for protein sequence generation and fitness level prediction, we explicitly enforce a latent space rich in information about both sequence and fitness information. Importantly, this fitness information may encompass a variety of different properties such as binding affinity and fluorescence, which are smoothly embedded in the latent space of our trained model. We then add regularizations that reflect principles of protein engineering, reshaping the latent space in the process. Leveraging these regularizations and the architecture of the model, we show how gradient-ascent optimization can deliver improvements in protein optimization when searching over protein sequence space.

The departure of this approach from other methods demonstrates a novel and promising avenue for improving our ability to design and optimize proteins. Furthermore, the reliance of this method solely on sequence information paired to a fitness value suggests that ReLSO-like architectures can be applied to other biomolecules such as DNA and RNA. In particular, one application to nucleic acids would be to optimize gene editing tools such as CRISPR–Cas9 to reduce off-target effects. Specifically, we see an interesting avenue in tuning binding affinity to increase selectivity towards a certain target or isoform, but against others to mitigate off-target toxicity. With the growing prominence of biological therapeutics, this research direction has potential to deliver improvements in the development of improved therapeutics.

Methods

Model architecture and training details. ReLSO can be understood as being comprised of four main modules (encoder module, bottleneck module, sequence prediction module and fitness prediction module). Encoder module: the encoder module takes as input protein sequences, encoded as an array of token indices, and outputs an array of token embeddings. The encoder module is a transformer[27] with ten layers and four heads per layer. We use a token embedding size of 300 and hidden size of 400 in this work. Bottleneck module: next, amino-acid-level embeddings are passed through a bottleneck module made up of two fully connected networks. The first network reduces the dimensionality of each token to the latent dimension, 30, whereas the second predicts a vector of weights summing to 1. The outputs of these two networks are then combined in a pooling step where a weighted sum is taken across the 30-dimensional embeddings to produce a single sequence-level representation, $z \in \mathbb{R}^{30}$. This representation is of focus in this Article and is referred to as a protein sequence's latent representation. Sequence prediction module: to decode latent points to sequences, we use a deep convolutional network, comprised of four one-dimensional convolutional layers. ReLU (Rectified Linear Unit) activations and batchnorm layers are used between convolutional layers with the exception of the final layer. Fitness prediction module: the final module is a fitness prediction network, which predicts fitness from points latent in latent space. To encourage gradual changes to fitness in latent space we use a two-layer fully connected network regularized by a spectral norm penalty introduced in ref. [28]. As a result, the network is further encouraged to learn simpler organizations such as the pseudoconcave shape described in this work.

We train each model for 300,000 steps with a learning rate of 0.00002 on two 24 GB TITAN RTX graphics processing units, using a batch size of 64. For the negative sampling and interpolative sampling regularizations, samples of 32 artificial latent points are used in each forward pass, bringing the total effective batch size to 128.

Background and problem set-up. From a preliminary experimental screen of variants, a set of protein sequences $\mathcal{X}$, where each sequence is an ordered set of amino acids $x=(\sigma_1, \sigma_2, \ldots, \sigma_{N-1}, \sigma_N)$ composed from a finite alphabet of amino acids such that $\sigma_i \in .V$, $i \in [N]$, and their corresponding fitness values y, $y \in \mathbb{R}$, is produced. The final dataset $\mathcal{D}$ is then comprised of pairs (x_i, y_i) of sequences and their observed fitnesses. From these data, we wish to find sequences $x^\star \in S$ that possess a high degree of fitness, as measured by some threshold $\{y^\star \geq y_{\text{thresh}} \mid y^\star = \phi(x^\star), x^\star \in \Phi\}$. We also desire solutions in Φ to be diverse and novel.

Joint training. We formulate our method by first starting at the traditional perspective of sequence-based protein design. While directed evolution has yielded various successes in a range of domains over the years, it is susceptible to the underlying topology of the fitness landscape. This may lead to the accumulation of only locally optimal sequences at the completion of the optimization process. Recent work has sought to overcome the screening burden of directed evolution by performing in silico evaluations of a candidate sequence's fitness. This approach is comprised of training a model $\hat{\phi}_x$ to approximate the 'ground-truth' fitness landscape ϕ_x by minimizing an objective function of the form $\mathcal{L} = ||\hat{y} - y||$, where $\hat{y} = \hat{\phi}_x(x)$ and $y = \phi_x(x)$. Once the model has converged, it is then used to evaluate sequence candidates $\bar{x}$ using either an iterative modification or a sampling approach. In either case, the local sequence space around the original sequence is explored using minor changes to x, Δ_x. However, the difficult to predict relationship between Δ_x and changes in fitness Δ_y maintains the challenging nature of optimizing within sequence space.

A more recent approach to sequence-based protein design is to train a deep learning model to learn a representation of protein sequences by pretraining the model on a large corpus of protein sequence data[29] with an unsupervised task $||g(f(x)) - x||$. The end result of this pretraining is a trained encoder that has

learned a function $z=f_\theta(x)$, where z is understood to contain abstract and useful information about protein sequence composition. For the unsupervised model, it can be further stated that the learned latent code approximates the manifold on which the training data lies, where higher density is placed on realistic sequences.

Next, h_θ is trained on z to learn a fitness landscape $\hat{y} = \hat{\phi}_z = h_\theta(z)$ that will be used to evaluate new candidates $\bar{x}$. While this approach moves the optimization problem to a continuous setting, it suffers from a issue inherent in datasets from this domain. Often the datasets gathered from screening contain a small percentage of high-fitness sequences while the vast majority of sequences provide little to no measurable functionality. This produces high-fitness latent encodings that are hidden within a dense point cloud of low-fitness latent encodings, leading to inefficiencies when searching latent space through small perturbations to z, Δ_z. As a result, the optimization process will spend much of its time travelling through dense regions of low-fitness candidates. Again, a strong link between changes in sequence information, now encoded in Δ_z, and Δ_y has yet to be established.

Here we propose to connect these two important factors through the use of an autoencoder model trained jointly on the fitness prediction task, thereby combining the described two-step process into one. In this way, we add to the autoencoder model architecture, comprised of a encoder f and decoder g, a network h, which is tasked with predicting fitness from z. The final objective function of this set-up takes the form

$$\mathcal{L}_{\text{task}} = \gamma \mathcal{L}_{\text{recon}} + \alpha \mathcal{L}_{\text{reg}}$$

where $\mathcal{L}_{\text{recon}}$ represents the reconstruction task and $\mathcal{L}_{\text{reg}}$ represents the fitness prediction task. γ and α are scalar hyperparameters, which weight their respective tasks. We refer to this model as a JT-AE.

$$z^{(t+1)} \leftarrow z^{(t)} - \eta(\nabla_z \mathcal{L}_{\text{recon}} + \nabla_z \mathcal{L}_{\text{reg}}).$$

An important consequence of the joint-training set-up is that the latent code is updated during each training step with gradient signals from both sequence and fitness information. The resulting z encoding is thereby induced to resolve the two pieces of information. After model convergence, the latent space is endowed with a strong sequence–fitness association, which we leverage here for latent space optimization.

$$\mathcal{L} = ||g(f(x)) - x|| + ||h(f(x)) - y||.$$

Furthermore, we can observe that in each update step the encoder receives gradients from both the reconstruction loss and fitness prediction loss and is therefore directed to encode information about sequence and fitness in z. Indeed, when the dimensionality of z is set to some low value $d \ll N$ the latent encoder is forced to include only the most salient information about sequence and fitness and induces a greater connection between the two in z. Through the use of this training strategy, we strengthen the connection between Δ_z and Δ_y for downstream applications.

Negative sampling. A pervasive challenge of optimizing within the latent space of deep learning models is moving far from the training data into regions where the model's predictive performance deteriorates or is otherwise untrustworthy. Recent work has proposed techniques to define boundaries for model-based optimization, such as through a sequence mutation radius[11] or by relying on model-derived likelihood values[9]. In general, the gradients produced by a supervised network do not readily provide a stopping criterion or any strong notion of bounds in regards to the range of values the network predicts. This can be further shown by training a network to predict from a two-dimensional latent representation and overlaying the gradient directions onto the latent space. A unidirectional organization by the predicted attribute is the likely outcome, as shown by ref. [23] and Fig. 1c.

In this Article, we propose a solution that addresses the challenge of moving away from training points in latent space by focusing on the function learned by the neural network. During training, the model takes in data in batches of size N randomly sampled from the training data. As the output of the encoder module of ReLSO, these data points are encoded in a low-dimensional latent space. To keep latent embeddings close to the origin, we include a norm-based penalty on the encoding. This then allows for the generation of negative samples by randomly sampling high-norm points in latent space. We sample M latent points with L2-norms greater than that of the largest L2-norm observed in the original N points. A hyperparameter is used to scale the allowed difference between the max L2-norm of the training samples and the min L2-norm of the negative samples. The training samples and negative samples are then concatenated batchwise and passed through the fitness prediction network. In the calculation of the mean-squared regression loss, the predicted fitness values for the negative samples, y_{neg}, are compared with a preset value In this Article, we set this value as the minimum observed fitness in the dataset, $\min(Y)$. The fitness prediction loss term is now the following:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{n}(y_i - \hat{y_i})^2 + \frac{1}{M}\sum_{i=1}^{m}(y_{\text{neg}} - \min(Y))^2.$$

While negative sampling effectively restricts the function learned by h_θ to resemble a concave shape, the ability of neural networks to learn a wide variety of functions can still allow for complex, non-concave shaped solutions to persist. In the bottom rows of Fig. 1c, we further encourage learning of a smooth fitness function using spectral norm regularized layers from ref. [28], labelled as 'spectral'. We compare this with an unregularized fully connected layer, labelled as 'base'.

Smoothness index. As smoothness within the latent space encodings of our model plays a major role in our approach, we measure smoothness with a metric used in ref. [24]. We construct a symmetric KNN graph from the latent codes $Z=z_i,z_j,\ldots$ from a set of sequences such that z_i and z_j are connected by an edge either if z_i is within the k nearest neighbours of z_j or conversely if z_j is within the k nearest neighbours of z_i. By constructing our graphs in this way, we ensure our metric is scale invariant. The KNN graph A is then used to construct the combinatorial graph Laplacian operator $L=D-A$, where D is the degree matrix. From this operator we calculate our smoothness metric as

$$\lambda_y = \frac{1}{N}\, y^{\mathrm{T}} L y$$

where y is our signal of interest and N corresponds to the number of data points used to construct the graph. The quadratic form of the graph Laplacian operator can be interpreted as taking the sum of squared differences along edges in the underlying graph such that the resulting sum is lower if the signal is smooth, that is, small differences between neighbouring points.

Datasets. Quantitative readouts of fitness landscapes have remained elusive until novel breakthroughs in high-throughput molecular biology, such as directed evolution and deep mutational scanning. Broadly speaking, these methods aim to introduce mutations in the sequence (or a set of interesting positions) in a systematic (saturation mutagenesis) or random (directed evolution) manner.

GIFFORD dataset. Enrichment data from directed evolution: in this experiment, Liu et al.[15] pitted a vast library (10^{10} unique mutants) of an antibody against a single target. This library was then pitted through three consecutive rounds of selection, washing and amplification. Next-generation sequencing was used between rounds to identify which sequences were enriched. Here, the fitness is the log ratio of sequence enrichment between rounds of selection (that is, how well the sequence performed relative to other members of the library).

GB1 dataset. Wu et al. carried out a saturation mutagenesis study targeting four sites[29,30] and generated all 20^4 possible mutants to explore the local fitness landscape of GB1, an immunoglobulin-binding protein. This particular site is known to be an epistatic cluster. Fitness was measured by testing stability and binding affinity.

GFP dataset. Sarkisyan et al.[25] carried out random mutagenesis on a fluorescent protein (avGFP) to generate 51,175 unique protein coding sequences, with a mean of 3.7 mutations. Fitness was determined by measuring the fluorescence of mutated constructs via a FACS assay.

TAPE dataset. In addition to the datasets we pulled from previous work, we used the TAPE benchmark datasets for fluorescence from ref. [18]. Note that we also kept the train–test–validation splits consistent so as to establish a fair comparison. The data here are the same as in ref. [25] but simply split by sequence distance.

Optimization. *Sequence-space optimization.* We compare ReLSO with two popular approaches for ML-based protein sequence optimization. These methods manipulate sequences directly and use a machine learning model to screen candidates, effectively treating model inference as a substitute for wet-lab characterization. First, we compare against in silico directed evolution, as described in ref. [10]. Here a subset of residues from the protein sequence of interest is iteratively expanded and screened in silico. The best amino acid for each position is then held constant while the next position is optimized. Second, we compare against the Metropolis–Hastings MCMC approach used in ref. [11], where protein sequences undergo random mutagenesis. All mutations with improved fitness are accepted into the next iteration and a few mutations with reduced fitness are also carried forward. In our comparisons, we refer to this approach as MCMC Seq and the directed evolution approach as DE.

Gradient-free optimization. The first set of gradient-free algorithms employs a local search where a small perturbation in the latent encoding is added $z_{t+1}=z_t+\epsilon$, where t is the step index and z_{t+1} is accepted with a probability equal to $\min(1, \exp(\frac{y_{t+1}-y_t}{kT})$. The second group of gradient-free optimization algorithms uses a nearest-neighbour search and either moves in the direction of the most fit neighbour (hill climbing) or chooses uniformly from $\Phi=\{z_j \mid h(z_j)=y_j>y_i\}$ (stochastic hill climbing). Since we train the fitness prediction head directly from the latent encoding, we have access to the gradients of this network and can perform gradient ascent. We also examine two gradient-free methods that operate in sequence space. One such method is a form of in silico directed evolution where positions are independently and sequentially optimized. The second optimization

approach mirrors that of the Metropolis–Hastings Monte Carlo search approach used in latent space with the exception that the perturbation step is replaced with a mutation step.

Gradient-free latent space optimization. Next we consider a set of optimization algorithms that operate in the latent space of generative models. These methods still treat the prediction network as a black box, unable to access gradients, but manipulate sequences using their latent encodings. First, we pursue a simple hill-climbing algorithm, which takes a greedy search through latent space. We also test a stochastic variant of hill climbing, which should better avoid local minima. Here the algorithm samples z_{t+1} uniformly from $\{z \mid h_\theta(z+\epsilon) > h_\theta(z_t), \epsilon \sim \mathcal{N}(\mu, k)\}$, where k is a parameter. We refer to this variation of hill climbing as stochastic hill climbing. Furthermore, we use an MCMC scheme similar to the previously described approach of ref. [11]; however, here we perform this optimization in latent space. We apply a small perturbation in the latent encoding $z_{t+1} = z_t + \epsilon$, where t is the step index and z_{t+1} is retained according to a probabilistic acceptance step. Finally, we consider the approach of refs. [8] and [9], where an adaptive sampling procedure is used to generate improved sequences. We denote these approaches as DbAS and CbAS.

Gradient-based latent space optimization. We examine the performance of a latent space gradient-ascent optimization approach. Here we leverage the ability to extract gradient directions provided by the jointly trained h_θ. These directions allow for latent space traversal to areas of latent space associated with higher-fitness sequences. We first examine this approach through a jointly trained autoencoder without the aforementioned regularizations and denote this approach as JT-AE–GA. Next, we examine the performance of gradient ascent using the interpolation sampling and negative sampling regularizations of ReLSO and refer to this approach as ReLSO–GA.

CbAS and DbAS. In this work we also compare the DbAS and CbAS methods introduced in refs. [8] and [9], respectively. In this approach, the fitness prediction model is treated as a black-box oracle, which maps from design space to a distribution over properties of interest. Similarly to our approach, the authors of these methods consider the pathologies inherent to relying on deep learning models to optimize protein sequences. To address this, DbAS and CbAS use a model-based adaptive sampling technique, which draws from the latent space of a generative model. While DbAS assumes an unbiased oracle, CbAS conditions the sampling procedure on the predictions of a set of oracle models. in this study, we use an implementation sourced from a publicly available GitHub repository (https://github.com/dhbrookes/CbAS). To ensure representative performance, we first evaluated DbAS and CbAs on the datasets using a grid search over several values of q ([0.5, 0.6, 0.7, 0.8]) and number of epochs using in training ([1, 5, 10, 15]).

Data availability

The datasets used in this study are available in their processed form at the project's GitHub repository (https://github.com/KrishnaswamyLab/ReLSO-Guided-Generative-Protein-Design-using-Regularized-Transformers/tree/main/data). Additionally, we include links to the original data sources in the README file of the repository.

Code availability

Implementations of the models and optimization algorithms are available at the project's GitHub repository (https://github.com/KrishnaswamyLab/ReLSO-Guided-Generative-Protein-Design-using-Regularized-Transformers) and archived on Zenodo (https://doi.org/10.5281/zenodo.6946416).

Received: 27 November 2021; Accepted: 9 August 2022;
Published online: 26 September 2022

References

1. Tiessen, A., Pérez-Rodríguez, P. & Delaye-Arredondo, L. J. Mathematical modeling and comparison of protein size distribution in different plant, animal, fungal and microbial species reveals a negative correlation between protein size and protein number, thus providing insight into the evolution of proteomes. *BMC Res. Notes* **5**, 85 (2012).
2. Starr, T. N. & Thornton, J. W. Epistasis in protein evolution. *Protein Sci.* **25**, 1204–1218 (2016).
3. Romero, P. A. & Arnold, F. H. Exploring protein fitness landscapes by directed evolution. *Nat. Rev. Mol. Cell Biol.* **10**, 866–876 (2009).
4. Chen, K. & Arnold, F. H. Engineering new catalytic activities in enzymes. *Nat. Catal.* **3**, 203–213 (2020).
5. Arnold, F. H. Design by directed evolution. *Acc. Chem. Res.* **31**, 125–131 (1998).
6. Rohl, C. A., Strauss, C. E. M., Misura, K. M. S. & Baker, D. Protein structure prediction using Rosetta. *Methods Enzymol.* **383**, 66–93 (2004).
7. Norn, C. et al. Protein sequence design by conformational landscape optimization. *Proc. Natl Acad. Sci. USA* **118**, e2017228118 (2021).
8. Brookes, D. H. & Listgarten, J. Design by adaptive sampling. Preprint at https://arxiv.org/abs/1810.03714 (2018).
9. Brookes, D., Park, H. & Listgarten, J. Conditioning by adaptive sampling for robust design. *Proceedings of the 36th International Conference on Machine Learning* **97**, 773–782 (2019).
10. Yang, K. K., Wu, Z. & Arnold, F. H. Machine-learning-guided directed evolution for protein engineering. *Nat. Methods* **16**, 687–694 (2019).
11. Biswas, S., Khimulya, G., Alley, E. C., Esvelt, K. M. & Church, G. M. Low-*n* protein engineering with data-efficient deep learning. *Nat. Methods* **18**, 389–396 (2021).
12. Linder, J. & Seelig, G. Fast differentiable DNA and protein sequence optimization for molecular design. Preprint at https://arxiv.org/abs/2005.11275 (2020).
13. Angermueller, C. et al. Model-based reinforcement learning for biological sequence design. In *International Conference on Learning Representations* (2019).
14. Alley, E. C., Khimulya, G., Biswas, S., AlQuraishi, M. & Church, G. M. Unified rational protein engineering with sequence-based deep representation learning. *Nat. Methods* **16**, 1315–1322 (2019).
15. Liu, G. et al. Antibody complementarity determining region design using high-capacity machine learning. *Bioinformatics* **36**, 2126–2133 (2020).
16. Jumper, J. et al. Highly accurate protein structure prediction with AlphaFold. *Nature* **596**, 583–589 (2021).
17. Rao, R. et al. Evaluating protein transfer learning with TAPE. *Adv. Neural Inf. Process. Syst.* **32**, 9689–9701 (2019).
18. Rao, R., Ovchinnikov, S., Meier, J., Rives, A. & Sercu, T. Transformer protein language models are unsupervised structure learners. Preprint at *bioRxiv* https://doi.org/10.1101/2020.12.15.422761 (2020).
19. Rives, A. et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proc. Natl Acad. Sci. USA* **118**, e2016239118 (2021).
20. Vig, J. et al. BERTology meets biology: interpreting attention in protein language models. Preprint at https://arxiv.org/abs/2006.15222 (2020).
21. Hochreiter, S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* **6**, 107–116 (1998).
22. Detlefsen, N. S., Hauberg, S. & Boomsma, W. Learning meaningful representations of protein sequences. *Nat. Commun.* **13**, 1914 (2022).
23. Gómez-Bombarelli, R. et al. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Cent. Sci.* **4**, 268–276 (2018).
24. Castro, E., Benz, A., Tong, A., Wolf, G. & Krishnaswamy, S. Uncovering the folding landscape of RNA secondary structure using deep graph embeddings. 2020 IEEE International Conference on Big Data. 4519–4528 (2020).
25. Sarkisyan, K. S. et al. Local fitness landscape of the green fluorescent protein. *Nature* **533**, 397–401 (2016).
26. Rodrigues, C. H., Pires, D. E. & Ascher, D. B. Dynamut2: assessing changes in stability and flexibility upon single and multiple point missense mutations. *Protein Sci.* **30**, 60–69 (2021).
27. Vaswani, A. et al. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **30** (2017).
28. Yoshida, Y. & Miyato, T. Spectral norm regularization for improving the generalizability of deep learning. Preprint at https://arxiv.org/abs/1705.10941 (2017).
29. Mistry, J. et al. Pfam: the protein families database in 2021. *Nucleic Acids Res.* **49**, D412–D419 (2021).
30. Wu, N. C., Dai, L., Olson, C. A., Lloyd-Smith, J. O. & Sun, R. Adaptation in protein fitness landscapes is facilitated by indirect paths. *eLife* **5**, e16965 (2016).

Acknowledgements

E.C. is funded by a National Library of Medicine training grant (LM007056). D.B. is funded by a Yale–Boehringer Ingelheim Biomedical Data Science Fellowship. S.K. acknowledges funding from the NIGMS (R01GM135929, R01GM130847), NSF Career Grant (2047856), Chan–Zuckerberg Initiative Grants (CZF2019-182702 and CZF2019-002440) and the Sloan Fellowship (FG-2021-15883). We thank A. Tong and other members of the Krishnaswamy Lab for their suggestions and discussion on this project.

Author contributions

E.C. and S.K. conceived and planned the work presented here. E.C. implemented the models and performed numerical experiments, with help from A.G., J.R. and K.G. K.G. and D.B. performed in silico validation of the model predictions. E.C. and S.K. authored the manuscript with assistance from D.B. in proofreading and formatting. All authors discussed the results and commented on the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s42256-022-00532-1.

Correspondence and requests for materials should be addressed to Smita Krishnaswamy.

Peer review information *Nature Machine Intelligence* thanks Brandon Allgood, Markus Buehler and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.