LatentDE: Latent-based Directed Evolution accelerated by Gradient Ascent for Protein Sequence Design

Thanh V. T. Tran[*,1] **Nhat Khang Ngo**[*,†,1] **Viet Thanh Duy Nguyen**[1] **Truong Son Hy**[‡,2]
[1] FPT Software AI Center, Vietnam
[2] University of Alabama at Birmingham, United States

Abstract

Directed evolution has been the most effective method for protein engineering that optimizes biological functionalities through a resource-intensive process of screening or selecting among a vast range of mutations. To mitigate this extensive procedure, recent advancements in machine learning-guided methodologies center around the establishment of a surrogate sequence-function model. In this paper, we propose Latent-based Directed Evolution (LDE), an evolutionary algorithm designed to prioritize the exploration of high-fitness mutants in the latent space. At its core, LDE is a regularized variational autoencoder (VAE), harnessing the capabilities of the state-of-the-art Protein Language Model (pLM), ESM-2, to construct a meaningful latent space of sequences. From this encoded representation, we present a novel approach for efficient traversal on the fitness landscape, employing a combination of gradient-based methods and directed evolution. Experimental evaluations conducted on eight protein sequence design tasks demonstrate the superior performance of our proposed LDE over previous baseline algorithms. We public our code at `https://github.com/HySonLab/LatentDE`.

1 Introduction

The field of protein engineering aims to strategically create or identify proteins that have practical applications with desired biological functions, such as fluorescence intensity [8], enzyme activity [22], and therapeutic efficacy [37]. The amino acid sequence of a protein governs the properties associated with its function through its spontaneous folding into three-dimensional structures [23, 17]. The mapping from protein sequence to functional property forms a protein fitness landscape [70] that characterizes the protein functional levels. The evolution in nature can be regarded as a searching procedure on the protein fitness landscape [42]. This natural process inspires the innovation of *directed evolution* [3], the most widely-applied approach for engineering protein sequences. It starts with a protein having some level of the desired function. Subsequently, a series of mutation and screening rounds are undertaken, wherein mutations are introduced to generate a collection of variant proteins. Through this iterative process, the most optimal variant is identified, and the cycle continues until a satisfactory level of improvement is achieved.

Recent machine learning (ML) methods have been studied to improve the sample efficiency of this evolutionary search [46, 68, 47, 19]. However, these approaches require repetitive rounds of random mutagenesis and wet-lab validation, which are both money-consuming and time-intensive. In addition, while these methods can reduce experimental screening load by proposing potentially

[*] Equal contribution
[†] Now at Mila, McGill University
[‡] Correspondence at thy@uab.edu

AI for New Drug Modalities at NeurIPS 2024.

promising sequences, they only operate within a discrete space and make modifications to the amino acid sequence directly. In this manner, the challenge of navigating the sequence space remains unaddressed. The search space of protein sequences is discrete and combinatorially large, and most proteins have low fitness [11]. Moreover, protein fitness datasets are usually scarce as creating them requires costly wet-lab experiments [18]. These challenges make ML methods often stuck in local optima and prone to predict false positive samples [10]. An alternative methodology to working in the sequence space is to focus on acquiring a low-dimensional, semantically rich representation of proteins. These latent representations collectively form a latent space, offering more navigable and smoother terrain as they encapsulate the fitness landscape of the sequences into continuous forms. With this approach, a therapeutic candidate can be optimized using its latent representation in a procedure called latent space optimization (LSO). However, the integration of LSO with directed evolution remains a relatively unexplored territory.

Designing high-fitness protein sequences is regarded as a black-box model-based optimization (MBO) problem, where the primary goal is to explore design inputs maximizing a black-box objective (e.g., fitness). Typically addressed through an online iterative process, online MBO, in each iteration, proposes novel designs and solicits feedback from the unknown objective function to enhance subsequent design proposals. However, accessing the true objective function of protein fitness proves challenging due to constraints such as limited experimental data and computational resources. On the other hand, offline MBO emerges as an efficient for design desirable outputs without accessing true objection functions. Offline algorithms are allowed to observe a static dataset of designs and produce data samples that are later re-evaluated by the truth objective function. Albeit efficient for optimization, offline MBO often faces an out-of-distribution issue wherein the optimized design input x may be unseen to a learned proxy $\hat{f}(x)$, which replaces the ground truth objective for evaluating the design outputs. As $\hat{f}(x)$ is trained to approximate the fitness landscape of a training dataset, out-of-distribution inputs may negatively affect its predictions [36], thereby fooling the discovery algorithm to produce non-desirable designs.

To overcome the above challenges, in this paper, we propose **L**atent-based **D**irected **E**volution (LDE), a novel approach that, to the best of our knowledge, is the first latent-based method for directed evolution. Specifically, by leveraging the latent space of ESM-2 [39], a state-of-the-art pre-trained protein language model (pLM), LDE learns to reconstruct and predict the fitness value of the input sequences in the form of a variational autoencoder (VAE) regularized by supervised signals. Post-training, LDE addresses the limitations of the two MBO methods by combining the best of both worlds. Starting from a wild-type sequence, LDE first encodes it into the latent representation, on which the gradient ascent is performed as an efficient offline MBO algorithm that guides the latent codes to reach high-fitness regions on the simulated landscape. Within these regions, to address the out-of-distribution limitation of the offline counterpart, LDE integrates latent-based directed evolution. This involves iterative rounds of randomly adding scaled noise to the latent representations, facilitating local exploration around high-fitness regions. The noised latent representations are decoded into sequences and evaluated by the truth oracles (e.g., wet-lab experiments, large ML models), and top samples are selected for the next round of the algorithm, enabling an efficient gradient-free sampling strategy. In summary, the contributions of our paper are outlined as follows:

- We propose a latent-based method for directed evolution for biological sequence design.
- We address the limitations of offline and online MBO methods by combining the best of both worlds, using gradient ascent as the warm-up procedure to guide the latent representations to feasible regions for accelerating latent-based directed evolution.
- Empirical evaluations conducted on eight protein datasets demonstrate that our proposed LDE surpasses the SOTA method in terms of fitness scores. We additionally conduct several ablation studies to showcase the effectiveness of our method, as well as to gain more insights into ML-based protein design.

2 Related works

ML for Fitness Landscape Modeling. The application of ML in protein engineering has experienced a significant upswing, particularly in the domain of modeling the protein fitness landscape. Several methodologies have been proposed, leveraging co-evolutionary information extracted from

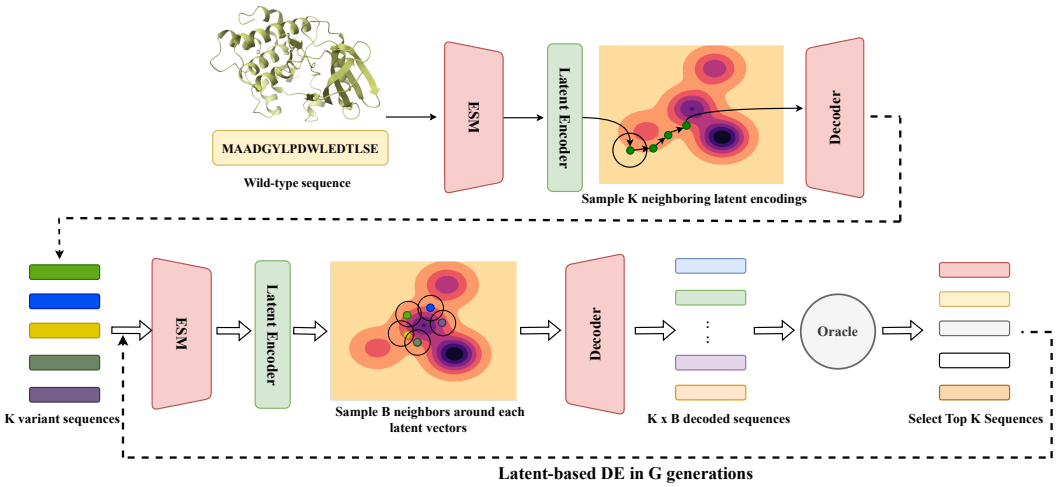


Figure 1: Overview of our proposed method. **Top**: We iteratively encode the wild-type sequence into latent variables and run gradient ascent to move them towards regions of high fitness on the approximated landscape. This process yields an initial population of K-diverse variant sequences. **Bottom** From K sequences, our latent-based directed evolution is performed in G generations. Sequences in the previous generation are encoded into the latent space at each generation. Then, B neighbors are sampled around those latent variables and decoded into B sequences, which are evaluated by a black-box oracle. The protein used in this figure is only for illustration purposes.

multiple sequence alignments to predict fitness scores [26, 50, 40]. Alternatively, pre-trained language models have been employed for transfer learning or zero-shot inference [48, 1, 43, 27]. The learned protein landscape models can be used to replace the expensive wet-lab validation to screen enormously designed sequences [49, 57].

Latent-based Methods for Sequence Design. Directed evolution represents a classical paradigm in protein sequence design that has achieved significant successes [6, 20, 62, 5]. Within this framework, various ML algorithms have been proposed to improve the sample efficiency of the evolutionary search [46, 49, 47, 19, 63]. However, the majority of these methods are designed to optimize protein sequences directly in the sequence space, dealing with discrete, high-dimensional decision variables. Alternatively, [24] and [13] employed a VAE model and applied gradient ascent to optimize the latent representation, which is subsequently used to decode into string-based representations of small molecules. Similarly, [38] employed an off-policy reinforcement learning method to facilitate updates in the representation space. Another notable approach [58] involves training a denoising autoencoder with a discriminative multi-task Gaussian process head, enabling gradient-based optimization of multi-objective acquisition functions in the latent space of the autoencoder. Recently, latent diffusion has been introduced for designing novel proteins, harnessing the capabilities of a protein language model [14]. Diverging from the aforementioned works, we introduce a novel approach wherein, within the latent representation of a VAE model, gradient ascent is employed as a warm-up phase for latent-based directed evolution.

3 Method

We present the preliminary section in Appendix A. Our work introduces a novel approach that utilizes directed evolution to efficiently discover optimal protein sequences directly from their latent representations.

We begin with the problem formulation in Section 3.1. Then, we present our pre-trained regularized variational autoencoders (VAEs) [32] in Section 3.2 and how to perform directed evolution, which is accelerated by gradient ascent, in the latent space of VAEs in Section 3.3.

3.1 Problem Formulation

We consider the problem of designing protein sequences to search for high-fitness sequences s in the sequence space $\mathcal{V}^L$, where $\mathcal{V}$ denotes the vocabulary of amino acids (i.e. $|\mathcal{V}| \approx 20$ because both animal and plant proteins are made up of about 20 common amino acids) and L is the desired sequence length. We aim to design sequences that maximize a black-box protein fitness function $\mathcal{O} : \mathcal{V}^L \mapsto \mathbb{R}$ which can only be measured by wet-lab experiments:

$$x^* = \text{argmax}_{x \in \mathcal{V}^L} \mathcal{O}(x). \tag{1}$$

For in-silico evaluation, given a static dataset of all known sequences and fitness measurements $\mathcal{D}^* = \{(x, y) | x \in \mathcal{V}^L, y \in \mathbb{R}\}$, an oracle $\mathcal{O}_\psi$ parameterized by ψ is trained to minimize the prediction error on $\mathcal{D}^*$. Afterward, $\mathcal{O}_\psi$ is used as an approximator for the black-box function $\mathcal{O}$ to evaluate computational methods that are developed on a training subset $\mathcal{D}_t$ of $\mathcal{D}^*$. In other words, given $\mathcal{D}_t$, our task is to generate sequences $\hat{x}$ that optimize the fitness approximated by $\mathcal{O}_\psi$.

In this paper, we focus on designing sequences upon wild-type sequences as the same with traditional evolution and exploration algorithms described in [4, 6, 49, 2]. Specifically, the algorithms commence with a single wild-type sequence and iteratively generate candidates with enhanced properties from previous references. Figure 1 illustrates the overview of our method.

3.2 Regularized Variational Autoencoder

Considering a dataset of protein sequences $x \in \mathcal{V}^L$ with corresponding fitness values $y \in \mathbb{R}$, denoted as $\mathcal{D}_t = \{(x_i, y_i)\}_{i=1}^N$, we train a VAE comprising an encoder $\phi : \mathcal{V}^L \mapsto \mathbb{R}^{d_h}$ and a decoder $\theta : \mathbb{R}^d \mapsto \mathcal{V}^L$. Each sequence x_i is encoded into a low-dimensional vector $h_i \in \mathbb{R}^{d_h}$. Subsequently, the mean $\mu_i \in \mathbb{R}^d$ and log-variance $\log \sigma_i \in \mathbb{R}^d$ of the variational posterior approximation are computed from h_i using two feed-forward networks, $\mu_\phi : \mathbb{R}^{d_h} \mapsto \mathbb{R}^d$ and $\sigma_\phi : \mathbb{R}^{d_h} \mapsto \mathbb{R}^d$. A latent vector $z_i \in \mathbb{R}^d$ is then sampled from the Gaussian distribution $\mathcal{N}(\mu_i, \sigma_i^2)$, and the decoder θ maps z_i back to the reconstructed protein sequence $\hat{x}_i \in \mathcal{V}^L$. The training objective involves the cross-entropy loss $\mathcal{C}(\hat{x}_i, x_i)$ between the ground truth sequence x_i and the generated $\hat{x}_i$, as well as the Kullback-Leibler (KL) divergence between $\mathcal{N}(\mu_i, \sigma_i^2)$ and $\mathcal{N}(0, I_d)$:

$$\mathcal{L}_{vae} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{C}(\hat{x}_i, x_i) + \frac{\beta}{N} \sum_{i=1}^{N} D_{\text{KL}}(\mathcal{N}(\mu_i, \sigma_i^2) \| \mathcal{N}(0, I_d)). \tag{2}$$

Here, β is the hyperparameter to control the disentanglement property of the VAE's latent space.

Latent Space Regularization One can consider the KL-divergence loss term in Equation (2) as a regularizer for training autoencoders. In particular, the KL-divergence regularizer makes the encoder ϕ produce latent space that is close to the unit Gaussian prior (i.e., the global optimum). The learned latent space could be smooth and convex, which is useful for model-based searching algorithms during optimization. However, forcing the approximate posterior $q_\phi(z|x)$ to be unit Gaussian may lead to the fact that the encoder network ϕ may encode zero information into the latent space [15], resulting in less meaningful latent representations. This is not a good phenomenon for our latent-based directed evolution algorithm that aims at locally searching for high-fitness proteins by iteratively modifying the reference sequences (see Appendix A). Regarding learning protein fitness landscape, the percentage of high-fitness sequences in datasets is relatively small compared to those with low fitness scores. Consequently, the latent representations of high-fitness proteins may be hidden within a dense of low-fitness ones [13], leading to inefficient searching processes. To address this issue, following [13], we jointly train our VAEs with a fitness predictor $f : \mathbb{R}^d \mapsto \mathbb{R}$ that maps the latent vectors $z \in \mathbb{R}^d$ to the fitness space. This leads to the final loss for training ϕ, θ, and f as:

$$\mathcal{L} = \mathcal{L}_{vae} + \frac{1}{N} \sum_{i=1}^{N} ||f(z_i) - y_i||_2^2, \tag{3}$$

where $y_i \in \mathbb{R}$ is the experimental fitness of sequence $x_i \in \mathcal{V}^L$, which is encoded into latent vector $z_i = \phi(x_i) \in \mathbb{R}^d$. This additional L_2-regularizer forces the VAE's encoder to generate latent representations of protein sequences with comparable fitness scores to be closely positioned in the latent space, thereby separating those with high fitness values from dense clouds associated with low-fitness regions.

3.3 Latent-Based Directed Evolution

Inspired by nature's evolutionary process, where subtle mutations in a protein's sequence can significantly improve its fitness, our method explores the local region of the wild-type sequence's latent encoding through iterative sampling from the approximate distribution encoded by the pre-trained encoder ϕ.

A crucial challenge arises from the possibility that the latent representation z of x^{wt} may be situated in low-fitness regions of the protein landscape. This can lead to slow convergence in directed evolution, where offspring generated by random mutations of low-fitness parents tend to inherit similar limitations. To address this problem, we incorporate gradient ascent as a preliminary step to generate the initial population $\mathcal{P}_0$, accelerating the evolutionary process towards more advantageous regions. Algorithm 1 provides a detailed, line-by-line breakdown of our approach.

Gradient Ascent (GA) At the beginning of our algorithm, the wild-type protein sequence x^{wt} is encoded by the encoder ϕ to produce its mean $\mu_\phi(x^{wt})$ and log variance $\log \sigma_\phi(x^{wt})$. A latent representation z is then sampled from $\mathcal{N}(\mu_\phi(x^{wt}), \sigma_\phi(x^{wt})^2)$. Subsequently, we perform gradient ascent with a learning rate of α in T iterations to move z to high-fitness regions as:

$$z_{t+1} = z_t + \alpha \nabla_z f(z)|_{z=z_t}, 0 \leq t < T. \tag{4}$$

Where $\nabla_z f(z)|_{z=z_t}$ is the gradient of the fitness predictor f with respect to z_t. After T iterations, we add z_T to the initial population $\mathcal{P}_0$. The procedure is iteratively executed until K pairs of decoded sequences along with their respective fitness scores are obtained. At this stage, the fitness scores are computed by the latent predictor f, allowing fast computation and efficient latent sampling (see lines 1 to 7).

Algorithm 1 Latent-based directed evolution accelerated by gradient ascent

Input: x^{wt}, encoder ϕ, decoder θ, fitness predictor f, # iterations T, # generations G, beam size B, oracle $\mathcal{O}$.

1: $\mathcal{P}_0 \leftarrow \emptyset$
2: **for** $i = 1$ **to** K **do**
3: Sample $z \sim \mathcal{N}(\mu_\phi(x^{wt}), \sigma_\phi(x^{wt})^2)$
4: $z_T \leftarrow$ gradient_ascent(z, f, T) in Equation (4)
5: $\hat{x} \leftarrow \theta(z_T)$
6: $P_0 \leftarrow P_0 \cup \{(\hat{x}, \mathcal{O}(\hat{x}))\}$
7: **end for**
8: **for** $g = 1$ **to** G **do**
9: $P_g \leftarrow \emptyset$
10: **for** $k = 1$ **to** K **do**
11: **for** $b = 1$ **to** B **do**
12: Sample $z_{k,b} \sim \mathcal{N}(\mu_\phi(x_k), \sigma_\phi(x_k)^2)$
13: $p \sim \mathcal{U}[0, 1)$
14: **if** $p >$ threshold **then**
15: Inject noise to $z_{k,b}$ based on Equation (5)
16: **end if**
17: $\hat{x}_{k,b} \leftarrow \theta(z_{k,b})$
18: $P_g \leftarrow P_g \cup \{(\hat{x}_{k,b}, \mathcal{O}(\hat{x}_{k,b})\}$
19: **end for**
20: $P_g \leftarrow P_g \cup P_{g-1}$
21: $P_g \leftarrow$ topK(P_g)
22: **end for**
23: **end for**

Evolutionary Process From lines 8 to 23, a latent-based directed evolution is conducted in G generations to generate K protein sequences with high fitness values. For each candidate x_k, we sample B latent variables from its posterior distribution, where each is denoted as $z_{k,b} \sim \mathcal{N}(\mu_\phi(x_k), \sigma_\phi(x_k)^2)$. These latent representations are then decoded into sequences by the decoder θ, i.e. $\hat{x}_{k,b} = \theta(z_{k,b})$. Wet-lab experiments then evaluate the decoded sequences to obtain their fitness scores. All sequences generated in generation g are added into $\mathcal{P}_g$. Finally, at the end of g, the top K sequences are selected from both $\mathcal{P}_{g-1}$ and $\mathcal{P}_g$.

Random Exploration Although sampling around the local areas of high-fitness latent codes can guarantee the superiority of the generated sequences, the search process may be prone to be trapped in these local regions after a certain number of generations, thereby hindering the exploration of potentially promising sequences, which are unseen before. As a result, we randomly add white noise to the latent variables when $p \sim \mathcal{U}[0, 1)$ is higher than a threshold. As demonstrated in line 14 of Algorithm 1, the formula is defined as:

$$z_l = z + (\gamma - \delta g)\epsilon, \epsilon \sim \mathcal{N}(0, I), \tag{5}$$

Table 1: Maximum fitness scores on eight protein datasets. Shaded rows indicate the results of ablation studies.

	avGFP	AAV	TEM	E4B	AMIE	LGK	Pab1	UBE2I
x^{wt}	1.408	-6.778	-0.015	0.774	-2.789	-1.260	0.014	-0.262
AdaLead	3.323	-1.545	0.248	-0.373	-1.483	-0.047	0.382	2.765
DyNA PPO	5.331	-2.817	0.570	-0.575	-2.790	-0.060	0.183	2.630
CbAS	5.187	-2.800	0.481	-0.658	-1.784	-0.056	0.276	2.693
CMA-ES	5.125	-3.267	0.590	-0.658	-2.790	-0.086	0.254	2.527
COMs	3.544	-3.533	0.472	-0.860	-20.182	-0.087	0.156	2.086
PEX	3.796	2.378	0.252	4.317	-0.364	0.009	1.326	3.578
GFN-AL	5.028	-4.444	0.654	-0.831	-37.360	-5.738	1.399	3.850
GGS	3.368	2.442	1.121	-1.147	-3.364	-0.972	0.059	4.101
LDE (ours)	**8.058**	**2.636**	**1.745**	**5.120**	-0.103	**0.018**	1.548	**4.297**
− w/o GA	6.407	2.148	1.220	4.597	**-0.099**	-0.531	**1.592**	3.254
− w/o DE	3.677	0.919	-0.024	3.052	-0.701	-1.597	0.285	0.766

where $\gamma \in \mathbb{R}$ denotes the step size that controls the exploration rate, and δ is the annealing factor at the generation g of the evolution process. We hypothesize that when g gets close to G, i.e., the total number of generations, the population tends to contain superior samples; thus, we slow down the exploration to avoid degeneration at the end of the algorithm.

Efficient Sampling and Optimization Traditional directed evolution is effective but computationally intensive for protein design. Our latent-based algorithm improves sampling efficiency by compressing long sequences into low-dimensional latent representations via VAEs, simplifying the mutation process as noise addition in latent space. This approach, unlike complex mutation operators [4, 63], requires no domain knowledge and is more computationally efficient. The gradient ascent benefits from VAE regularization, creating a smoother optimization landscape and reducing the risk of local optima. Furthermore, even if the optimization converges to a local optimum, latent-based directed evolution (DE) can explore surrounding regions through controlled random perturbations for a broader search, enabling a more comprehensive search of promising areas.

4 Experiments

In this section, we undertake a comprehensive set of experiments to assess and validate the efficacy of our proposed LDE on the task of protein sequence design.

4.1 Experimental Setup

Datasets Following [49] and [57], we assess the performance of our method across eight protein engineering benchmarks: (1) Green Fluorescent Protein (**avGFP**), (2) Adeno-Associated Viruses (**AAV**), (3) TEM-1 β-Lactamase (**TEM**), (4) Ubiquitination Factor Ube4b (**E4B**), (5) Aliphatic Amide Hydrolase (**AMIE**), (6) Levoglucosan Kinase (**LGK**), (7) Poly(A)-binding Protein (**Pab1**), (8) SUMO E2 Conjugase (**UBE2I**), The detailed data descriptions and statistics, including protein sequence length, data size, and data percentiles, are provided in Appendix C.

Implementation Details The model training is conducted using a single NVIDIA A100 card, employing the Adam optimization algorithm [31] with a learning rate of 2e-4. Each dataset is randomly split into training and validation sets at a ratio of 9:1. To control the disentanglement property in the latent representation, we adopt the strategy proposed by [55] and set the expected KL values to be 20. We train the VAE model for 130 epochs and choose the best checkpoint for later inference. The experiments are run five times, and the average scores are reported. Additional models and experimental settings are provided in Appendix D.1. For inference, we perform gradient ascent as the warm-up phase for $T = 500$ iterations with the learning rate $\alpha \in [0.001, 0.01]$. The latent-based directed evolution involves $G = 10$ iterative processes, with the candidate number set to

$K \times B = 128$. In our implementation, we set the number of samples and beam size to $K = 128$ and $B = 1$, respectively. Finally, for the random exploration, we try multiple combinations of annealing factor $\delta = 0.1$ and exploration step size $\gamma \in [1.5, 6]$ and report the best outcomes.

Baseline Algorithms We compare our method against the following representative baselines: (1) Greedy search (**AdaLead**) [56], (2) Model-based reinforcement learning (**DyNA PPO**) [2], (3) Model-based adaptive sampling (**CbAS**) [10], (4) Covariance matrix adaptation evolution strategy (**CMA-ES**) [25] (5) Conservative model-based optimization (**COMs**) [61], (6) Proximal exploration (**PEX**) [49] (7) GFlowNet (**GFN-AL**) [29], and (8) Gibbs sampling with graph-based smoothing (**GGS**) [33]. To ensure precise evaluation, we re-execute and re-evaluate all baseline methods using the same oracle. For the implementation from (1) to (4), we employ the open-source implementation provided by [56]. Regarding other baseline methods, we utilize the codes released by their respective authors.

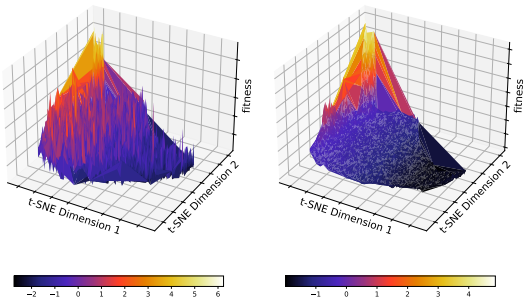
Oracles To ensure unbiased evaluation and avoid circular use of oracles, following [35], we use two separate oracles for each fitness dataset: (1) the optimization oracle that guides the model optimization and (2) the evaluation oracle that assesses the performance of the methods. Following [49], we freeze the ESM-based encoders and fine-tune an attention 1D decoder stacked after them to predict the fitness scores (see details in Appendix D.2). For fair comparisons, we only train the optimization oracle with the pre-trained 33-layer ESM-2 as the encoder, while using the pre-trained evaluation oracle provided by [49] to assess our method and other baselines.

Evaluation Metrics We use three metrics defined in [28] to evaluate our method and compare with other baselines: (1) **MFS**: maximum fitness score, (2) **Diversity**, (3) **Novelty**. Descriptions of these metrics can be found in Appendix E. It is crucial to emphasize that greater diversity and novelty do *not* equate to superior performance but offer insights into the exploration and exploitation trade-offs exhibited by different methods.

4.2 Results

Comparison with Baseline Algorithms As demonstrated in Table 1, our proposed LDE outperforms other algorithms in eight protein benchmarks. It is worth noting that AAV only contains sequences with up to 28 amino acids. For datasets with longer sequences exceeding 200 amino acids, using continuous latent representations demonstrably enhances LDE's optimization capabilities. This is particularly evident on the avGFP benchmark (comprising sequences of 237 amino acids), where it achieves a statistically significant outperformance over the baselines. This suggests that LDE's efficient exploration is particularly well-suited for tackling complex and longer protein sequences. In addition to the max fitness score, we also report the diversity and novelty metrics of the algorithms in Tables 5 and 6, respectively.

Effect of Gradient Ascent and Directed Evolution We conduct an ablation study to demonstrate how each component of our method contributes to the performance. In particular, we compare LDE with its variants, including (1) without gradient ascent (w/o GA): we do not use GA as the warm-up step for the latent-based DE and (2) without directed evolution (w/o DE): we only run gradient ascent to optimize fitness of the sequences. As shown by the shaded rows in Table 1, removing gradient ascent notably reduces the performance of the evolution algorithm. Similarly, performing gradient ascent without DE only leads to sub-optimal solutions.



(a) (b)

Figure 2: The fitness landscape approximated by regularized VAEs of E4B proteins. (a): Fitness scores are the ground truth provided by the dataset. (b): Fitness scores are predicted by the predictor f.

Visualizations of Latent Space Figures 2a and 2b illustrate the approximated fitness landscapes of protein sequences in the E4B family.

Table 2: Reported metrics of LDE with different latent sizes on two datasets. All experiments were conducted five times using learning rate $\alpha = 0.002$ and exploration step size $\gamma = 3$.

Dataset	Latent Dim	Fitness	Novelty	Diversity
E4B (L = 102)	128	4.433	2.040	5.322
	256	4.597	0.615	0.968
	320	**5.120**	2.038	4.504
	512	3.052	0.862	0.968
TEM (L = 286)	128	1.408	17.109	44.906
	256	1.220	267.013	1.783
	320	**1.745**	62.508	61.652
	512	-0.024	269.992	0.044

These visualizations imply that by applying the regularization to VAEs as outlined in Equation (3), the latent representations can be organized following their fitness levels. Furthermore, as illustrated in Figure 2a, our VAEs demonstrate the capability to generate meaningful latent representations when grounded in actual fitness values for labeling these vectors. In contrast, as depicted in Figure 2b, the jointly-trained predictor f effectively approximates a smooth fitness landscape by predicting fitness values of latent vectors in close alignment with their true values, thereby facilitating the gradient ascent step. These observations provide a rationale for the effectiveness of our proposed latent-based directed evolution, leveraging the expressiveness power of deep learning.

Effect of Latent Dimension To have a comprehensive understanding of latent-based DE, we study the effect of latent dimensions on the evolution process. Table 2 shows that a large latent dimension can lead to worse performance in optimizing protein fitness, while smaller latent sizes can result in sub-optimal fitness. This is an expected behavior in LSO where high dimensions may impose challenges for optimization algorithms and lower dimensions can not encode sufficient information about the input data. Furthermore, we observe that novelty and diversity are independent of good fitness and vary specifically for different fitness landscapes.

Active Learning Protein fitness datasets are often fragmented due to the cost of wet lab experiments [18], and even then, they only capture a small protein of real-world protein behavior. This limitation can trap machine learning models, which learn from training samples, in local optima, hindering their generalizability and accuracy [10]. To overcome this issue, in this work, we propose using active learning to update the fitness landscape approximated by our regularized VAEs. Indeed, Algorithm 2 demonstrates our method in detail.

In particular, we perform an outer active learning loop with N rounds to iteratively update the latent space, as well as the simulated landscape produced by the encoder ϕ and the latent fitness predictor f as mentioned in Section 3.2. For each round i, we fine-tune the VAE model M with the dataset $\mathcal{D}_{i-1}$. The fine-tuned model is then used in Algorithm 1 to explore sequences with higher fitness scores. In our study, we avoid the circular use of oracles by using the optimization oracle $\mathcal{O}(\cdot)$ during the optimization process to guide the exploration, and the evaluation oracle $\mathcal{E}(\cdot)$ is used to evaluate of the final population. We remove all duplicated samples in $\mathcal{P}$ and use them as training samples $\mathcal{D}_i$ for the next round in our algorithm.

To validate our proposed method, we conduct additional experiments on the four smallest benchmark datasets, each containing fewer than $8,000$ training data points: TEM, AMIE, LGK, and UBE2I. For this demonstration, we set the number of active learning rounds N to 10 and decrease the number of directed evolution iterations G to 5. Additionally, in each active learning loop, the regularized VAE M is fine-tuned in 30 epochs. As outlined in Table 3, LDE demonstrates improved performance when combined with active learning. These results empirically validate our hypothesis and confirm the efficacy of our proposed method.

Algorithm 2 Active Learning with Latent-Based DE

Input: a VAE $M = (\phi, \theta, f)$, training dataset $\mathcal{D}_t$, number of rounds N, optimization oracle $\mathcal{O}$, number of epochs n.

1: Train M on $\mathcal{D}_t$
2: $\mathcal{D}_0 \leftarrow \emptyset$
3: **for** $i = 1$ **to** N **do**
4: Run Algorithm 1 with oracle $\mathcal{O}$ and M to find population $\mathcal{P} = \{(x, \mathcal{O}(y)) | x \in \mathcal{V}^L, y \in \mathbb{R}\}$.
5: $\mathcal{D}_i \leftarrow \mathcal{D}_{i-1} \cup$ RemoveDuplicate($\mathcal{P}$)
6: Update M on the data $\mathcal{D}_i$ in n epochs
7: **end for**

Return P_G

Table 3: Max fitness scores on four smallest protein datasets.

Model	TEM	AMIE	LGK	UBE2I
LDE	1.095	-0.558	-0.005	2.976
− w/ active learning	**2.167**	**-0.015**	**0.022**	**3.698**

5 Conclusion and Future Work

In this work, we present Latent-based Directed Evolution (LDE), a novel method that combines directed evolution with gradient ascent in a regularized VAE latent space to efficiently optimize and design protein sequences. This approach leverages deep representation learning capabilities of generative models to significantly speed up the evolutionary process, achieving superior results compared to traditional methods solely operating in sequence space. LDE holds significant promise for accelerating protein engineering and drug discovery efforts, and we invite further research on integrating it with in vitro protein characterization for real-world validation.

On-going Direction However, our present work is not complete and has limitations, including its reliance on a single fitness predictor that could destabilize the optimization if poorly calibrated. To address this, we are planning to use ensembles of surrogate models with risk-aware strategies to handle uncertainty, along with robustness checks and sensitivity analyses to examine model stability and parameter impact. Previous studies [45, 65] have explored this issue and shown promising results. A potential extension would involve integrating structural information [64, 60] into the optimization process, ensuring that the generated structures closely resemble the wild-type to maintain functionality. Additionally, LDE could be adapted for multi-objective optimization. These ongoing efforts will enhance the robustness, applicability, and effectiveness of our method, connecting computational predictions with real-world biological outcomes.

References

[1] Ethan C. Alley, Grigory Khimulya, Surojit Biswas, Mohammed AlQuraishi, and George M. Church. Unified rational protein engineering with sequence-based deep representation learning. *Nature Methods*, 16(12):1315–1322, October 2019.

[2] Christof Angermueller, David Dohan, David Belanger, Ramya Deshpande, Kevin Murphy, and Lucy Colwell. Model-based reinforcement learning for biological sequence design. In *International Conference on Learning Representations*, 2020.

[3] Frances H. Arnold. Directed evolution: Creating biocatalysts for the future. *Chemical Engineering Science*, 51(23):5091–5102, 1996.

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

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

[6] Frances H Arnold and Alexander A Volkov. Directed evolution of biocatalysts. *Current opinion in chemical biology*, 3(1):54–59, 1999.

[7] Karl Johan Åström and Tore Hägglund. *Advanced PID control*. ISA-The Instrumentation, Systems and Automation Society, 2006.

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

[9] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. In Stefan Riezler and Yoav Goldberg, editors, *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany, August 2016. Association for Computational Linguistics.

[10] David Brookes, Hahnbeom Park, and Jennifer Listgarten. Conditioning by adaptive sampling for robust design. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 773–782. PMLR, 09–15 Jun 2019.

[11] David H Brookes, Amirali Aghazadeh, and Jennifer Listgarten. On the sparsity of fitness functions and implications for learning. *Proceedings of the National Academy of Sciences*, 119(1):e2109649118, 2022.

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

[13] Egbert Castro, Abhinav Godavarthi, Julian Rubinfien, Kevin Givechian, Dhananjay Bhaskar, and Smita Krishnaswamy. Transformer-based protein generation with regularized latent space optimization. *Nature Machine Intelligence*, 4:1–12, 09 2022.

[14] Tianlai Chen, Pranay Vure, Rishab Pulugurta, and Pranam Chatterjee. AMP-diffusion: Integrating latent diffusion with protein language models for antimicrobial peptide generation. In *NeurIPS 2023 Generative AI and Biology (GenBio) Workshop*, 2023.

[15] Xi Chen, Diederik P. Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational lossy autoencoder. In *International Conference on Learning Representations*, 2017.

[16] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics.

[17] Cyrus Chothia. Principles that determine the structure of proteins. *Annual Review of Biochemistry*, 53(1):537–572, June 1984.

[18] Christian Dallago, Jody Mou, Jody Mou, Kadina Johnston, Bruce Wittmann, Nicholas Bhattacharya, Samuel Goldman, Ali Madani, and Kevin Yang. Flip: Benchmark tasks in fitness landscape inference for proteins. In J. Vanschoren and S. Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1. Curran, 2021.

[19] Patrick Emami, Aidan Perreault, Jeffrey Law, David Biagioni, and Peter St. John. Plug and play directed evolution of proteins with gradient-based discrete mcmc. *Machine Learning: Science and Technology*, 4(2):025014, April 2023.

[20] Edgardo T Farinas, Thomas Bulter, and Frances H Arnold. Directed enzyme evolution. *Current opinion in biotechnology*, 12(6):545–551, 2001.

[21] Elad Firnberg, Jason W. Labonte, Jeffrey J. Gray, and Marc Ostermeier. A Comprehensive, High-Resolution Map of a Gene's Fitness Landscape. *Molecular Biology and Evolution*, 31(6):1581–1592, 02 2014.

[22] Richard J Fox, S Christopher Davis, Emily C Mundorff, Lisa M Newman, Vesna Gavrilovic, Steven K Ma, Loleta M Chung, Charlene Ching, Sarena Tam, Sheela Muley, John Grate, John Gruber, John C Whitman, Roger A Sheldon, and Gjalt W Huisman. Improving catalytic function by prosar-driven enzyme evolution. *Nature Biotechnology*, 25(3):338–344, February 2007.

[23] N Go. Theoretical studies of protein folding. *Annual Review of Biophysics and Bioengineering*, 12(1):183–210, June 1983.

[24] Rafael Gómez-Bombarelli, Jennifer N. Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D. Hirzel, Ryan P. Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science*, 4(2):268–276, 2018. PMID: 29532027.

[25] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

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

[27] Chloe Hsu, Hunter Nisonoff, Clara Fannjiang, and Jennifer Listgarten. Learning protein fitness models from evolutionary and assay-labeled data. *Nature Biotechnology*, 40(7):1114–1122, January 2022.

[28] 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. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 9786–9801. PMLR, 17–23 Jul 2022.

[29] Moksh Jain, Emmanuel Bengio, Alex Hernandez-Garcia, Jarrid Rector-Brooks, Bonaventure FP Dossou, Chanakya Ajit Ekbote, Jie Fu, Tianyu Zhang, Michael Kilgour, Dinghuai Zhang, et al. Biological sequence design with gflownets. In *International Conference on Machine Learning*, pages 9786–9801. PMLR, 2022.

[30] Kadina E Johnston, Clara Fannjiang, Bruce J Wittmann, Brian L Hie, Kevin K Yang, and Zachary Wu. Machine learning for protein engineering. *arXiv preprint arXiv:2305.16634*, 2023.

[31] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[32] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[33] Andrew Kirjner, Jason Yim, Raman Samusevich, Shahar Bracha, Tommi S Jaakkola, Regina Barzilay, and Ila R Fiete. Improving protein optimization with smoothed fitness landscapes. In *The Twelfth International Conference on Learning Representations*, 2023.

[34] Justin R. Klesmith, John-Paul Bacik, Ryszard Michalczyk, and Timothy A. Whitehead. Comprehensive sequence-flux mapping of a levoglucosan utilization pathway in e. coli. *ACS Synthetic Biology*, 4(11):1235–1243, September 2015.

[35] Sathvik Kolli, Amy X. Lu, Xinyang Geng, Aviral Kumar, and Sergey Levine. Data-driven optimization for protein design: Workflows, algorithms and metrics. In *ICLR2022 Machine Learning for Drug Discovery*, 2022.

[36] Aviral Kumar and Sergey Levine. Model inversion networks for model-based optimization. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 5126–5137. Curran Associates, Inc., 2020.

[37] H.A. Daniel Lagassé, Aikaterini Alexaki, Vijaya L. Simhadri, Nobuko H. Katagiri, Wojciech Jankowski, Zuben E. Sauna, and Chava Kimchi-Sarfaty. Recent advances in (therapeutic protein) drug development. *F1000Research*, 6:113, February 2017.

[38] Minji Lee, Luiz Felipe Vecchietti, Hyunkyu Jung, Hyunjoo Ro, Meeyoung Cha, and Ho Min Kim. Protein sequence design in a latent space via model-based reinforcement learning, 2023.

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

[40] Yunan Luo, Guangde Jiang, Tianhao Yu, Yang Liu, Lam Vo, Hantian Ding, Yufeng Su, Wesley Wei Qian, Huimin Zhao, and Jian Peng. Ecnet is an evolutionary context-integrated deep learning framework for protein engineering. *Nature Communications*, 12(1), September 2021.

[41] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In Lluís Màrquez, Chris Callison-Burch, and Jian Su, editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[42] John Maynard Smith. Natural selection and the concept of a protein space. *Nature*, 225(5232):563–564, February 1970.

[43] 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. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 29287–29303. Curran Associates, Inc., 2021.

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

[45] Pascal Notin, José Miguel Hernández-Lobato, and Yarin Gal. Improving black-box optimization in VAE latent space using decoder uncertainty. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[46] Yuchi Qiu, Jian Hu, and Guo-Wei Wei. Cluster learning-assisted directed evolution. *Nature Computational Science*, 1(12):809–818, December 2021.

[47] Yuchi Qiu and Guo-Wei Wei. Clade 2.0: Evolution-driven cluster learning-assisted directed evolution. *Journal of Chemical Information and Modeling*, 62(19):4629–4641, September 2022.

[48] Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Peter Chen, John Canny, Pieter Abbeel, and Yun Song. Evaluating protein transfer learning with tape. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[49] Zhizhou Ren, Jiahan Li, Fan Ding, Yuan Zhou, Jianzhu Ma, and Jian Peng. Proximal exploration for model-guided protein sequence design. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 18520–18536. PMLR, 17–23 Jul 2022.

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

[51] 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):e2016239118, 2021.

[52] 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):397–401, May 2016.

[53] Stanislau Semeniuta, Aliaksei Severyn, and Erhardt Barth. A hybrid convolutional variational autoencoder for text generation. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 627–637, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[54] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

[55] Huajie Shao, Shuochao Yao, Dachun Sun, Aston Zhang, Shengzhong Liu, Dongxin Liu, Jun Wang, and Tarek Abdelzaher. ControlVAE: Controllable variational autoencoder. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8655–8664. PMLR, 13–18 Jul 2020.

[56] Sam Sinai, Richard Wang, Alexander Whatley, Stewart Slocum, Elina Locane, and Eric D. Kelsic. Adalead: A simple and robust adaptive greedy search algorithm for sequence design. *CoRR*, abs/2010.02141, 2020.

[57] Zhenqiao Song and Lei Li. Importance weighted expectation-maximization for protein sequence design. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 32349–32364. PMLR, 23–29 Jul 2023.

[58] 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. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 20459–20478. PMLR, 17–23 Jul 2022.

[59] Lea M. Starita, Jonathan N. Pruneda, Russell S. Lo, Douglas M. Fowler, Helen J. Kim, Joseph B. Hiatt, Jay Shendure, Peter S. Brzovic, Stanley Fields, and Rachel E. Klevit. Activity-enhancing mutations in an e3 ubiquitin ligase identified by high-throughput mutagenesis. *Proceedings of the National Academy of Sciences*, 110(14):E1263–E1272, 2013.

[60] Jin Su, Chenchen Han, Yuyang Zhou, Junjie Shan, Xibin Zhou, and Fajie Yuan. Saprot: Protein language modeling with structure-aware vocabulary. In *The Twelfth International Conference on Learning Representations*, 2024.

[61] Brandon Trabucco, Aviral Kumar, Xinyang Geng, and Sergey Levine. Conservative objective models for effective offline model-based optimization. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 10358–10368. PMLR, 18–24 Jul 2021.

[62] Cara A Tracewell and Frances H Arnold. Directed enzyme evolution: climbing fitness peaks one amino acid at a time. *Current opinion in chemical biology*, 13(1):3–9, 2009.

[63] Thanh V. T. Tran and Truong Son Hy. Protein design by directed evolution guided by large language models. *bioRxiv*, 2024.

[64] Michel van Kempen, Stephanie S. Kim, Charlotte Tumescheit, Milot Mirdita, Jeongjae Lee, Cameron L. M. Gilchrist, Johannes Söding, and Martin Steinegger. Fast and accurate protein structure search with foldseek. *Nature Biotechnology*, 42(2):243–246, Feb 2024.

[65] Yanzheng Wang, TIANYU SHI, and Jie Fu. Sample-efficient antibody design through protein language model for risk-aware batch bayesian optimization. In *NeurIPS 2023 AI for Science Workshop*, 2023.

[66] Jochen Weile, Song Sun, Atina G Cote, Jennifer Knapp, Marta Verby, Joseph C Mellor, Yingzhou Wu, Carles Pons, Cassandra Wong, Natascha van Lieshout, Fan Yang, Murat Tasan, Guihong Tan, Shan Yang, Douglas M Fowler, Robert Nussbaum, Jesse D Bloom, Marc Vidal, David E Hill, Patrick Aloy, and Frederick P Roth. A framework for exhaustively mapping functional missense variants. *Molecular Systems Biology*, 13(12):957, 2017.

[67] Ronald J. Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, 1989.

[68] Bruce J. Wittmann, Yisong Yue, and Frances H. Arnold. Informed training set design enables efficient machine learning-assisted directed protein evolution. *Cell Systems*, 12(11):1026–1045.e7, November 2021.

[69] Emily E. Wrenbeck, Laura R. Azouz, and Timothy A. Whitehead. Single-mutation fitness landscapes for an enzyme on multiple substrates reveal specificity is globally encoded. *Nature Communications*, 8(1):15695, Jun 2017.

[70] S. Wright. The roles of mutation, inbreeding, crossbreeding and selection in evolution. *Proceedings of the XI International Congress of Genetics*, 8:209–222, 1932.

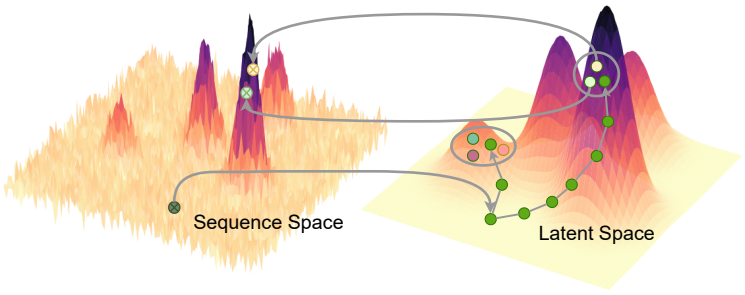


Figure 3: Optimizing protein fitness directly in sequence space is notoriously difficult due to its non-smooth and sparse landscape. We overcome this obstacle by performing directed evolution within the smooth and continuous latent space of a generative model. Our algorithm begins by identifying high-fitness regions through gradient ascent. Within these regions, we strategically sample a defined number of neighboring points (represented by gray circles) to create a diverse population for the evolutionary process. This population then undergoes iterative selection and mutation within the latent space, ultimately converging to sequences with enhanced fitness.

A Preliminaries

Directed Evolution Theory (DE) is a conventional approach in protein engineering that aims to search for global maximal protein sequences from a large database of unlabeled candidates $\mathcal{S}$ with minimal experimental validation [30]. DE employs an accelerated cycle of mutation and selection, which iteratively generates a pool of protein variants and selects those having improved desired properties as the next generation. In essence, directed evolution is a local exploration around the regions of high-fitness proteins in large sequence space. The algorithm typically starts with wild-type protein sequences and repeatedly mutates a relatively small number of their amino acids to obtain variants with enhanced properties. This is motivated by the observation of [42] showing that functional protein sequences are clustered into groups in a vast space of non-functional ones, making locally exploring around the seed sequences possible to find ones with improved functions. Indeed, we illustrate this phenomenon in the left part of Figure 3.

Latent Space Optimization (LSO) is a model-based optimization technique that performs in the latent space $\mathcal{Z}$ of deep generative models. In particular, an objective function $f : \mathcal{Z} \mapsto \mathbb{R}$ is trained to predict the objective values of data samples directly from their latent representations. When $\mathcal{Z}$ is a low-dimensional and continuous space, LSO acts as an efficient and effective optimization method as it overcomes the difficulties of optimizing in discrete and high-dimensional spaces by turning the problem into the continuous version, which is simpler to solve with a wide range of available techniques like gradient ascent and Bayesian optimization (BO) [54]. Additionally, f can be trained by using an encoder $\phi : \mathcal{X} \mapsto \mathcal{Z}$ that maps the input sample $x \in \mathcal{X}$ to its corresponding latent point $z \in \mathcal{Z}$.

B Variational Autoencoders

Variational autoencoders (VAEs) [32] is a class of generative model wherein each data sample x is generated by a generative distribution $p_\theta(x|z)$, parameterized by θ, that conditions on the unseen low dimensional latent variable $z \in \mathbb{R}^d$, which can be sampled from a prior distribution $p(z)$ (usually assumed as Gaussian). In other words, θ is trained to maximize the marginal log-likelihood $\log p_\theta(x) := \log \int_z p_\theta(x|z)p(z)dz$. However, this is intractable as all the configurations of the latent variable z must be examined during the optimization. To address this issue, [32] proposed to use amortized variational inference to approximate the true posterior distribution $p_\theta(z|x)$ via a variational distribution $q_\phi(z|x)$ with learnable parameters ϕ. They, instead, optimize the evidence of the lower bound (ELBO) of $\log p(x)$, which is written as:

$$\log p(x) \geq \mathbb{E}_{q_\phi(z|x)}(\log p_\theta(x|z)) - \beta D_{\mathrm{KL}}(q_\phi(z|x)||p(z)). \tag{6}$$

From the perspective of autoencoding, ϕ and θ are regarded as an encoder (inference network) and decoder (generative network), respectively. The encoder ϕ maps the high-dimensional input variable x to the low-dimensional latent variable z within the latent space. In particular, the mean

$\mu_\phi(x)$ and log-variance $\log \sigma_\phi(x)$ of the posterior $q_\phi(z|x)$ are computed, and z is sampled from $q_\phi(z|x) \sim \mathcal{N}(\mu_\phi(x), \sigma_\phi(x)^2)$. Finally, the decoder θ maps z back to the input space.

Popular VAE applications often involve a trade-off between reconstruction accuracy and some other application-specific goals (e.g., to produce *new and original* candidates), effectively manipulated through KL-divergence. ControlVAE [55] combines control theory [7] with the basic VAE to stabilize the KL-divergence to a desired value. It designs a PI controller to dynamically tune the weight β in the Equation (6) by using the actual KL-divergence as feedback during training as follows:

$$\beta(t) = K_p \sigma(-e(t)) - K_i \sum_{j=0}^{t} e(j) + \beta_{\min}, \tag{7}$$

where $e(t) = C - D_{\text{KL}}(q_\phi(z|x)||p(z))(t)$, which is the difference between desired KL-divergence C and the actual one at training step t; $\sigma(\cdot)$ is a sigmoid function; $\beta_{\min}$ is a constant; K_p and K_i are positive co-efficients for the P term and I term respectively.

C Task Details

In this section, we present how the eight datasets in this study are collected and processed. Across all datasets, we filter out sequences containing special characters and those lacking corresponding fitness scores. Specifically for the AAV dataset, only sequences with a length of 28 are retained. Detailed statistical information, including protein sequence length, dataset size, and percentile distribution, is provided in Table 4. We submit all the pre-processed data and training code in the supplementary materials.

Table 4: Detailed information and statistics of the eight protein datasets.

Dataset	Organism	Protein	Optimization Target	Length	Size	Percentiles		
						0.25	0.50	0.75
avGFP [52]	Aequorea victoria	GFP	Brightness	237	51,715	1.428	3.287	3.161
AAV [12]	Homo sapiens	VP1	AAV viabilities	28	42,330	−3.964	−0.840	1.321
TEM [21]	Escherichia coli	TEM-1 β-Lactamase	Thermodynamic stability	286	5,199	0.049	0.444	0.934
E4B [59]	Mus musculus	UBE4B	Ubiquitin ligase activity	102	91,032	−1.830	−0.984	−0.093
AMIE [69]	Escherichia coli	Amidase	Hydrolysis activity	341	6,417	−1.228	−0.666	−0.263
LGK [34]	Lipomyces starkeyi	Levoglucosan kinase	Levoglucosan utilization	439	7,633	−0.871	−0.562	−0.394
Pab1 [44]	Saccharomyces cerevisiae	Poly(A)-binding	mRNA binding	75	36,389	−0.116	−0.022	0.036
UBE2I [66]	Homo sapiens	UBE2I	Growth rescue rate	159	3,022	0.068	0.492	0.766

- **Green Fluorescent Protein (avGFP):** Derived from Aequorea victoria, Green Fluorescent proteins (GFPs) are capable of manifesting vivid green fluorescence upon exposure to light within the blue to ultraviolet spectrum. These proteins are commonly employed as biosensors for detecting gene expressions and protein locations. The goal is to optimize sequences with higher log-fluorescence intensity values. We collect data following [52]
- **Adeno-Associated Viruses (AAV):** The engineer of a 28-amino acid segment (position 561–588) within the VP1 protein, situated in the capsid of the Adeno-associated virus, has garnered significant interest in the realm of machine learning-guided design. The target is to generate more capable sequences as gene delivery vectors measured by AAV viabilities. We collect data following [12].
- **TEM-1 β-Lactamase (TEM):** The investigation of the TEM-1 β-Lactamase protein's resistance to penicillin antibiotics in E. coli is a subject of extensive scrutiny, aiming to comprehend mutational impacts and the associated fitness landscape. The objective is to propose high thermodynamic-stable sequences upon wild-type TEM-1. We gather data following [21].
- **Ubiquitination Factor Ube4b (E4B):** The ubiquitination factor Ube4b plays a pivotal role in cellular waste degradation through interactions with ubiquitin and other proteins. The aim is to design sequences with higher enzyme activity. We merge data following [59].
- **Aliphatic Amide Hydrolase (AMIE):** The enzyme encoded by amiE, known as Amidase, holds industrial relevance and is derived from Pseudomonas aeruginosa. The goal is to optimize amidase sequences with greater enzyme activity. We merge data following [69].

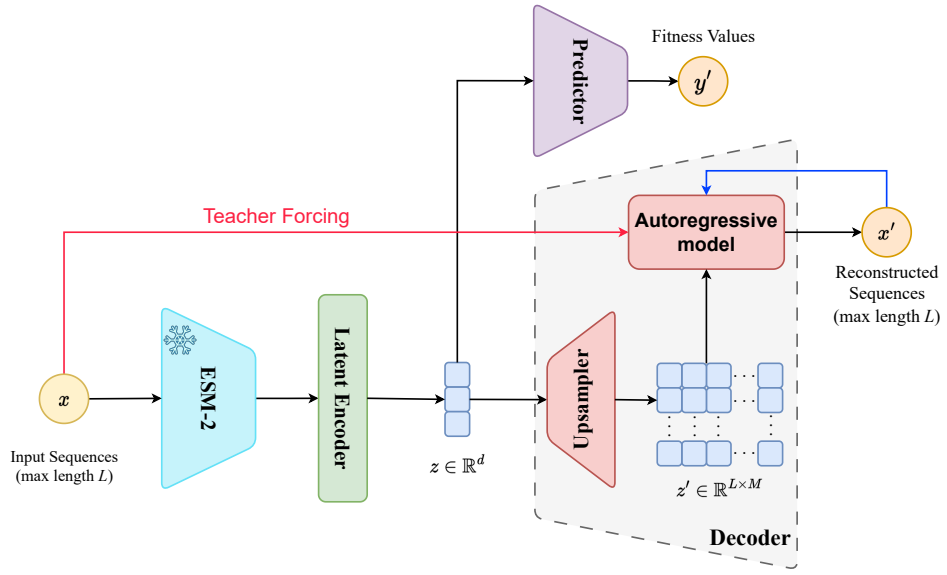


Figure 4: Schematic illustration of the VAE model used in the study.

- **Levoglucosan Kinase (LGK):** Levoglucosan kinase catalyzes the conversion of Levoglucosan (LG) to the glycolytic intermediate glucose-6-phosphate through an ATP-dependent reaction. The target is to optimize LGK protein sequences with improved enzyme activity. We gather data following [34].
- **Poly(A)-binding Protein (Pab1):** Pab1 utilizes the RNA recognition motif (RRM) to bind to multiple adenosine monophosphates (poly-A). The aim is to design sequences with higher binding fitness to multiple adenosine monophosphates. We collect data following [44].
- **SUMO E2 Conjugase (UBE2I):** The utilization of variants for the functional mapping of human genomes holds considerable significance in both scientific research and clinical treatment. The goal is to identify human SUMO E2 conjugase with a higher growth rescue rate. Data are obtained following [66].

D Implementation Details

D.1 VAEs' Architecture

In this section, we go into detail regarding the architecture of the VAE used in our study. As mentioned in Section 3.2, our regularized VAE consists of an encoder, a predictor, and a decoder.

Encoder Figure 4 depicts that the encoder incorporates a pre-trained ESM-2 [39] followed by a latent encoder to compute the latent representation z. In our study, we leverage the powerful representation of the pre-trained 30-layer ESM-2 [39] by making it the encoder of our model. Given an input sequence $x = \langle x_0, x_1, \cdots, x_L \rangle$, where $x_i \in \mathcal{V}$, the transformer-based ESM-2 computes representations for each token x_i in x, resulting in a token-level hidden representation $H = \langle h_0, h_1, \cdots, h_L \rangle, h_i \in \mathbb{R}^{d_h}$. We calculate the global representation $h \in \mathbb{R}^{d_h}$ of x via a weighted sum of its tokens:

$$h = \sum_{i=1}^{L} \frac{\omega^T \exp(h_i)}{\sum_{i=1}^{L} \omega^T \exp(h_i)} h_i. \tag{8}$$

here, ω is a learnable global attention vector. Then, two multi-layer perceptrons (MLPs) are used to compute $\mu = \text{MLP}_1(h)$ and $\log \sigma = \text{MLP}_2(h)$, where the latent dimension is d. Finally, a latent representation $z \in \mathbb{R}^d$ is sampled from $\mathcal{N}(\mu, \sigma^2)$, which is further proceeded to the decoder to reconstruct the sequence $\hat{x}$. We use an auxiliary MLP as a **fitness predictor** that maps the latent z to the fitness score, i.e. $y' = \text{MLP}(z)$. The hidden dimensional size of the predictor is set to 512, with the dropout of 0.2. We set $d_h = 1280$ and $d = 320$ for the main experiments in our study.

Decoder Inspired from [53], we construct our decoder as the combination of two components: an 'upsampler' component comprising 3 layers of transposed convolutions with stride of 2 to upsample the latent vector z to a sequence matching the output sequence's length; and an autoregressive component consisting of an attention-based GRU [16, 41] with 512 units. To cope with the optimization difficulties reported when training VAE with powerful autoregressive decoders [9, 53], we follow [9] by applying 40% dropout to the amino acid context supplied as input to the GRU during training. This encourages the network to depend on the information conveyed through the upsampled latent code along with the conditional information in the masked amino acid sequence to make predictions. Additionally, we apply teacher forcing [67] with a ratio of 50% for faster convergence.

D.2 Oracles

We establish the optimization oracle $\mathcal{O}(\cdot)$, utilized for latent-based directed evolution, by leveraging features generated by the pre-trained 33-layer ESM-2 [39] with a dimension of 1280. Subsequently, we fine-tune an Attention1D model to predict fitness values based on these representations. As for the evaluation oracle $\mathcal{E}(\cdot)$, which acts as the "ground-truth" evaluator, we employ the trained oracle provided by [49]. This evaluation oracle combines the pre-trained ESM-1b [51] with a Attention1D model with a dimension of 512, and it is used to assess all methods.

D.3 Training Configurations

As mentioned in the Appendix B, we employ the ControlVAE mechanism to prevent KL vanishing and enhance the diversity of generated data during training. Across all tasks, we configure the coefficients K_p and K_i of the P term and I term to 0.01 and 0.0001, respectively. For the LGK benchmark, the desired KL-divergence C is set to 40, while for all other tasks, $C = 20$ is utilized. The batch size for each task is determined to be as large as possible, as long as the total steps in one epoch for each task are higher than 100.

E Evaluation Metrics

We provide mathematical definitions of four metrics: maximum fitness score (MFS), diversity, and novelty. Let $\mathcal{P} = \{p_1, p_2, \dots p_N\}$ be the population generated by LDE and $\mathcal{E}(\cdot)$ be the evaluation oracle, we define:

- **MFS** = $\max(\{\mathcal{E}(p_i)\}_{i=1}^{N})$,
- **Diversity** = $\dfrac{\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N} d(p_i, p_j)}{N(N-1)}$,
- **Novelty** = $\dfrac{\sum_{i=1}^{N} \min_{s_j \in \mathcal{D}} d(p_i, s_j)}{N}$,

where $d(\cdot, \cdot)$ is the Levenshtein distance, and $\mathcal{D}$ is the initial dataset (i.e., training dataset).

F Additional Results

F.1 Comparison with other Baselines

This section extends Section 4.2 by reporting results for two metrics: diversity and novelty in Tables 5 and 6, respectively. Again, it is important to note that while these metrics offer valuable insights into the exploration and exploitation trade-offs exhibited by different methods, they do *not* determine the efficacy of a method.

F.2 Differences between optimization oracle and evaluation oracle

In Table 7, we present a comparison of results for designs using the optimization oracle $\mathcal{O}(\cdot)$ (Opt. Oracle) and the evaluation oracle $\mathcal{E}(\cdot)$ (Eval. Oracle). The table demonstrates that, with the exception of Pab1, all other benchmarks experienced a decrease in performance when using the evaluation

Models	avGFP	AAV	TEM	E4B	AMIE	LGK	Pab1	UBE2I	Average
AdaLead	9.814	6.904	8.071	7.412	6.898	7.430	7.441	7.439	7.676
DyNA PPO	204.376	114.229	157.964	140.617	171.123	205.098	185.145	179.212	169.721
CbAS	205.709	115.437	159.502	142.033	172.526	206.823	186.766	180.769	171.196
CMA-ES	173.365	96.882	134.633	119.388	145.151	174.274	157.098	152.058	144.106
COMs	70.319	45.761	73.191	68.398	100.731	126.623	115.242	109.650	88.739
PEX	7.048	4.782	6.692	6.625	6.388	7.031	7.012	7.219	6.600
LDE (ours)	94.646	1.604	61.652	4.504	35.762	108.053	9.040	14.666	41.806

Table 5: Diversity on eight protein datasets

Models	avGFP	AAV	TEM	E4B	AMIE	LGK	Pab1	UBE2I	Average
AdaLead	13.486	17.805	41.405	48.934	41.167	78.868	73.526	78.548	47.967
DyNA PPO	201.702	111.566	156.784	139.227	170.368	205.815	185.686	179.801	168.869
CbAS	201.825	111.549	156.845	139.172	170.031	205.404	185.354	179.549	168.716
CMA-ES	202.155	111.467	156.968	139.126	157.701	193.991	175.414	170.746	163.446
COMs	184.177	98.831	111.931	101.930	123.590	150.657	134.298	129.795	129.401
PEX	4.323	1.930	4.461	4.748	3.031	8.614	4.356	4.779	4.530
LDE (ours)	91.863	0.657	62.508	2.037	10.423	65.145	2.875	126.495	45.250

Table 6: Novelty on eight protein datasets

oracle. This decline is expected, as different oracle architectures result in different approximate fitness scores. Therefore, as we optimize the results based on the optimization oracle and solely utilize the evaluation oracle to assess the final performance of the method, the score produced by the optimization oracle should be higher.

	avGFP	AAV	TEM	E4B	AMIE	LGK	Pab1	UBE2I
Opt. Oracle	15.266	2.736	5.986	5.594	0.327	0.939	0.786	7.405
Eval. Oracle	8.058	2.636	1.745	5.120	-0.103	0.018	1.548	4.297

Table 7: Comparison of optimization and evaluation oracles on the max fitness scores across eight protein benchmarks.

F.3 Autoregressive vs. Non-autoregressive

In addition to the autoregressive decoder utilized in LDE, we report the performance of a non-autoregressive decoder implemented following the architecture proposed by [13]. This decoder comprises four 1-dimensional convolutional layers with ReLU activations and batch normalization layers are incorporated between convolutional layers, except for the final layer. As outlined in Table 8, it is observed that the autoregressive decoder consistently outperforms the non-autoregressive decoder across all tasks. This paves the way for further study on how different types of decoders affect the latent-based evolutionary algorithms.

F.4 Active Learning

Table 8: Maximum fitness scores on eight protein datasets of two decoder versions of LDE.

	avGFP	AAV	TEM	E4B	AMIE	LGK	Pab1	UBE2I	Average
Non-autoregressive LDE	3.733	1.368	1.095	3.123	-0.558	-0.005	0.089	2.976	1.430
(Autoregressive) LDE	**8.058**	**2.636**	**1.745**	**5.120**	**-0.103**	**0.018**	**1.548**	**4.297**	**3.204**