Check for updates

Cluster learning-assisted directed evolution

Yuchi Qiu [1], Jian Hu[2,3] and Guo-Wei Wei [1,3,4 ✉]

Directed evolution, a strategy for protein engineering, optimizes protein properties (that is, fitness) by expensive and time-consuming screening or selection of a large mutational sequence space. Machine learning-assisted directed evolution (MLDE), which screens sequence properties in silico, can accelerate the optimization and reduce the experimental burden. This work introduces an MLDE framework, cluster learning-assisted directed evolution (CLADE), which combines hierarchical unsupervised clustering sampling and supervised learning to guide protein engineering. The clustering sampling selectively picks and screens variants in targeted subspaces, which guides the subsequent generation of diverse training sets. In the last stage, accurate predictions via supervised learning models improve the final outcomes. By sequentially screening 480 sequences out of 160,000 in a four-site combinatorial library with five equal experimental batches, CLADE achieves global maximal fitness hit rates of up to 91.0% and 34.0% for the GB1 and PhoQ datasets, respectively, improved from the values of 18.6% and 7.2% obtained by random sampling-based MLDE.

irected evolution (DE) is a protein-engineering approach that is used to improve a particular property (for example, fitness) of a target protein by mimicking the process of natural selection[1]. The evaluation of fitness is expensive and time-consuming, especially when high-throughput selection or screening is not available. The fitness landscape is a high-dimensional surface that maps amino-acid sequences to properties such as activity, selectivity, stability and other physicochemical features. The goal of DE is to find the global maximal sequence using minimal experimental resources in an unlabeled candidate sequence library, $\mathcal{S}$:

$$x^* = \arg\max_{x \in \mathcal{S}} f(x), \tag{1}$$

where x is a sequence and $f(x)$ is an unknown sequence-to-fitness map. DE is one type of black-box optimization problem that sequentially queries sequences for experimental screening. Greedy search is effective at finding improved sequences with minimal experiments, but it is generally restricted to exploring local optima due to the prevalent epistasis in the fitness landscape[2–4]. On the other hand, random exploration via multi-site-saturation mutagenesis is inevitably associated with a huge combinatorial library, which often overwhelms the screening capacity[5]. An effective searching strategy for the epistatic landscape with minimal experimental burdens is highly desirable.

The last decade has witnessed the rapid development of machine learning (ML) (including deep learning, DL) algorithms for biological data[6–10]. Supervised models can learn relationships between proteins and fitness, and provide quantitative predictions of enzyme activity and selectivity[3], protein thermostability[11], protein folding energy[12,13], protein solubility[14], protein–ligand binding affinity[15] and protein–protein binding affinity[16]. Owing to the high cost of acquiring supervised labels, self-supervised protein embedding has emerged as an important paradigm in protein modeling. Trained on vast unlabeled sequence data resulting from natural evolution, self-supervised protein embedding can capture the substantial latent biological information of sequences and pass the information to the downstream supervised task[17,18]. Adapted from natural language processing, many model architectures (such as variational

autoencoders[19], recurrent neural networks[20,21] and transformers[22]) have been used to train the protein embedding models[17]. On the other hand, unsupervised clustering methods can identify the internal characteristics of unlabeled data by dividing them into multiple subspaces. Clustering methods, including distance-based clustering[23,24], community-based clustering[25], density-based clustering[26] and graph-based clustering[27,28], have been widely applied to transcriptomic data analysis[29], pattern recognition[30] and image processing[31] to reveal data heterogeneity.

Machine learning-assisted directed evolution (MLDE) is a new strategy for protein engineering that can be applied to a range of biological systems, such as enzyme evolution[3,32], engineering of fluorescence proteins[33], the localization of membrane proteins[34], protein thermostability optimization[35] and therapeutic antibody optimization[36]. Active learning is a popular approach in MLDE, where sequential selections of sequences are decided by the combination of a surrogate model and an acquisition function. The former is used to learn the sequence-to-fitness map from labeled data and the latter utilizes the predictions from the surrogate model to prioritize a set of sequences to be screened at the next round of experiments[37]. The acquisition function needs to balance the exploration–exploitation trade-off[38,39]. Uncertainty surrogate models such as the Gaussian process (GP) have been widely applied in MLDE[33–35]. Rather than making use of sequential iterations in experiments, focused training of the MLDE method was proposed to minimize the experimental burden to only two iterations[2]. This utilizes unsupervised zero-shot predictors[19,22,40,41] to predict fitness without experiments, and is used to restrict the training set selection within a small informative subset. The downstream supervised learning model performs a greedy search to optimize protein fitness. With this approach, state-of-the-art results were achieved.

In this Article we propose a cluster learning-assisted directed evolution (CLADE) framework to guide protein engineering. The CLADE framework introduces an unsupervised clustering strategy to supervised learning to preselect the training sets and virtually navigate the fitness landscape. Through the unsupervised clustering, the fitness heterogeneity can be identified where clusters have substantially different populations of high-fitness variants.

[1]Department of Mathematics, Michigan State University, East Lansing, MI, USA. [2]Department of Chemistry, Michigan State University, East Lansing, MI, USA. [3]Department of Biochemistry and Molecular Biology, Michigan State University, East Lansing, MI, USA. [4]Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI, USA. ✉e-mail: weig@msu.edu

By exploiting the fitness heterogeneity, we identify and sample the clusters enriched with high-fitness variants through sequential iterations with experimental screening. By introducing a hierarchical clustering, CLADE makes the random-sampling-based MLDE more accurate and robust. CLADE is a two-stage strategy in which the first-stage clustering sampling can improve the sampling efficacy by selectively exploring critical subspaces and the second-stage greedy search using the ensemble regressor has advantages over the conventional GP in MLDE. CLADE shows further improvements by coupling with zero-shot predictors. On sequentially screening a total of 480 sequences in five equal batches, CLADE successfully identified a global maximum with frequency of 91% and 34% for the benchmark datasets GB1 and PhoQ, respectively. This general CLADE framework provides improvement over state-of-the-art methods, suggesting it is an accurate, robust and efficient framework for protein engineering.

Results

Overview of CLADE. The CLADE framework is a two-stage procedure consisting of three components: experimental screening, unsupervised clustering and supervised learning. Unsupervised clustering sampling complements supervised learning to guide experimental screening to discover variants with optimal fitness in DE (Fig. 1a). Before CLADE analysis, a target protein and an unlabeled candidate mutant sequence library, $\mathcal{S}$, need to be constructed by expert selection (Fig. 1b). The unknown specific fitness information can be determined through experimental screening, but usually only a small subset of variants is screened because of experimental constraints. Although specific fitness information is largely unknown, sequence encoding methods can reveal general biological information for all variants in the library (Fig. 1b). At the first stage of CLADE, unsupervised clustering guides coarse exploration and selection over clusters. Encoded with general biological information, unsupervised clustering divides the sequence library into multiple clusters with different internal characteristics. Variants in the same cluster have similar general biological properties as well as fitness properties of interest, although their values are unknown. Instead of global sampling over the entire sequence library, CLADE performs a clustering sampling. To select one variant, one cluster is first selected according to the predefined cluster-wise sampling probabilities (clusters containing more high-fitness variants have higher probabilities to be selected). A sampling method is then employed to select a variant within this cluster. Random sampling is the simplest sampling method for the in-cluster sampling, but arbitrary sampling methods such as ϵ-greedy, Thompson and upper confidence bounds (UCBs) can also be implemented easily with CLADE. The selected variants are experimentally screened to obtain their fitness values. The clustering sampling iteratively selects variants and updates both the cluster-wise and in-cluster sampling strategies. The second stage of CLADE takes the labeled sample set as training data to train a supervised learning model and provides predictions of the rest of the sequence library. Greedy search is used in this stage, where the top predicted variants are screened by experiments. Optimal variants can be picked from all experimentally measured variants (Fig. 1c). In this process, the same sequence encoding method (that is, general biological information) is used for both clustering and supervised learning.

In clustering sampling, cluster-wise sampling probabilities are dynamically updated after each batch of variants is screened (Fig. 1d). In the first few batches, all clusters are selected uniformly to obtain a coverage of all clusters. The sampling strategy then tends to explore the high-fitness clusters. The sampling probability for each cluster is defined by the average fitness of selected variants in this cluster normalized by the summation of the average fitness of selected variants in each cluster (Methods). To further explore the high-fitness clusters, we propose a deep hierarchical clustering

structure (Fig. 1d). Clusters with higher average fitness are divided into more subclusters, then the same sampling procedure is applied to clusters at the new hierarchy. For maximum hierarchy N, the increment of clusters at hierarchy i, $K_i (i \leq N)$, needs to be defined before the simulation (Methods). Three examples of simulated sampling using random in-cluster sampling are presented to further illustrate the sampling process (Supplementary Section 3 and Supplementary Fig. 1).

In experimental screening, a batch of variants is usually screened in parallel and the batch size varies in systems with different throughputs. To adopt CLADE in systems with different throughputs, the frequency for updating the sampling probability or generating clusters at a new hierarchy needs to be multiples of the batch size, as well as the number of training data and the number of top-predicted variants being screened. In this work we take batch sizes of 96 and 1 to simulate medium-throughput and low-throughput systems (Methods). The outcome of CLADE consists of variants in the training data and the top 96 predicted variants. The max fitness and mean fitness are used to evaluate the CLADE outcome. Another metric, the global maximal fitness hit rate, measures the frequency with which CLADE successfully picks the global maximal variant in training data, top predictions or their union. Details and more metrics are provided in the Methods.

To test the performance of CLADE, the popular benchmark GB1 library was first used, then the PhoQ library, which was used previously in an early MLDE study[42]. Although both datasets provide suitable fitness for the CLADE algorithm, the PhoQ dataset may be limited because its fitness may only weakly correlate to a meaningful protein property (Datasets).

Revealing fitness heterogeneity with unsupervised clustering. We describe how unsupervised clustering assists the selection of training data. As a proof of principle we employed K-means clustering and four physicochemical descriptors based on amino-acid (AA) encoding, a subset of amino acid index dataset (AAindex) (Methods), as the sequence encoding method on the GB1 dataset, where the fitness is the binding affinity to an antibody (Datasets). We first divided the fitness landscape into $K_1 = 3$ clusters. The three clusters contain a similar number of variants and are well separated in the projected principal components space. The population of high-fitness variants (>0.3) is rare in the fitness landscape. Interestingly, we found heterogeneity of high-fitness variants in these clusters, with cluster 3 containing over 11-fold more high-fitness variants (that is, 911 variants) than either cluster 1 (80 variants) or cluster 2 (59 variants) (Fig. 2a).

Next, we performed K-means clustering with various numbers of clusters K_1 (10, 40 and 100), and multiple independent repeats were performed for each K_1 value. In a single simulation, clusters were given a unique cluster ID, where cluster ID indicates the descending ranking of the average fitness for all variants within the corresponding cluster. The expected average fitness for clusters with identical cluster IDs in multiple repeats was calculated (Fig. 2b). The distribution of cluster average fitness reveals the fitness heterogeneity, where the cluster with lower numbering has higher average fitness (Fig. 2b). We found that the distribution of cluster average fitness becomes more polarized near the origin as K_1 increases. Specifically, 32%, 52% and 67% of high-fitness variants (that is, >0.3) are contained in the top 10% clusters for K_1 values at 10, 40 and 100, respectively (Fig. 2b).

The clustering sampling is then able to oversample the high-fitness clusters with the identified heterogeneity. In sampled data, distributions of the expected cluster average fitness recapitulated the polarized distributions revealed by the ground-truth fitness, and the distributions become more polarized as K_1 increases (Fig. 2b). Indeed, K-means can capture the fitness heterogeneity, and our clustering sampling can recapitulate this heterogeneity
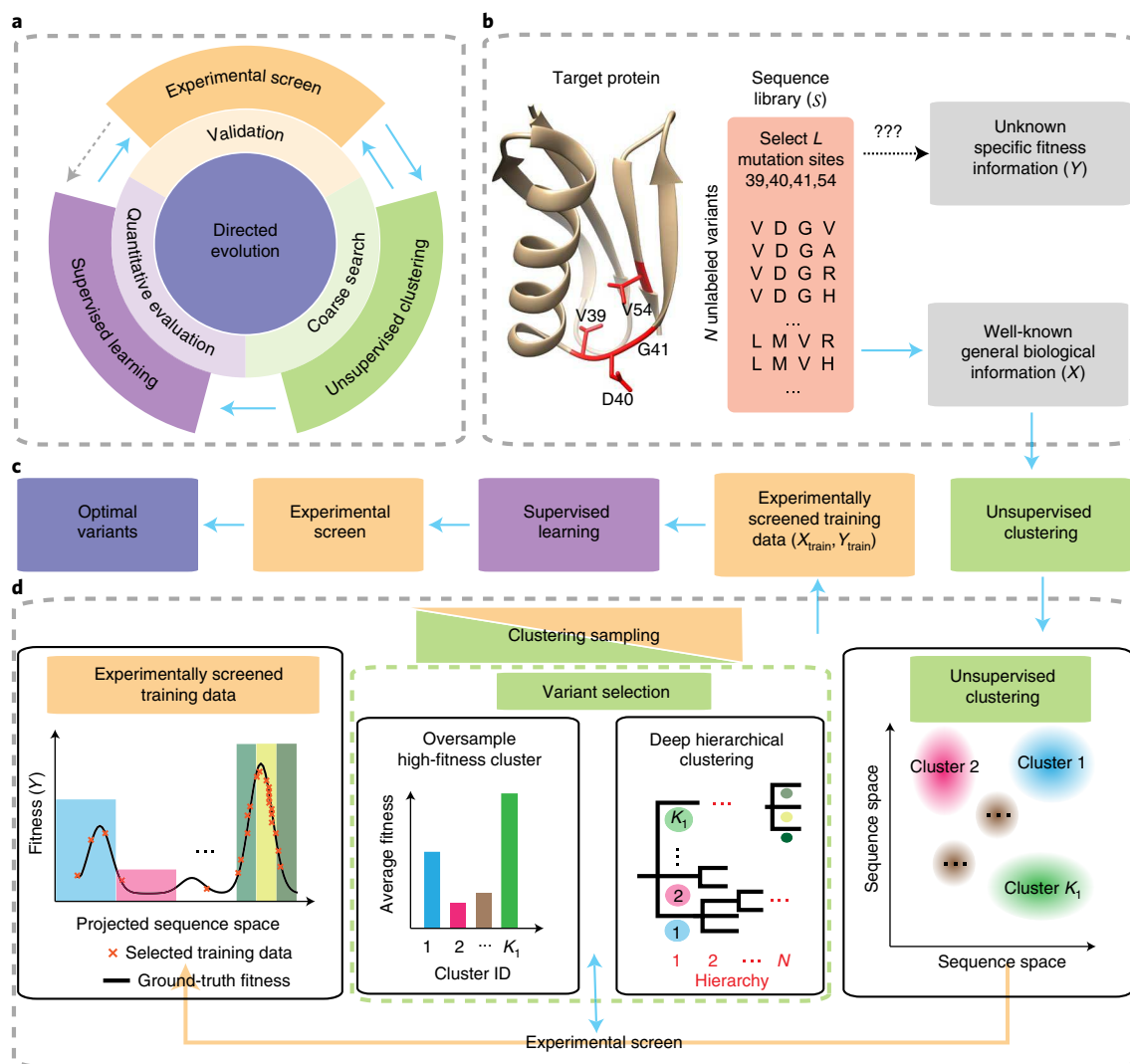
Fig. 1 | Overview of CLADE. a, Conceptual diagram of CLADE. CLADE consists of three components: experimental screening, unsupervised clustering and supervised learning. Blue arrows illustrate the flow of information. The gray dashed arrow shows the flow of information not considered in this work. **b**, Sequence library construction, showing a combinatorial library for site-saturation mutagenesis for the GB1 protein (PDB 2GI9). The $L = 4$ mutation sites are V39, D40, G41 and V54. Well-known general biological information encodes the library into a feature matrix X. The specific fitness information is determined by experimental screening, but usually only a small subset of variants can be screened with limited experimental capacity. **c**, Flowchart of CLADE. Unsupervised clustering divides the combinatorial library into multiple clusters by using the feature matrix X. Clustering sampling selects and screen variants to construct a labeled sample set through iterations between the experimental screening and unsupervised clustering. The labeled sample set is taken as training data passing to the supervised learning. Supervised learning learns from the training data, and predicts and prioritizes optimal variants for screening. **d**, Schematic of clustering sampling. Through iterations with experimental screening, clusters with high average fitness tend to be oversampled with higher probabilities. Deep hierarchical clustering divides the high-fitness clusters into more clusters at a new hierarchy to further oversample these clusters.

to select more samples with high fitness. A community-based clustering method, Louvain clustering[25], also successfully captured the fitness heterogeneity (Supplementary Section 6 and Supplementary Fig. 2).

Improving CLADE outcome with deep hierarchical structure. Utilizing the fitness heterogeneity, CLADE performed differently under different clustering architectures. First, we explored the maximum hierarchy, N, for CLADE. Random in-cluster sampling and simulated medium-throughput systems were employed. The GB1 dataset was used and encoded by AA encoding. For shallow hierarchy $N = 1$, CLADE using K-means improves over random-sampling-based MLDE on all evaluated metrics, including expected max fitness, expected mean fitness, global maximal fitness hit rate,

normalized discounted cumulative gain (NDCG), cross-validation errors and testing errors (Supplementary Data 1). In particular, the global maximal fitness hit rate can reach 40.2% when $K_1 = 90$, a 2.2-fold improvement over the random-sampling-based MLDE (Table 1). Similarly, CLADE using Louvain clustering can lead to an almost twofold improvement in global maximal fitness hit rate (36.4%, Table 1). For clustering with deep hierarchy, the number of variants in a cluster decreases quickly with its hierarchy. To ensure that any cluster has enough variants for partition at the next hierarchy, cluster increments (K_1, K_2, K_3 and so on) were explored in smaller ranges for deep hierarchy. CLADE performance was further improved with deeper hierarchy (Supplementary Data 1). A 2.7-fold improvement of the global maximal fitness hit rate (50.8%) was observed for both $N = 2$ and $N = 3$ (Table 1). Moreover, the
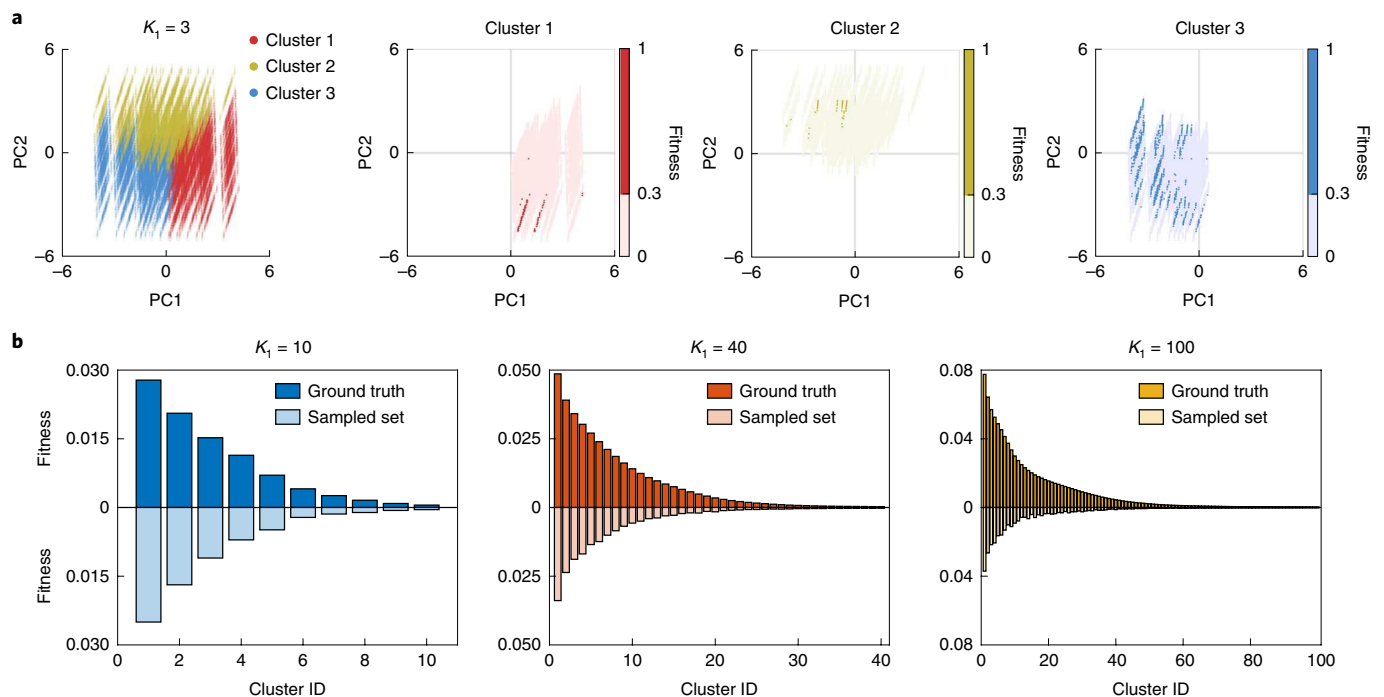
Fig. 2 | Fitness heterogeneity revealed by K-means clustering. a, Visualization of GB1 variants in the reduced two-dimensional space spanned by the first two principal components (PC1 and PC2). Three clusters were obtained from K-means clustering. AA encoding was employed. Dots with different colors represent variants in different clusters. Individual clusters were plotted in each subplot from the second to the fourth subplot, where variants with fitness lower or higher than 0.3 are denoted by light or dark colors. **b**, K-means clustering and the follow-up clustering sampling on the GB1 dataset with 500 independent repeats. Three sets of parameters are presented individually in different plots: $K_1 = 10$ (blue), 40 (red) and 100 (yellow). In a single simulation, each cluster is numbered by a unique cluster ID, where cluster ID indicates the descending ranking of the average fitness for all variants within the corresponding cluster. Then, clusters with identical cluster IDs in multiple repeats were used to calculate the expected average fitness shown in the bar plots. Bar plots above the abscissa (dark color) show the expected average ground-truth fitness for all variants contained in each cluster. Bar plots below the abscissa (light color) show the expected average fitness for variants selected from the clustering sampling in each cluste.

Table 1 | CLADE performance with various clustering hierarchies

Dataset	Hierarchy	Expected max fitness	Expected mean fitness	Global maximal fitness hit rate	Notes
GB1	$N=0$ (random sampling)	0.774	0.305	18.6%	–
GB1	$N=1$	0.871	0.406	40.2%	–
GB1	$N=1$	0.848	0.357	36.4%	Louvain clustering
GB1	$N=2$	0.890	0.421	50.8%	–
GB1	$N=3$	**0.893**	**0.423**	**50.8%**	–
GB1	$N=3$	0.909	0.431	55.6%	Low throughput
PhoQ	$N=0$ (random sampling)	0.387	0.077	7.2%	–
PhoQ	$N=3$	**0.547**	**0.096**	**20.6%**	–

CLADE performance on the GB1 and PhoQ datasets is shown using different hierarchies, N, for clustering sampling. $N=0$ indicates random sampling-based MLDE, following the procedure in ref. [2]. The sampling strategy for the selected clusters was random sampling. The case with the highest expected max fitness for each architecture is shown. Hyperparameters are extensively explored in Supplementary Data 1. Unless explicitly indicated, the batch size is 96 to simulate medium-throughput systems and K-means clustering is used. Bold entries indicate the clustering architecture achieving the highest expected max fitness in medium-throughput systems. All statistics were obtained from 500 independent repeats. The expected mean fitness was evaluated on the top 96 variants from the supervised learning model. The expected max fitness and the global maximal fitness hit rate were evaluated on the union of the top 96 variants from the supervised learning model and the 384 variants in training data. AA encoding was used for the GB1 dataset and Georgiev encoding for the PhoQ dataset (Methods).

simulated low-throughput systems can lead to better performance for all metrics; the global maximal fitness rate, in particular, reaches a value of 55.6% (Table 1).

We also tested CLADE on the PhoQ dataset. Unlike the fitness of GB1, measuring a simple protein physical property, the fitness of PhoQ measures an outcome from a complicated signaling cascade (Datasets). A more comprehensive encoding method was used that integrates over 500 amino-acid indices in the AAindex database[43]—Georgiev encoding[44,45]. Deep CLADE again demonstrated

substantial improvement compared to the case using global random-sampled training data, showing a 36% improvement on expected max fitness and a 2.9-fold improvement (from 7.2% to 20.6%) on global maximal fitness hit rate (Table 1). Despite CLADE showing a lower global maximal fitness hit rate and expected max fitness for the PhoQ dataset than for the GB1 dataset, the relative fitness improvement over wild-type protein measured by expected max fitness is much higher for PhoQ (7.8- and 67-fold, respectively, for GB1 and PhoQ; Supplementary Fig. 3b).

In applications, the robustness of CLADE performance to hyperparameters is more desirable because only one set of hyperparameters can be picked and applied. Surprisingly, the robustness was enhanced as the maximum hierarchy increased (Supplementary Figs. 4–6 and Supplementary Data 1). With shallow hierarchy $N = 1$, the global maximal fitness hit rate is relatively low and varies in a relatively large range from 30.6% to 41.2% for GB1. For deep hierarchy $N = 3$, the global maximal fitness hit rate is relatively higher and varies in a relatively small range from 41.6% to 50.8%, where a 2.2-fold improvement over random-sampling-based MLDE is guaranteed. CLADE performance on PhoQ is also relatively robust for $N = 3$, where global maximal fitness ranges from 14.0% to 20.6%, at least 1.9-fold improvement over random-sampling-based MLDE (Supplementary Data 1). Overall, deep CLADE ensures robust and accurate performance in DE.

Assessing the performance of stage-wise predictions. The proposed CLADE is a two-stage procedure in which supervised learning comes after the training data selection from clustering sampling. The first-stage sampling mainly explores the sequence library to select a diverse and informative training set. The second-stage ML mainly exploits fitness through greedy search from its predictions. Here we further dissect the roles and advantages of each stage.

First, the second-stage ML is critical to the final performance, regardless of the first-stage sampling methods. In CLADE, despite the majority of sequences being selected in the first stage (for example, fourfold in this work), the second-stage ML has a greater contribution to the final optimal sequences than the first-stage selection, and 35% and 41% higher expected max fitness can be achieved for GB1 and PhoQ, respectively (Fig. 3a,c). Similarly, ML followed by arbitrary sampling methods can substantially improve the final outcome. Many popular single-stage MLDE approaches, such as GP, can automatically calibrate the balance between exploration and exploitation, and it usually tends to exploit fitness at the late stage. Here we extend GP-based models to the two-stage approach by combining them with ML (GP-ML), where GP is used for the first few batches and ML is only applied at the last batch in the simulated medium-throughput system. We note that the inclusion of ML in GP leads to substantial improvement in discovered fitness for all acquisition functions tested, including Thompson sampling, ϵ-greedy and UCB (Fig. 3b,d). For example, over 50% improvement on expected max fitness was observed for Thompson sampling for both the GB1 and PhoQ datasets. Although UCB sampling achieves the highest expected max fitness among other sampling methods, improvement can still be observed with the proposed two-stage approach. Such a striking improvement relies on the more accurate predictions from ML models than GP models (Supplementary Data 1 and 2).

Second, the first-stage clustering sampling selectively explores informative clusters and ensures robust and accurate CLADE outcomes. The clustering sampling selectively picks clusters and restricts sampling within these clusters, and it can simply pair with a GP for sampling in selected clusters. Alternately, the two-stage strategy using GP sampling selects sequences in a global manner. We compared the performance of our two-stage procedure using clustering sampling (CLADE) with that using global sampling (GP-ML). A clear improvement in max fitness can be observed by introducing the clustering sampling, regardless of the acquisition functions used in the comparison (Fig. 3b,d). In particular, substantial improvement is achieved for Thompson sampling and the exploration sampling in ϵ-greedy (Fig. 3b,d). Although our two-stage approach with global sampling has largely different performance with respect to the acquisition function, the performance of CLADE is relatively robust and consistent. CLADE with UCB acquisition leads to the best performance and its global maximal fitness hit rate can reach 76% and 23% for GB1 and PhoQ, respectively (Supplementary Data 1).
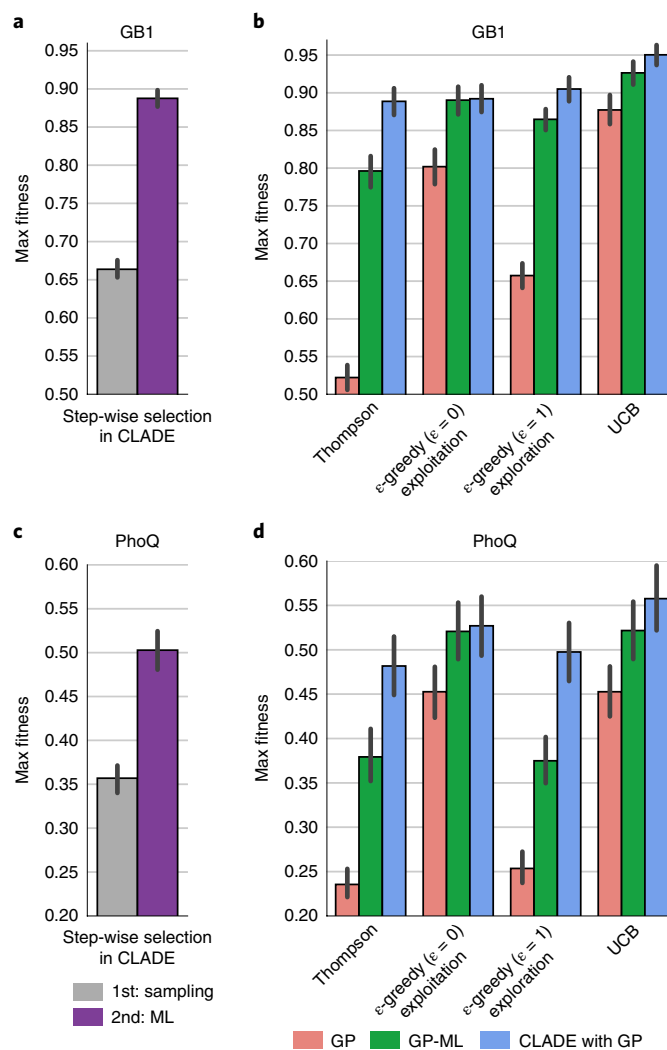

Fig. 3 | Two-stage procedure in CLADE. a,c, GB1 (**a**) and PhoQ (**c**) max fitness of CLADE selection in each stage, including the first-stage sampling (gray) and the second-stage supervised model (purple). **b,d**, GB1 (**b**) and PhoQ (**d**) max fitness from different MLDE methods: GP (pink), GP combining with ML (GP-ML; green) and CLADE integrating GP-type sampling (blue). Thompson, ϵ-greedy and UCB acquisition functions were used. Hyperparameters were extensively explored (Supplementary Data 1 and 2). Each bar plot represents the expected max fitness and the error bars show the 95% confidence interval. Simulations were run on medium-throughput systems in which the first-stage sampling selects 384 variants and the second-stage selection picks 96 top-predicted variants. In **a** and **c**, the sampling strategy for selected clusters was random sampling and 500 independent repeats were carried out. In **b** and **d** there were 200 independent repeats for each bar. AA and Georgiev encodings were used for GB1 and PhoQ, respectively.

Zero-shot predictor-based CLADE. Although clustering sampling can accurately select informative sequences (high-fitness) at a late stage, early-stage sampling cannot avoid exploring regions enriched with low-informative (zero- or low-fitness) regions to accumulate knowledge for the fitness landscape (Supplementary Figs. 7 and 8). Focused-training MLDE (ftMLDE) provides an approach to target informative sequences without the initial global search[2,46]. The zero-shot predictors employed by ftMLDE are capable of predicting protein fitness without the need for experimental screening. Predictions from two sequence-based zero-shot predictors—EvMutation[40] and multiple sequence alignment (MSA) transformer using a mask-filling
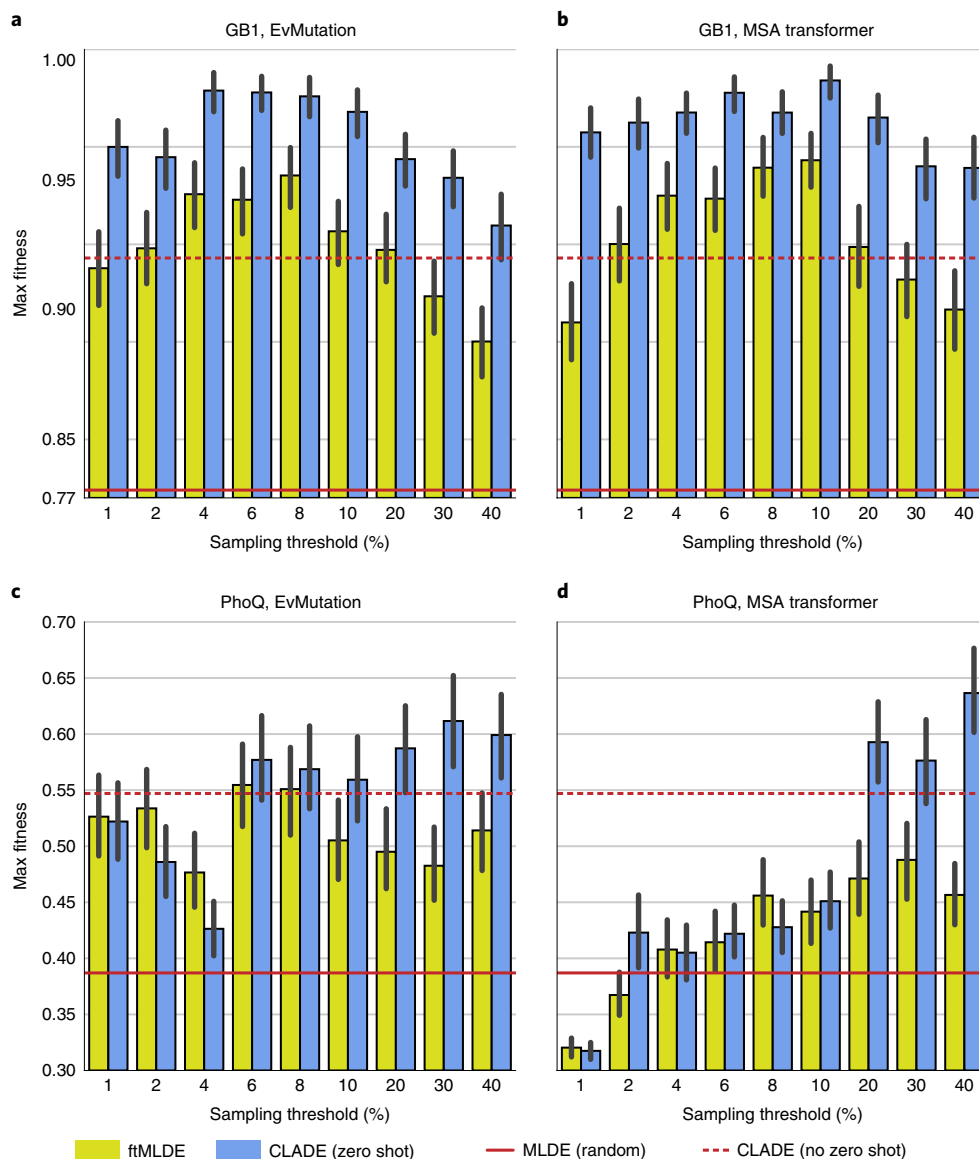
Fig. 4 | Improved CLADE performance using zero-shot predictions. a–d, The max fitness achieved by ftMLDE (yellow) and CLADE using zero-shot predictions (blue) for the GB1 dataset and EvMutation (**a**) and MSA transformation (**b**) and the PhoQ dataset and EvMutation (**c**) and MSA transformation (**d**). Each bar corresponds to a different sampling threshold (in %), with the percentile relating to the size of the sequence library (that is, 160,000 here). The sampling threshold determines the number of top sequences from zero-shot predictions used for training set sampling. Each bar plot represents the expected max fitness from 200 independent repeats, and the error bars show the 95% confidence interval. The solid line shows the expected max fitness from the random-sampling-based MLDE. The dashed red line shows the expected max fitness from the best-performing CLADE without zero-shot predictions (Table 1). Data for two datasets (GB1 and PhoQ) and two zero-shot predictions (EvMutation and MSA transformation) are presented. Simulations were run on the medium-throughput systems. AA and Georgiev encodings were used for GB1 and PhoQ, respectively.

protocol[22]—showed high correlations with fitness in GB1, with Spearman rank correlation coefficients (ρ) of 0.21 and 0.24, respectively[2]. Further validations on the PhoQ dataset showed even higher correlations, with ρ of 0.35 and 0.41 for EvMutation and MSA transformer, respectively (zero-shot calculations are described in the Methods). The zero-shot predictions rank the sampling priority for all variants in the sequence library. By picking up a sampling threshold, ftMLDE randomly selects training data within the subset consisting of top-ranked variants given by the zero-shot predictor.

Instead of random sampling over the top-predicted variants in ftMLDE, CLADE can also integrate with zero-shot predictors by performing clustering sampling. We similarly employed random sampling as the in-cluster sampling method in CLADE to compare with ftMLDE using global random sampling. Two zero-shot

predictors (EvMutation and MSA transformer) and nine sampling thresholds ranging from 1% to 40% of the size of the sequence library (that is, 160,000) were explored on both the GB1 and PhoQ datasets. For CLADE, we picked maximal hierarchy $N = 3$ and an identical increment of clusters for all hierarchies (that is, $K_1 = K_2 = K_3$). For the lower sampling threshold, the lower value of K_i was picked. Two reasonable K_i values were picked for each sampling threshold, and the case with largest expected max fitness was picked to compare with ftMLDE (Fig. 4 and Supplementary Data 3). For the GB1 dataset, both ftMLDE and CLADE show an improvement in max fitness over the random-sampling-based MLDE under all sampling thresholds (Fig. 4a,b). The best-performing ftMLDE achieves 0.943 expected max fitness and 74.5% global maximal fitness hit rate at 10% sampling threshold by using MSA transformer zero-shot

predictions, showing further improvement over the best-performing CLADE without zero-shot predictions. Furthermore, CLADE using zero-shot predictions achieves more accurate and robust performance, improving over ftMLDE for all sampling thresholds and outperforming CLADE without zero-shot predictions (Fig. 4a,b). With sampling thresholds of 4% and 10%, the best-performing CLADE achieves 0.979 and 0.984 on expected max fitness and 91% and 90.5% on global maximal fitness hit rate for EvMutation and MSA transformer zero-shot predictors, respectively. For the PhoQ dataset, both ftMLDE and CLADE show improvement over the random-sampling-based MLDE, except for a few cases with low sampling threshold using MSA transformer zero-shot predictions (Fig. 4c,d). Interestingly, CLADE can outperform ftMLDE even without using zero-shot predictions under most sampling thresholds, and the best-performing ftMLDE only shows negligible improvement with an expected max fitness of 0.555 and global maximal fitness hit rate of 22.5% at 6% sampling threshold using EvMutation zero-shot predictions. Although CLADE may have lower expected max fitness under low sampling thresholds for both zero-shot predictors, it has substantially improved max fitness using sufficiently large sampling thresholds. With sampling threshold of 30% and 40%, the best-performing CLADE achieves expected max fitnesses of 0.612 and 0.637 and global maximal fitness hit rates of 34% and 33.5% for EvMutation and MSA transformer zero-shot predictors, respectively.

Discussion

The clustering sampling in CLADE builds a hierarchical clustering with a tree structure. Similar searching approaches that use a hierarchical tree, such as hierarchical optimistic optimization (HOO)[47], deterministic optimistic optimization (DOO) and simultaneous optimistic optimization (SOO)[48], were previously proposed to optimize a smooth black-box function defined on continuum space. The partition with infinitely deep hierarchy ensures its fast convergence to the global maximum. However, the hierarchy of clustering cannot be too deep in CLADE because of the discrete sequence library and limited number of experimental batches. Indeed, downstream supervised learning is necessary to assist the clustering sampling to find optimal variants. Batched acquisitions can also be used to improve the sampling efficiency[37,42]. MLDE algorithms can be evaluated by using a (nearly) complete combinatorial library obtained from a screening of limited mutational sites. However, MLDE methods can also be applied to a library obtained from a large number of mutational sites (for example, a chimeras recombination library[34,35]). For the latter, insufficient data are typically available to define the complete landscape and the global maximal fitness hit rate cannot be evaluated.

CLADE can be implemented with any sequence-encoding method. Physicochemical descriptors have been widely applied in many ML tasks for predicting protein physical functions[12,16,49]. In this Article, two physicochemical sequence-encoding methods were tested. Interestingly, application of CLADE on GB1 using AA encoding achieves better performance than using Georgiev encoding, whereas PhoQ shows the opposite behavior. AA encoding represents a small subset of AAindex, whereas Georgiev gives a comprehensive low-dimensional representation of AAindex. For the GB1 dataset, the AA encoding may be sufficient to learn the relatively simple physical fitness for binding affinity, and Georgiev encoding may contain redundant information that leads to its underperformance. For the PhoQ dataset, the fitness is an outcome from a complicated signaling cascade. Four physicochemical descriptors from AA encoding may not be sufficient to learn the fitness, so Georgiev encoding outperforms AA encoding. Recently, the development of self-supervised pretraining methods has provided data-driven approaches for sequence-encoding methods[17,50]. However, the deep pretrained encodings usually perform worse than physicochemical

encoding[2] (Supplementary Section 4 and Supplementary Table 1). The consideration of homologs of the target protein in the pretrained model, for example, using MSA transformer[22], can capture the local mutational effects of variants and build up more informative encoding for MLDE[2]. Protein three-dimensional structural abstraction from topological and geometric tools would be another interesting featurization approach for CLADE[12,16].

Unlike active learning, the utilization of zero-shot predictors in the ftMLDE approach can largely reduce the experimental burden, requiring only two rounds of screening. The similar combination of CLADE and zero-shot predictors provides improvement over ftMLDE, but additional experimental iterations are required. With the rapid decrease in the cost of gene synthesis and the development of high-throughput site-directed mutagenesis[51], the increased cost in CLADE would be sufficiently compensated by the substantially improved performance in terms of increased expected max fitness and global maximal fitness hit rate. CLADE can also give instant feedback to experiments because of its computational efficiency, with the first-stage sampling taking just a few minutes and the second-stage supervised learning a few hours to run. In practice, the top predicted variants can be screened sequentially until the optimal variants are found. Although the larger number may lead to continually improved max fitness, the improvement is not substantial when this number is too large (Supplementary Fig. 9). The sequence-based zero-shot predictors have shown great generalization to various fitness landscapes[19,40,41], as has also been shown in this work. On the other hand, the structure-based zero-shot predictor applied on ftMLDE achieved a state-of-the-art 99.7% global maximal fitness hit rate on the GB1 dataset[2]. However, this powerful zero-shot predictor may be limited to well-defined fitness associated with a predictable protein function, which is not the case for the PhoQ dataset.

Methods

Datasets. In this work, a popular benchmark GB1 library was used to test CLADE. A PhoQ library that was used in an early MLDE study[42] was also considered. For both datasets, their fitness values were normalized into the range [0, 1] when applied to CLADE.

The GB1 dataset[4] is an empirical fitness landscape for protein G domain B1 (GB1; PDB 2GI9) binding to an antibody. Fitness was defined as the enrichment of folded protein bound to the antibody IgG-Fc. This dataset contains 149,361 experimentally labeled variants out of $20^4 = 160,000$ at four amino-acid sites (V39, D40, G41 and V54). The fitness of the remaining 10,639 unlabeled variants is imputed, but their values are not considered in this study. By normalizing the fitness to its global maximum, 92% of variants have fitness lower than 0.01 and 99.3% variants have fitness lower than 0.3 (Supplementary Fig. 3a).

For the PhoQ dataset[52], a high-throughput assay for the signaling of a two-component regulatory system—PhoQ–PhoP sensor kinase and a response regulator—was developed with a yellow fluorescent protein (YFP) reporter expressed from a PhoP-dependent promoter. Extracellular magnesium concentration stimulates the phosphatase or kinase activity of PhoQ, which can be reported by YFP levels. The combinatorial library was constructed at four sites (A284, V285, S288 and T289) located at the protein–protein interface between the sensor domain and kinase domain of PhoQ. Two libraries were constructed by using different extracellular magnesium treatments. In each library, the variants with comparable YFP levels to wild type were selected by fluorescence-activated cell sorting (FACS) and used for enrichment ratio calculations. Comparable YFP levels were strictly defined by two thresholds. The PhoQ dataset was previously studied using an MLDE model[42]. In this work, we took the enrichment ratios from the library with high extracellular magnesium treatment as fitness. The fitness value correlates to the probability that a variant has fluorescence in the given range, with this range defined as the wild-type-like activity in the original PhoQ work[52] (Supplementary Fig. 10). The fitness landscape has nearly complete coverage, with 140,517 quality read variants out of $20^4 = 160,000$. Like GB1, the PhoQ dataset was found to be overwhelmed with low- or zero-fitness variants, with 92% of variants having fitness lower than 0.01 and 99.96% of variants having fitness lower than 0.3, and the high-fitness variants are rarer than in the GB1 dataset (Supplementary Fig. 3a).

For the MLDE algorithms alone, both GB1 and PhoQ datasets, using enrichment ratios as fitness, provide suitable labels to learn and optimize. In applications, the optimization of fitness usually intends to improve a meaningful protein property. The fitness in GB1 directly correlates to a specific protein activity, that is, the binding affinity between GB1 and its antibody IgG-Fc, serving as an

excellent benchmark. However, fitness in the PhoQ dataset may only weakly correlate to protein activities, such as PhoQ–PhoP interaction strength and YFP fluorescence level. As such, the results from MLDE for the PhoQ dataset cannot be directly interpreted as a meaningful protein property.

Sequence encoding. In this work, two types of physicochemical sequence encoding method—AA and Georgiev—were used to test CLADE. The encoding matrix of the combinatorial library was standardized via StandardScalar() in scikit-learn[53] before further usage. The same encoding matrix was used for both unsupervised clustering and supervised learning models (Supplementary Section 1). First, the AA encoding consists of four physicochemical descriptors: molecular mass, hydropathy, surface area and volume (Supplementary Table 2). Molecular mass, hydropathy and surface area were obtained from the AAindex database[43] and volume from experimental work[54]. This encoding was previously used in protein stability change predictions[12]. Instead of picking a subset of the AAindex database, the Georgiev encoding[44,45] comprehensively integrated over 500 amino-acid indices in the AAindex database and gave a low-dimensional representation of these indices with 19 dimensions.

Gaussian process. The GP regression model[55] was used to infer the value of an unknown function $f(x)$ at a novel point x, given a set of observations X with labels Y. The posterior distribution of $f(x)$ given by GP can be predicted with mean $\mu(x)$ and standard deviation $\sigma(x)$. The GP regression was implemented by scikit-learn package[53]. The default radial basis function (RBF) kernel and other default parameters were used.

The next round of sequence selection was prioritized by the values of acquisition functions $\alpha(x)$, where the sequence with the largest acquisition in the unlabeled set X_0 will be screened first:

$$x_* = \arg\max_{x \in X_0} \alpha(x). \qquad (2)$$

Specifically, in this work, we selected a batch of unlabeled sequences with top values in acquisition functions for the next batch of screening.

The designs of the acquisition function depend on the posterior mean and variance. The simple greedy acquisition is defined by the posterior mean, which can maximize and exploit the expected fitness at each round:

$$\alpha_g(x) = \mu(x). \qquad (3)$$

On the other hand, with the acquisition identical to the posterior variance we can explore the uncertain regions to increase the knowledge and accuracy of the regression model. To balance the exploration–exploitation dilemma for these two extreme cases, ϵ-greedy acquisition takes the combination of them[38]:

$$\alpha_\epsilon(x) = \begin{cases} \mu(x), & \text{with probability } 1 - \epsilon, \\ \sigma(x), & \text{with probability } \epsilon. \end{cases} \qquad (4)$$

where ϵ is a hyperparameter to mediate this trade-off. In this work, we took ϵ as a constant and explored its values, while an alternate design would let ϵ decreases sequentially to enhance exploitation.

Another popular UCB acquisition can both exploit samples with large mean and explore samples with large variance, which has substantial theoretical background[39]. This takes the form

$$\alpha_{UCB}\mu(x) + \sqrt{\beta}\sigma(x). \qquad (5)$$

The trade-off parameter β decides the size of the confidence interval to be considered. For example, the acquisition function considers a 95% confidence interval when $\beta = 4$.

Thompson sampling exploits the label through random sampling according to the posterior mean and variance. The acquisition function is sampled from a normal distribution:

$$\alpha_T(x) \sim \mathcal{N}\left(\mu(x), \sigma(x)^2\right). \qquad (6)$$

Zero-shot predictions. The calculations of zero-shot predictions were followed by the ftMLDE package[2]. In this work, we tested two zero-shot predictors using EVmutation[40] and MSA transformer using a mask-filling protocol[22].

Before calculations of these zero-shot predictors, the EVcouplings webapp[56] generates MSAs and trains an EVmutation model for the target protein. The sequence of the target protein is the only input required. The alignments were searched against the UniRef100 dataset. Except bitscore, all other parameters were used as their default values (search iterations = 5, position filter = 70%, sequence fragment filter = 50%, removing similar sequences = 90%, downweighting similar sequences = 80%). The entire 56-residue sequence of GB1 (PDB 2GI9) was used for alignments:

MQYKLILNGKTLKGETTTEAVDAATAEKVFKQYANDNGVDGEWTYDD ATKTFTVTE

Bitscore was taken as 0.4 according to ref. [2], resulting in 56 redundancy-reduced sequences. The sequence of PhoQ (UniProtKB P23837) has 486 residues:

MKKLLRLFFPLSLRVRFLLATAAVVLVLSLAYGMVALIGYSVSFDKTTFRLL RGESNLFY TLAKWENNKLHVELPENIDKQSPTMTLIYDENGQLLWAQRDVP WLMKMIQPDWLKSNGFH EIEADVNDTSLLLSGDHSIQQQLQEVREDDDDA EMTHSVAVNVYPATSRMPKLTIVVVDT IPVELKKS**SYMVWSWFIYVLSANLLL VIPLLWVAAWWSLRPIEALAKEVRELEEHNRELLN PATTRELTSLVRNLNR LLKSERERYDKYRTTLTDLTHSLKTPLAVLQSTLRSLRSEKMSV SDAEPV MLEQISRISQQIGYYLHRASMRGGTLLSRELHPVAPLLDNLTSALNKVYQR KGV NISLDISPEISFVGEQ**NDFVEVMGNVLDNACKYCLEFVEISARQTDEH LYIVVEDDGPGI PLSKREVIFDRGQRVDTLRPGQGVGLAVAREITEQYEGKI VAGESMLGGARMEVIFGRQH SAPKDE

The four mutational sites (A284, V285, S288 and T289) are located at the interface between the sensor domain and kinase domain. In EVcouplings, we took 189 residues in the protein–protein interface (positions 188~376; bold fragment) to search for a more relevant homolog that covers the mutational sites. The authors of EVcouplings suggest generating ≥10L redundancy-reduced sequences[40,56]. By tuning bitscore, we took it to be 0.5, resulting in 2,998 redundancy-reduced sequences.

The zero-shot predictions from EVmutation were calculated for the combinatorial libraries using the model downloaded from the EVcouplings webapp. When applying MSA transformer, MSAs may need to be subsampled to make the model memory efficient. We used the hhfilter function in the HHsuite package[57] to subsample the alignments by maximizing the diversity, as suggested by the original MSA transformer publication[22]. For the MSAs of GB1 there were only 56 sequences, and subsampling was omitted. For the MSAs of PhoQ, the --diff parameter in hhfilter was taken as 100, which generates 128 sequences. The zero-shot predictions using MSA transformer were calculated by the mask-filling protocols using naive probability[2].

Unsupervised clustering and clustering sampling. In this work, two unsupervised clustering algorithms, K-means[23] and Louvain[25], were tested on CLADE. K-means clustering was computed using the scikit-learn package with default kmeans++ initialization[53]. Louvain clustering was computed on a shared nearest-neighbor graph implemented by the Seurat package[58] (Supplementary Section 6).

In clustering sampling, a cluster is selected according to the cluster-wise sampling probabilities. The cluster-wise sampling probabilities depend on the average fitness of selected variants in each cluster. The cluster with higher average fitness has a higher probability to be selected. In the kth cluster at the ith hierarchy, the sampling probability is given by

$$P_k^{(i)} = \frac{\frac{1}{\#C_k^{(i)}} \sum_{j \text{ in } C_k^{(i)}} y_j}{\sum_l \frac{1}{\#C_l^{(i)}} \sum_{j \text{ in } C_l^{(i)}} y_j}, \qquad (7)$$

where $C_l^{(i)} \subset I$ is the index set of the lth cluster at the ith hierarchy and I is the index set of the combinatorial library that gives each variant a unique index. Here, y_j is the fitness of the jth variant. Once the cluster is selected, in-cluster sampling is used to select a variant within this cluster. In one approach, the random sampling uniformly picks a variant. Another approach is GP-based model sampling. The GP model is trained on all labeled sequences. The difference for the in-cluster sampling with conventional GP is that we only pick variants within the selected cluster to maximize the acquisition function instead of searching globally.

In deep hierarchical clustering, only K-means is applied because it is easy to control the number of clusters with a single hyperparameter K. For maximum hierarchy N, the increment of clusters at the ith ($i \leq N$) hierarchy is given by K_i. The total number of clusters at the maximum hierarchy is the sum of these numbers $\sum_{i=1}^{N} K_i$. At a new hierarchy, clusters with higher average fitness are divided into more subclusters, and clusters with low average fitness are divided into fewer subclusters or not divided. The kth parent cluster at the $(i-1)$th hierarchy will be divided into $L_k^{(i)}$ subclusters at the ith hierarchy, and $L_k^{(i)}$ is given by

$$L_k^{(i)} = \begin{cases} [P_k^{(i)} K_i] + 1, & \text{if } k \neq k_0 \\ K_i - \sum_{j \neq k_0} [P_j^{(i)} K_i] + 1, & \text{if } k = k_0 \end{cases} \qquad (8)$$

where $k_0 = \arg\max_k \frac{1}{\#C_k^{(i)}} \sum_{j \text{ in } C_k^{(i)}} y_j$ is the index of the cluster having the largest average fitness from selected variants over all clusters. $[x]$ represents the largest integer not greater than x.

Here we summarize the workflow of clustering sampling together with the required hyperparameters. The structure of clusters needs to be determined before the sampling process, with $N+1$ hyperparameters, including maximum hierarchy N and the increment of clusters at each hierarchy K_i. The batch size, NUM$_\text{batch}$, is taken to be the number of variants being screened in parallel in the experiment. The batch size decides the frequency for updating the sampling probability

and clusters at the new hierarchy, and a lower batch size usually leads to more accurate CLADE prediction but higher cost in experiments. During sampling, the first-round selection chooses NUM_{1st} variants, which are equally picked over clusters to have a rough coverage of all clusters. After the first-round selection, the cluster-wise sampling probability is updated for every batch according to equation (7), and a new hierarchy is generated after every set of $NUM_{hierarchy}$ variants is screened until reaching the maximum hierarchy N. The sampling method to pick variants from the selected clusters can be either random sampling or GP-based sampling. The sampling process generates NUM_{train} labeled variants to train the downstream supervised learning model. The top M variants predicted by CLADE are experimentally screened. These numbers—NUM_{1st}, $NUM_{hierarchy}$, NUM_{train} and M—are all required to be multiples of batch size NUM_{batch}. Two batch sizes, 96 and 1, were used in this work. Batch size 96 was followed according to the small 96-well plate commonly seen in many experimental systems[3,33] and is referred to as a medium-throughput system in this work. Batch size 1 was used to simulate systems with extremely low throughput in which variants need to be screened one by one. The hyperparameters for medium- and low-throughput systems are provided in Supplementary Table 3. In application, NUM_{batch} can be picked according to the experimental protocol and T can be picked according to the screening capacity. The other three numbers can be selected according to our experiment and scaled to suitable values.

For clustering sampling using zero-shot predictions, we only sample within a subspace of the combinatorial library given by the top-ranking variants from the zero-shot predictions. The other steps are identical to the case without using zero-shot predictions.

Ensemble supervised learning. The MLDE package[2] was used for the supervised learning model in this work. An ensemble of 17 regression models optimized by Bayesian hyperparameter optimizations was used. Fivefold cross-validation was performed on training data and used to evaluate the performance of each model measured by mean square errors. Bayesian hyperparameter optimizations were performed to find the best-performing hyperparameters for each model. After hyperparameter optimizations, the top three models were picked and averaged to predict the fitness of unlabeled variants. Details are provided in Supplementary Section 2 and Supplementary Tables 4 and 5.

Evaluating metrics. Various metrics were used to evaluate the training data diversity and CLADE outcome. Mean fitness and max fitness were calculated in three sets: training data, the top M predicted variants and their union. In selecting the top M predicted variants, only variants that could be constructed by the recombination of variants in the training data were considered. This enhances the confidence of predictions by reducing extrapolations, especially when a less diverse training set is used. 'Global maximal fitness hit rate' calculates the frequency with which the global maximal variant is successfully picked in multiple independent repeats. 'Normalized discounted cumulative gain (NDCG)' is a measure of ranking quality to evaluate the predictive performance of CLADE on all unlabeled data. Its value is between 0 and 1. When this is close to 1, it indicates that variants ranked by the predicted fitness are similar to that ranked by the ground-truth fitness. Mean square error and Pearson correlation are used to evaluate the performance of the supervised learning for both cross-validation and testing. 'Modified functional attribute diversity' (MFAD) is a quantity used to measure data diversity[59]. In this Article we use it to measure the fitness and sequence diversity for training data. If T is the training data size, MFAD is given by

$$\text{MFAD} = \frac{\sum_{i=1}^{T}\sum_{j=1}^{T} d_{ij}}{T},$$ (9)

where d_{ij} represents the dissimilarity between the ith and jth samples. For fitness diversity, the dissimilarity is calculated by the difference of fitness between two samples:

$$d_{ij}^{\text{fitness}} = |y_i - y_j|.$$ (10)

For sequence diversity, the dissimilarity is calculated by the Euclidean distance between two samples of the physicochemical encoding:

$$d_{ij}^{\text{sequence}} = ||x_i - x_j||_2$$ (11)

where x_i is the physicochemical encoding feature vector of the ith variant and $||\cdot||$ is the Euclidean distance.

Data availability
The GB1 dataset[4] is available at https://www.ncbi.nlm.nih.gov/bioproject/PRJNA278685/ with accession code PRJNA278685. The PhoQ dataset has been reported in the literature[52]. The processed version of it used in this work is owned by the Michael T. Laub laboratory and is available at https://github.com/WeilabMSU/CLADE. Source data are provided with this paper.

Code availability
All source codes and models are publicly available at https://github.com/WeilabMSU/CLADE[60].

References
1. Tawfik, O. K. & S, D. Enzyme promiscuity: a mechanistic and evolutionary perspective. *Annu. Rev. Biochem.* **79**, 471–505 (2010).
2. Wittmann, B. J., Yue, Y. & Arnold, F. H. Informed training set design enables efficient machine learning-assisted directed protein evolution. *Cell Syst.* **12**, 1026–1045.e7 (2021).
3. Wu, Z., Kan, S. J., Lewis, R. D., Wittmann, B. J. & Arnold, F. H. Machine learning-assisted directed protein evolution with combinatorial libraries. *Proc. Natl Acad. Sci. USA* **116**, 8852–8858 (2019).
4. 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).
5. Yang, K. K., Wu, Z. & Arnold, F. H. Machine-learning-guided directed evolution for protein engineering. *Nat. Methods* **16**, 687–694 (2019).
6. Siedhoff, N. E., Schwaneberg, U. & Davari, M. D. Machine learning-assisted enzyme engineering. *Methods Enzymol.* **643**, 281–315 (2020).
7. Narayanan, H. et al. Machine learning for biologics: opportunities for protein engineering, developability and formulation. *Trends Pharmacol. Sci.* **42**, 151–165 (2021).
8. Mazurenko, S., Prokop, Z. & Damborsky, J. Machine learning in enzyme engineering. *ACS Catal.* **10**, 1210–1223 (2019).
9. Bojar, D. & Fussenegger, M. The role of protein engineering in biomedical applications of mammalian synthetic biology. *Small* **16**, 1903093 (2020).
10. Kim, G. B., Kim, W. J., Kim, H. U. & Lee, S. Y. Machine learning applications in systems metabolic engineering. *Curr. Opin. Biotechnol.* **64**, 1–9 (2020).
11. Tian, J., Wu, N., Chu, X. & Fan, Y. Predicting changes in protein thermostability brought about by single- or multi-site mutations. *BMC Bioinformatics* **11**, 370 (2010).
12. Cang, Z. & Wei, G.-W. Analysis and prediction of protein folding energy changes upon mutation by element specific persistent homology. *Bioinformatics* **33**, 3549–3557 (2017).
13. Quan, L., Lv, Q. & Zhang, Y. STRUM: structure-based prediction of protein stability changes upon single-point mutation. *Bioinformatics* **32**, 2936–2946 (2016).
14. Khurana, S. et al. DeepSol: a deep learning framework for sequence-based protein solubility prediction. *Bioinformatics* **34**, 2605–2613 (2018).
15. Hie, B., Bryson, B. D. & Berger, B. Leveraging uncertainty in machine learning accelerates biological discovery and design. *Cell Syst.* **11**, 461–477 (2020).
16. Wang, M., Cang, Z. & Wei, G.-W. A topology-based network tree for the prediction of protein–protein binding affinity changes following mutation. *Nat. Mach. Intell.* **2**, 116–123 (2020).
17. Rao, R. et al. Evaluating protein transfer learning with tape. *Adv. Neural Inf. Process. Syst.* **32**, 9689–9701 (2019).
18. Yang, K. K., Wu, Z., Bedbrook, C. N. & Arnold, F. H. Learned protein embeddings for machine learning. *Bioinformatics* **34**, 2642–2648 (2018).
19. Riesselman, A. J., Ingraham, J. B. & Marks, D. S. Deep generative models of genetic variation capture the effects of mutations. *Nat. Methods* **15**, 816–822 (2018).
20. 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).
21. Bepler, T. & Berger, B. Learning protein sequence embeddings using information from structure. In *Proc. International Conference on Learning Representations* (2018).
22. Rao, R. et al. MSA transformer. In *Proc. 38th International Conference on Machine Learning* Vol. 139, 8844–8856 (PMLR, 2021).
23. Hamerly, G. & Elkan, C. Learning the k in k-means. *Adv. Neural Inf. Process. Syst.* **16**, 281–288 (2004).
24. Frey, B. J. & Dueck, D. Clustering by passing messages between data points. *Science* **315**, 972–976 (2007).
25. Blondel, V. D., Guillaume, J.-L., Lambiotte, R. & Lefebvre, E. Fast unfolding of communities in large networks. *J. Stat. Mech.* **2008**, P10008 (2008).
26. Schubert, E., Sander, J., Ester, M., Kriegel, H. P. & Xu, X. DBSCAN revisited, revisited: why and how you should (still) use DBSCAN. *ACM Trans. Database Syst.* **42**, 1–21 (2017).
27. Sha, Y., Wang, S., Zhou, P. & Nie, Q. Inference and multiscale model of epithelial-to-mesenchymal transition via single-cell transcriptomic data. *Nucleic Acids Res.* **48**, 9505–9520 (2020).
28. Kuang, D., Ding, C. & Park, H. Symmetric nonnegative matrix factorization for graph clustering. In *Proc. 2012 SIAM International Conference on Data Mining* 106–117 (SIAM, 2012).

29. Oller-Moreno, S., Kloiber, K., Machart, P. & Bonn, S. Algorithmic advances in machine learning for single cell expression analysis. *Curr. Opin. Syst. Biol.* **25**, 27–33 (2021).

30. Saxena, A. et al. A review of clustering techniques and developments. *Neurocomputing* **267**, 664–681 (2017).

31. Zhong, Y., Ma, A., Soon Ong, Y., Zhu, Z. & Zhang, L. Computational intelligence in optical remote sensing image processing. *Appl. Soft Comput.* **64**, 75–93 (2018).

32. Li, G., Dong, Y. & Reetz, M. T. Can machine learning revolutionize directed evolution of selective enzymes? *Adv. Synth. Catal.* **361**, 2377–2386 (2019).

33. Saito, Y. et al. Machine-learning-guided mutagenesis for directed evolution of fluorescent proteins. *ACS Synth. Biol.* **7**, 2014–2022 (2018).

34. Bedbrook, C. N., Yang, K. K., Rice, A. J., Gradinaru, V. & Arnold, F. H. Machine learning to design integral membrane channelrhodopsins for efficient eukaryotic expression and plasma membrane localization. *PLoS Comput. Biol.* **13**, e1005786 (2017).

35. Romero, P. A., Krause, A. & Arnold, F. H. Navigating the protein fitness landscape with Gaussian processes. *Proc. Natl Acad. Sci. USA* **110**, E193–E201 (2013).

36. Mason, D. M. et al. Deep learning enables therapeutic antibody optimization in mammalian cells by deciphering high-dimensional protein sequence space. *bioRxiv* https://doi.org/10.1101/617860 (2019).

37. Hie, B. L. & Yang, K. K. Adaptive machine learning for protein engineering. Preprint at https://arxiv.org/abs/2106.05466 (2021).

38. Schulz, E., Speekenbrink, M. & Krause, A. A tutorial on Gaussian process regression: modelling, exploring and exploiting functions. *J. Math. Psychol.* **85**, 1–16 (2018).

39. Srinivas, N., Krause, A., Kakade, S. & Seeger, M. Gaussian process optimization in the bandit setting: no regret and experimental design. In *Proc. 27th International Conference on Machine Learning* 1015–1022 (ACM, 2010).

40. Hopf, T. A. et al. Mutation effects predicted from sequence co-variation. *Nat. Biotechnol.* **35**, 128–135 (2017).

41. Meier, J. et al. Language models enable zero-shot prediction of the effects of mutations on protein function. Preprint at *bioRxiv* https://doi.org/10.1101/2021.07.09.450648 (2021).

42. Yang, K. K., Chen, Y., Lee, A. & Yue, Y. Batched stochastic Bayesian optimization via combinatorial constraints design. In *Proc. 22nd International Conference on Artificial Intelligence and Statistics* 3410–3419 (PMLR, 2019).

43. Kawashima, S., Ogata, H. & Kanehisa, M. AAindex: amino acid index database. *Nucleic Acids Res.* **27**, 368–369 (1999).

44. Ofer, D. & Linial, M. ProFET: feature engineering captures high-level protein functions. *Bioinformatics* **31**, 3429–3436 (2015).

45. Georgiev, A. G. Interpretable numerical descriptors of amino acid space. *J. Comput. Biol.* **16**, 703–723 (2009).

46. Wittmann, B. J., Johnston, K. E., Wu, Z. & Arnold, F. H. Advances in machine learning for directed evolution. *Curr. Opin. Struct. Biol.* **69**, 11–18 (2021).

47. Bubeck, S., Munos, R., Stoltz, G. & Szepesvári, C. X-armed bandits. *J. Mach. Learn. Res.* **12**, 1655–1695 (2011).

48. Munos, R. Optimistic optimization of a deterministic function without the knowledge of its smoothness. *Adv. Neural Inf. Process. Syst.* **24**, 783–791 (2011).

49. Pahari, S. et al. SAAMBE-3D: predicting effect of mutations on protein-protein interactions. *Int. J. Mol. Sci.* **21**, 2563 (2020).

50. 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).

51. Strain-Damerell, C. & Burgess-Brown, N. A. in *High-Throughput Protein Production and Purification* 281–296 (Springer, 2019).

52. Podgornaia, A. I. & Laub, M. T. Pervasive degeneracy and epistasis in a protein-protein interface. *Science* **347**, 673–677 (2015).

53. Pedregosa, F. et al. Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011).

54. Zamyatnin, A. Protein volume in solution. *Progr. Biophys. Mol. Biol.* **24**, 107–123 (1972).

55. Rasmussen, C. E. in *Summer School on Machine Learning* 63–71 (Springer, 2003).

56. Hopf, T. A. et al. The EVcouplings Python framework for coevolutionary sequence analysis. *Bioinformatics* **35**, 1582–1584 (2019).

57. Steinegger, M. et al. HH-suite3 for fast remote homology detection and deep protein annotation. *BMC Bioinformatics* **20**, 473 (2019).

58. Butler, A., Hoffman, P., Smibert, P., Papalexi, E. & Satija, R. Integrating single-cell transcriptomic data across different conditions, technologies and species. *Nat. Biotechnol.* **36**, 411–420 (2018).

59. Schmera, D., Erős, T. & Podani, J. A measure for assessing functional diversity in ecological communities. *Aquatic Ecol.* **43**, 157–167 (2009).

60. YuchiQiu/CLADE: *Nature Computational Science* publication accompaniment (v1.0.0) (Zenodo, 2021); https://doi.org/10.5281/zenodo.5585394

Author contributions
All authors conceived this work and contributed to the original draft, review and editing. Y.Q. performed experiments and analyzed the data. G.-W.W. provided supervision and resources and acquired funding.

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/s43588-021-00168-y.

Correspondence and requests for materials should be addressed to Guo-Wei Wei.

Peer review information Reviewer recognition statement *Nature Computational Science* thanks the anonymous reviewers for their contribution to the peer review of this work.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.