DNA representations and generalization performance of sequence-to-expression models

Yuxin Shen[1], Grzegorz Kudla[2], Diego A. Oyarzún[1,3,†]

[1]School of Biological Sciences, University of Edinburgh, UK

[2]Institute for Genetics and Cancer, University of Edinburgh, UK

[3]School of Informatics, University of Edinburgh, UK

[†]Corresponding author: d.oyarzun@ed.ac.uk

Abstract

The increasing demand for biological products drives many efforts to engineer cells that produce heterologous proteins at maximal yield. Recent advances in massively parallel reporter assays can deliver data suitable for training machine learning models and support the design of microbial strains with optimized protein expression phenotypes. The best performing sequence-to-expression models have been trained on one-hot encodings, a mechanism-agnostic representation of nucleotide sequences. Despite their excellent local predictive power, however, such models suffer from a limited ability to generalize predictions far away from the training data. Here, we show that libraries of genetic constructs can have substantially different cluster structure depending on the chosen sequence representation, and demonstrate that such differences can be leveraged to improve generalization performance. Using a large sequence-to-expression dataset from *Escherichia coli*, we show that non-deep regressors and convolutional neural networks trained on one-hot encodings fail to generalize predictions, and that learned representations using state-of-the-art large language models also struggle with out-of-domain accuracy. In contrast, we show that despite their poorer local performance, mechanistic sequence features such as codon bias, nucleotide content or mRNA stability, provide promising gains on model generalization. We explore several strategies to integrate different feature sets into a single predictive model, including feature stacking, ensemble model stacking, and geometric stacking, a novel architecture based on graph convolutional neural networks. Our work suggests that integration of domain-agnostic and domain-aware sequence features offers an unexplored route for improving the quality of sequence-to-expression models and facilitate their adoption in the biotechnology and pharmaceutical sectors.

1 Introduction

Cell engineering has witnessed remarkable progress and is rapidly paving the way for new approaches to chemical manufacture. A central task in cell engineering is the production of proteins for the pharmaceutical, materials, food, and many other sectors [1, 2, 3]. Traditional techniques for optimising protein expression have often been laborious and costly, requiring continual iterations between DNA sequence design and strain phenotyping. This gap has created a need for computational models with predictive capabilities that can aid the various stages of the design cycle. Most recently, new techniques for massively parallel reporter assays have enabled the creation of datasets that are well suited for building sequence-to-expression models using methods from machine learning and artificial intelligence (AI). Such measurement technologies combine high-throughput DNA assembly, DNA sequencing and rapid phenotyping assays to produce large genotype-phenotype association datasets [4, 5, 6].

Sequence-to-expression models have been built for libraries of various genetic elements such as promoter sequences [7, 8, 9], ribosomal binding sites [10, 11] and other non-coding regions [12, 13]. These predictors are useful for sequence optimization and discovering variants with improved expression phenotypes. Several studies have embedded such models into algorithms for discovery of new variants using optimization methods [8] and techniques from generative models [14, 15, 16, 17]. Although the current literature has a strong focus on improvements to model architectures that can deliver greater predictive power [18], with the size of sequence-to-expression datasets growing into thousands up to millions of variants, it is becoming increasingly clear that off-the-shelf deep learning architectures such as convolutional neural networks, recurrent neural networks or transformers can readily provide high predictive accuracy [19].

A much less explored aspect is the strategy employed for encoding DNA sequences into a numerical format suitable for model training. To date, the best performing models have been trained on one-hot encoded sequences, whereby each nucleotide is represented as a 4-dimensional binary vector. Studies have consistently shown that one-hot encodings provide the best predictive accuracy and, crucially, they outperform a number of mechanistic sequence features known to correlate with protein expression. For example, despite their known impact on translational processes, mechanistic features such as RNA secondary structure [20, 21, 22], codon bias [23, 24, 25, 26, 27], nucleotide content [28, 29] or peptide hydrophobicity [30] are comparatively poorer predictors of expression than one-hot encodings [10, 13, 31, 32]. These results demonstrate the remarkable ability of mechanism-agnostic models to extract hidden correlations between sequence and expression phenotypes. However, recent studies have also raised awareness of the limited ability of such models to extrapolate their predictions beyond

the training data [33, 34]. Such poor generalization is a well-recognized caveat of sequence-to-expression models and limits their utility as tools for sequence discovery in regions distant from the training library.

Here, we perform a comparative study of sequence-to-expression models trained on different sequence features. Using a large dataset in *Escherichia coli* [22], we show that the shape and coverage of the library can change substantially across sequence representation methods, as demonstrated by qualitatively different cluster structure. We show that such alternative representations can be employed to improve generalization performance and thus deliver accurate predictions in regions that are distant from the training data. We compared various strategies that combine models and features into a single predictive model, so as to combine the high-quality local predictions of one-hot encodings with the improved generalization power afforded by mechanistic features. The results suggest that feature integration can produce stronger predictions and offer a new route for training sequence-to-expression models with improved generalization performance.

2 Results

2.1 Shape of the variant library in different DNA representation spaces

Our study focuses on the dataset reported by Cambray and colleagues [22], which contains a large library of 5' CDS variants (96nt) preceding an super folding green fluorescent protein (sfGFP) gene expressed in *Escherichia coli*. The full library contains over 200,000 variants distributed across 56 mutational series with approximately 4,000 variants each, accompanied by their fluorescence readouts averaged over four experimental repeats. This is a useful dataset for our study because variants were designed on the basis of mechanistic features that impact translational efficiency in different mutational series (Figure 1A). Previous analyses of these data [13] shows that models trained on one-hot encoded sequences can predict sfGFP expression with high accuracy, while models trained on mechanistic features have substantially poorer performance. The low predictive power of mechanistic features agrees with similar findings in other libraries designed to affect translational efficiency [10, 31].

We describe the variants in the Cambray library using two sets of features: one-hot encodings that lead to a 384-dimensional binary vector, and a 6-dimensional vector of mechanistic features that include the predicted minimum free energy of secondary structures computed for three overlapping windows, the nucleotide (AT) content, the codon adaptation index (CAI), and the mean hydropathy index (MHI) of the peptide chain (Figure 1A).

To gain a first understanding of the distribution of sequence variants across these two feature spaces, we employed the Uniform Manifold Approximation (UMAP [35]) algorithm to represent the variants in a two-dimensional space. The results in Figure 1B–C suggest substantial differences in the structure of the whole library as seen in the two feature spaces. We observe that individual mutational series appear to be clustered in specific regions of the one-hot encoded space, whereas in the mechanistic feature space they tend to be homogeneously spread across the full library; visualizations for other choices of UMAP hyperparameters can be found in Supplementary Figure S1–S2.

Since dimensionality reduction methods such as UMAP can produce visual artefacts [36], we employed k-means clustering to quantitatively examine the distribution of variants in both feature sets. The clustering results (Figure 1D, top) show that both feature spaces have substantially different clustering structure; the slope of the k-means loss (sum-of-square-errors, SSE) reflects the improvement in the quality of the k-means clustering for increasing number of clusters. The smooth decrease of the SSE for mechanistic features suggest a lack of a clear cluster structure in the data. But in the case of one-hot encodings we observe a quasi-linear decrease followed by an "elbow" between $k = 56$ and $k = 60$ clusters; this suggests variants are more clustered in the one-hot feature space than in the mechanistic feature space. We employed the Adjusted Rand Index (ARI) to score the similarity of the k-means clusters and the original labels of the 56 mutational series. In Figure 1D(middle) we observe that for $k = 56$ clusters, the one-hot encoding clusters have a high degree of similarity with the 56 series used to assemble the dataset. We did not observe such similarity with respect to the clusters obtained from the mechanistic features, for any number of clusters. To further quantify the quality of the clusters, we computed the average silhouette score (SC) across all variants for each of the clusterings (Figure 1D, bottom); the SC is a measure of how similar each variant is to its own cluster as compared to other clusters, and thus provides a way to assess the quality of clusters in both feature spaces. The results show that mechanistic features have a weak cluster structure, reflected by low SC values with decreasing trend, while the one-hot encodings appear to have well-separated clusters for $k > 50$, reflected by higher SC values starting around $k > 50$. This provides further evidence that the cluster-like structures suggested by the UMAP representations (Figure 1B) also appear in the full feature space and that one-hot encoded variants tend to cluster according to the mutational series they belong to.

We further sought to detect series-specific biases in both sequence representation strategies. To this end, we performed global and series-specific principal component (PC) analyses of all variants using both mechanistic and one-hot encoded features. The results in Figure 1E suggest

that global PCs computed on mechanistic features explain on average the same amount of variance as their series-specific counterpart, while the global PCs on one-hot encoding explain much less variance than the series-specific PCs. This is consistent with the different cluster structures found for both feature sets (Figure 1D). Moreover, it suggests that mechanistic features can represent the variant space consistently within and across mutational series, while one-hot encodings have an improved representation power within each mutational series.

2.2 Impact of sequence representations on model generalization

Given the observed differences in the shape and representation power of the two feature spaces, we reasoned that these would affect the ability of machine learning models to generalize predictions beyond the variants employed for training. The rationale is that strongly clustered features would lead to highly specialized models, whilst non-clustered input features may contain information that correlates with phenotypes far away from the training set. To test this hypothesis, we conducted a computational experiment designed to evaluate model performance when trained and tested in different mutational series.

We focused on three well-adopted machine learning models (linear ridge model; random forest, RF; multilayer perceptron, MLP) trained and tested on all pairwise combinations of mutational series. Models were trained on 90% of variants from each series (∼2800 sequences per series), and tested on 10% left-out variants of the same series and 100% of variants from other mutational series. For fair comparisons, we employed the model hyperparameters reported by Nikolados *et al* [13], which were determined via grid search on a large cross-validation dataset including variants from all mutational series. Model performance was scored with the coefficient of determination (R^2) between predictions and ground truth measurements:

$$R^2 = 1 - \frac{\sum_i (y_i - f_i)^2}{\sum_i (y_i - \overline{y})^2}, \tag{1}$$

where y_i and f_i are the ith measurement and prediction on the test set, respectively, and $\bar{y}$ is the average expression measured. The R^2 score is exactly zero for the naive regressor, i.e. a model that predicts average expression in the test data for all variants in the test set, and negative R^2 values are indicate of a wrong model structure that is worse than predicting the average observation.

The cross-testing results (Figure 2A) reveal a strong impact of the chosen feature set on the ability of models to generalize predictions to different mutational series. Mechanistic features have substantially higher generalization performance for the three considered models. We found several combinations of series for which generalization performance is near or even matches the quality of local predictions, represented by the diagonal of the heatmaps in Figure 2A. For

example, models trained and tested on the series triplets $\{1, 2, 3\}$ or $\{31, 32, 33\}$ have excellent R^2 scores throughout. Moreover, since the employed model hyperparameters were optimized for one-hot encodings only [13], further improvements to generalization performance can likely be achieved by re-optimizing hyperparameters for the mechanistic feature set.

In Figure 2B we show representative scatter plots of model predictions against the ground truth measurements, which illustrate cases in which local performance is either retained or partially lost when testing in a different mutational series (Figure 2B top and middle), and a case in which model structure is wholly unsuited for test set and leads to negative R^2 scores (Figure 2B, bottom). The cross-testing results in Figure 2A generally suggest that for some specific mutational series, all three model architectures consistently fail to generalize predictions even when trained on mechanistic features. For example, any model trained on series 34 or 38 have poor generalization performance across all other series. We confirmed this observation with hierarchical clustering of the R^2 scores using the ridge model as baseline, which revealed that the generalization patterns are largely preserved across models (Supplementary Figure S3). This suggests that in some cases poor generalization may depend on the genotype-phenotype data rather than the model architecture itself. We thus reasoned that poor performance may stem from difference in the variant space across series, but examination of the UMAP representations for those series revealed no qualitative difference in the spread of variants in the 6-dimensional feature space (Supplementary Figure S4). Instead, we found statistically significant differences in the fluorescence phenotypes across series (Figure 2C). Pairs of series with poor generalization performance, i.e. with a nil or negative R^2 score, are associated higher Jensen-Shannon distances between their expression phenotype distribution. This suggests that poor generalization performance can be ascribed to phenotypic differences between mutational series.

To assess the model performance in more detail, we quantified the generalization performance of each model by computing the average R^2 scores when testing on the other 55 mutational series. The local performance is the R^2 score computed on the same mutational series. A closer examination of the two encodings reveal that the increased generalization afforded by the mechanistic features comes at the expense of slightly weaker local predictions than models trained on one-hot encodings (Figure 2D–E). Furthermore, the comparisons among different models indicate that the ridge model, which is the weakest local model, provides the best generalization performance across different mutational series. This is in line with considerations on model complexity and the bias-variance trade-offs from statistical learning [37]. In the case of the linear ridge model, only seven parameters were learned, resulting in a model with low complexity and strong generalization ability.

Recently, various large language models (LLMs) have been introduced as novel paradigm for representation of nucleotide sequences, providing encouraging performance for gene identification [38] and RNA structure prediction [39]. These models produce unsupervised sequence representations that can be used for a variety of downstream supervised prediction tasks. To establish the generalization power of such representations, we embedded the sequences in our dataset using HyenaDNA [38], a DNA language model pretrained on a human reference genome, and performed downstream regression of protein expression levels using the ridge, MLP and RF models as in Figure 2. The results in Supplementary Figure S5 show a remarkable ability of HyenaDNA embeddings to deliver accurate local predictions, without the need for fine-tuning the LLM to our specific variant library in *E. coli*. A challenge here is the high dimensionality of the HyenaDNA embeddings ($p = 12,544$ dimensions for constructs of size 96nt), which makes training on small datasets prone to overfitting. In terms of generalization, however, we observed a similarly poor performance when testing the predictive power of LLM embeddings across different mutational series.

2.3 Integration of sequence representations into a single predictive model

Our results suggest that one-hot encodings can deliver excellent local predictions while mechanistic features help with generalization performance. We thus sought to explore the integration of both feature sets into a single predictive model. To this end, we explored three complementary strategies: *feature stacking*, where both feature sets are merged into a single feature vector, *ensemble stacking*, where models trained on different feature sets are stacked into a single predictor, and *geometric stacking*, a novel model architecture based on convolutional graph neural networks.

Feature stacking We first sought to increase model performance by simply merging the mechanistic and one-hot encoded feature vectors (Figure 3A). Each variant is thus represented by a 390-dimensional feature vector (6 mechanistic features plus 384 one-hot encoded features). We retrained the random forest model employed in Figure 2A on the stacked feature vectors, and tested the accuracy of local and out-of-domain predictions. We opted for the random forest model because it shows the best local performance when trained on one-hot encoded features (Figure 2D) and worst generalization performance when trained on mechanistic features (Figure 2E). The results in Figure 3A show that the model retains the best of both strategies: a slight improvement in R^2 scores in local predictions, and a generalization R^2 comparable to the model trained on mechanistic features.

The feature stacking strategy takes advantage of both sequence representations and capi-

talizes on local and generalization accuracy. We examined this observation in more detail by computing feature importance scores for all 56 models across the full 390-dimensional feature vector (Figure 3B). The importance scores suggest that the model effectively ascribes comparable importance to mechanistic and one-hot encoded features. We also note that, in agreement with the regression analysis of Cambray *et al* [22], the importance scores are higher for the first third of the construct and the MFE1 mechanistic feature, which corresponds to the predicted mRNA structure stability of the same region. This verifies that the model is detecting similar sequence-to-expression correlations that were reported in the original work [22].

Ensemble model stacking The results in Figure 2 suggest that different models can also exhibit a varying ability to generalize predictions. For example, from Figure 2D–E we observe that the random forest model is best at local predictions, while the ridge model is best at generalization. We thus explored the use of ensemble learning, a strategy commonly employed to combine weak learners into a meta-model with improved performance [40]. As illustrated in Figure 4A, the model ensemble aggregates predictions of many combinations of models and features sets; to direct our search towards models with improved generalization performance, we focused on combinations of mechanistic features and the stacked features from Figure 3A. The combined predictions are further fed into an output regressor that produces the final prediction.

We trained many combinations of intermediate and final output models, following the same training strategy from the previous analyses. Although we found that an output Gradient Boosting regressor consistently improved performance across many of the tested ensembles, specific combinations of intermediate models showed large beneficial and detrimental variations in predictive power. In Figure 4B we map the performance of four representative ensembles, alongside the reference models from Figure 2D–E with best local and generalization R^2 scores (shown in gray). Ensembles 1 and 4 achieve a balance between local predictions and generalization ability, and they both use mechanistic features only. The use of stacked feature vectors in ensembles 2 and 3 tends to bring local performance on par with the best performing model in local predictions, with similar generalization performance as ensembles 1 and 4. Altogether, this analysis suggests that ensemble learning can be an effectively tool to control the balance between local and generalization performance. As in the previous cases, further performance gains could be achieved with model and feature-specific hyperparameter optimization.

Geometric stacking with graph neural networks Deep learning architectures have demonstrated great predictive power across a range of sequence-to-expression tasks [19]. Convolutional neural networks (CNN), in particular, have been extensively employed to regress protein expres-

sion, often achieving high performance [8, 9, 12, 13]. We thus sought to determine if feature integration could also improve the generalization ability of CNN models. To this end, we first employed the CNN architecture proposed in [13], which is a variation of similar model structures employed in the literature [9, 12], and trained it on stacked features like those in Figure 3A. As an alternative approach, we developed a new strategy that integrates both feature sets into a graph neural network [41], a large class of deep learning models that has found applications in diverse domains with structured data, such as natural language processing [42], chemistry [43], drug discovery [44] and weather forecasting [45]. The core idea is to build a graph with nodes that represent sequence variants labelled with the sfGFP fluorescence measurements, and edges weighted by a pairwise similarity score between variants in the mechanistic feature space. This allows to embed the one-hot encoded sequences onto a graph that models the mechanistic similarity between variants, and use this information to improve generalization performance using semi-supervised learning on a node regression task (Figure 4C). As model architecture, we employed a graph convolutional neural network (GCN [46]) so as to retain the excellent local predictive power of one-hot encodings processed through convolutional architectures [8, 9, 12, 13].

We trained both CNN and GCN models on each mutational series and cross-tested their predictive power in the other 55 mutational series; details on model training can be found in the Methods, and model hyperparameters and learning curves in Tables S3–S4 and Supplementary Figure S7. The cross-testing results in Figure 4D show that the GCN provides substantial improvements in generalization over the convolutional model. Since both models have simultaneous access to one-hot encodings and mechanistic features, this result suggests that our graph-based strategy can effectively improve generalization performance. We also found that the GCN model suffers from poorer local performance (Figure 4E), and this is an area for further refinement of this architecture.

3 Discussion

Thanks to the availability of large screens of genotype-phenotype associations, recent years have witnessed an increased interest in machine learning models of protein expression. In synthetic biology, these models serve as valuable tools for identifying DNA sequences with improved expression levels. Leveraging such models, designers can optimize their designs by querying sequences that have not undergone experimental measurement, using the predictions to refine their constructs towards specific objectives. For example, Höllerer and colleagues trained deep learning models to design novel ribosomal binding sites with improved expression output [10], while Kotopka *et al* [8] used a similar approach to design novel yeast promoters with stronger

expression levels. Other more recent works include the use of generative models to design of various regulatory elements [14, 17].

A critical requirement for these models is the capacity to extrapolate or generalize their predictions to regions of the sequence space not covered during training. This becomes particularly pertinent in sequence optimization routines, where query sequences may deviate significantly from the training data, leading to low-confidence predictions that, in turn, impact the success rate of candidate sequences during downstream experimental testing. Several recent works have focused on the limited generalization performance of sequence-to-expression models [33, 34], and various recent studies have tackled this challenge either by training on larger data, or by improving the algorithms for sequence search. For example, Vaishnav *et al.* [9] utilized one of the largest promoter-expression datasets in *Saccharomyces cerevisiae* to construct a convolutional neural network with improved generalization performance. In scenarios with limited data, Nikolados *et al.* [13] highlighted that increasing the diversity of training sequences can enhance the generalization performance of deep learning models. Other approaches aim to address poor generalization by navigating the sequence space differently when querying models. Linder *et al.* [15] developed a sequence optimization algorithm that balances sequence diversity with model uncertainty, while recent advancements in generative modeling techniques have proven successful in generating robust sequences [14].

In this study, we propose a novel strategy to enhance model generalization by employing alternative sequence representations for model training. Our analysis shows that the shape of a variant library across different feature sets can be exploited to improve model generalization. Using a comprehensive dataset of the leading sequences in coding region in *Escherichia coli* [22], we demonstrate that while one-hot encodings exhibit a strongly clustered structure, an alternative representation incorporating mechanistic features related to nucleotide content, codon usage, and biophysical properties results in a more uniform distribution of variants across the feature space. Despite the inferior local performance of these mechanistic features, we show that their inclusion can enhance model generalization. We explore various strategies to integrate both feature sets into a predictive model, achieving improved performance in both local and out-of-domain testing regimes..

The adoption of sequence-to-expression models requires a careful balance between predictive power and the size/cost of the training data. Our results offer insights that can guide the design of variant libraries for model training, for example through the use of Design-of-Experiments approaches that cover the sequence space in ways that trade-off local and out-of-domain predictive performance. We have shown that the incorporation of prior biological knowledge into

sequence-to-expression models can enhance their utility as tools for sequence design and discovery. The integration of domain-aware and domain-agnostic features holds promise for advancing the development of data-driven models of protein synthesis, and thus expedite the development of microbial strains tailored for industrial biotechnology applications.

4 Methods

4.1 Data preprocessing and visualization

We employed the sequence-to-expression dataset of 5' CDS variants and protein expression levels from Cambray *et al* [22]. We first cleaned the data, deleting all the sequences with missing GFP fluorescence information, and took the average GFP fluorescence of four replicates. Expression levels were normalized to the $[0, 1]$ range. The encoding methods of DNA are either by one-hot or by mechanistic features. One-hot encodings were produced with custom Python code to turn each nucleotide into binary 4-tuples, which after concatenation form the 384-dimensional one-hot feature vectors. The mechanistic features were obtained from the original source [22]; we selected 6 mechanistic features from the sequence properties data: the predicted minimum free energy of mRNA secondary structures computed for three overlapping windows (MFE1 - MFE3), the nucleotide (AT) content, the codon adaptation index (CAI), and the mean hydropathy index (MHI) of the peptide chain. We excluded additional features codon ramp bottleneck position and strength because of their low correlation with the expression levels reported in the original work [22]. All six features were MinMax scaled to the $[0, 1]$ range.

To visualize the distribution of different mutational series in the library, we employed the UMAP algorithm [35] on the feature space of both the one-hot encoding and mechanistic features of DNA sequences. We conducted the parameter grid of two parameters in UMAP: the number of the local neighbors, and the minimum distance apart that points are allowed to be in the 2D space; results for nine different hyperparameter combinations can be found in Supplementary Figure S1–S2. For the UMAP plots in Figure 1B, the number of neighbours is set to 15, 30 and 45 respectively, with the minimal distance equal to 0.1. In Figure 1C, the number of neighbours is set to 30, with the minimal distance equal to 0.1. In both figures, mutational series 1 and 30 are labelled in red and yellow, respectively.

4.2 Data clustering

To examine the cluster structure of the library, we clustered variants using the k-means algorithm on both feature sets. We evaluated the sum loss of k-means clustering, the adjusted rand index

(ARI) and the silhouette scores for Figure 1D. The loss of k-means clustering is the sum of squared distances of each sample point to its closest cluster center. The Adjusted Rand Index (ARI) is a similarity measure between two clustering. It considers all pairs of samples and counting pairs that are assigned in the same or different clusters in the predicted and true clusterings. Here, we set the labels of the original mutational sequences as ground truth, and compared them to the label of k-means clusterings. The silhouette score is calculated based on the mean intra-cluster distance and the mean near-cluster distance to quantify the clustering behaviour. In Figure 1D we calculated the k-means loss and ARI across 25 repeats with different random seeds, while the silhouette scores were computed across 10 repeats with different random seeds.

4.3 Data split and training of non-deep models

The dataset consists of 56 different mutational series. To enable performance comparisons, we employed 3,137 sequences stratified from the original dataset for each mutational series, making sure the training and test sets for different mutational series have the same size. We used 90% of the mutational series (2,823 sequences) to construct the training dataset for model training on a specific mutational series. To test local performance on the mutational series employed for training, we used the 10% held-out sequences as test set. To test generalization performance, we used all the 3,137 sequences on each series as the test set. The non-deep models (Linear Ridge, MLP and RF) were trained using the scikit-learn Python package, using hyperparameters from previous work [13] and shown in Supplementary Table S1.

4.4 Large Language Model

The results in Supplementary Figure S5 were produced by first embedding our sequences using code provided in the HyenaDNA paper [38]. This lead to embeddings of dimension 12,544 for each 96 nucleotide sequence in our dataset. We did not modify the pretrained weights and employed the embeddings for supervised regression using the three non-deep models considered earlier, without changes to their hyperparameters.

4.5 Feature and ensemble stacking

We conducted feature stacking in the following order: 6 mechanistic features (CAI, MFE1, MFE2, MFE3, AT, MHI) and then 384 one-hot encoding features (Figure 3A). We visualized the feature importance scores for all 56 random forest models across the full 390-dimensional

feature vector to generate Figure 3B. Feature importance scores were computed with the feature_importances_ function for random forest regressors in scikit-learn.

The ensemble stacking models in Figure 3A–B were trained using the StackingRegressor function in scikit-learn. We employed 3 or 4 non-deep estimators as weak learners coupled to final Gradient Boosting regressor. The hyperparameters for each non-deep estimator and the final regressor are shown in Supplementary Table S2.

4.6 Convolutional Neural Network

CNN models were trained on the stacked features shown in Figure 3A using the PyTorch package. Since the CNNs processes one-hot encodings through 2D convolutional layers, (i.e. the inputs are a binary matrix of dimension 4×96), we zero padded the mechanistic features into an 8-dimensional vectors and stacked them into a 4×98 input matrix to the CNN. The network was trained on a batch size of 256 and learning rate 3×10^{-4} using the Adam optimizer [47]. The hyperparameters for CNN models (number of layers, filter size, etc.) were take from previous work [13] and shown in Supplementary Table S3. Training curves for all CNN models can be found in Supplementary Figure S7.

4.7 Graph Neural Network for geometric stacking

The Graph Neural Network was built and trained by the PyTorch and PyTorch Geometric [48] packages. Graph Neural Network models were trained on the mixed encoding of DNA sequences, where the one-hot encoding is used as the node embedding, and the mechanistic features are used as positional features to build the graph connectivity. For every pair of training-testing mutational series, we computed the cosine similarity among all the sequences, and used a k-nearest neighbour graph to convert the similarity matrix into the adjacency matrix of the graph. The k-nearest neighbour graph ($k = 30$) ensures that each sequence node is connected to 30 other nodes with the most similar mechanistic features (largest cosine similarities). To save computational resources and avoid graphs with a large number of nodes, we re-trained the GCN architecture for every pair of training and testing mutational series; this lead to graphs with $2 \times 3,137$ nodes each. In this semi-supervised approach, the trained GCN has access to the features of the sequences in the training and testing sequences, and the labels of the training sequences only; labels of the test sequences were not included in training.

We employed the Graph Convolutional Network (GCN) architecture proposed previously [46]. To further improve the GCN performance, a residual block [49] is also used in the convolutional layers. The GCN was trained as a whole batch, with learning rate 1×10^{-3}, and

using the Adam optimizer. The number of layers (depth) of the network is 5. The detailed hyperparameters for GCN are shown in Table S4.

Data availability

The dataset employed in this study was originally reported in [22]. The data reorganized in a form suitable for machine learning analyses can be found in the Zenodo [50].

Code availability

Python code for model training and data analysis has been provided as supplementary files.

Acknowledgements

YS was supported by the UKRI Biotechnology and Biological Sciences Research Council (BBSRC) grant number BB/T00875X/1. DAO was supported by the United Kingdom Research and Innovation (grant EP/S02431X/1), UKRI Centre for Doctoral Training in Biomedical AI.

Author contributions

YS performed computations, model implementation and data analyses. GK and DAO designed the research, and provided supervision and direction of the work.

Competing interests

The authors have no competing interests.

References

[1] Breitling, R. & Takano, E. Synthetic biology advances for pharmaceutical production. *Current Opinion in Biotechnology* **35**, 46–51 (2015).

[2] Qian, Z.-G., Pan, F. & Xia, X.-X. Synthetic biology for protein-based materials. *Current Opinion in Biotechnology* **65**, 197–204 (2020).

[3] Roell, M.-S. & Zurbriggen, M. D. The impact of synthetic biology for future agriculture and nutrition. *Current Opinion in Biotechnology* **61**, 102–109 (2020).

[4] Gilliot, P.-A. & Gorochowski, T. E. Sequencing enabling design and learning in synthetic biology. *Current Opinion in Chemical Biology* **58**, 54–62 (2020).

[5] Santos, G. *et al.* Model-based genotype-phenotype mapping used to investigate gene signatures of immune sensitivity and resistance in melanoma micrometastasis. *Scientific Reports* **6**, 24967 (2016).

[6] Ritchie, M. D., Holzinger, E. R., Li, R., Pendergrass, S. A. & Kim, D. Methods of integrating data to uncover genotype–phenotype interactions. *Nature Reviews Genetics* **16**, 85–97 (2015).

[7] de Boer, C. G. *et al.* Deciphering eukaryotic gene-regulatory logic with 100 million random promoters. *Nature Biotechnology* **38**, 56–65 (2020).

[8] Kotopka, B. J. & Smolke, C. D. Model-driven generation of artificial yeast promoters. *Nature Communications* **11**, 2113 (2020).

[9] Vaishnav, E. D. *et al.* The evolution, evolvability and engineering of gene regulatory DNA. *Nature* **603**, 455–463 (2022).

[10] Höllerer, S. *et al.* Large-scale DNA-based phenotypic recording and deep learning enable highly accurate sequence-function mapping. *Nature Communications* **11**, 3551 (2020).

[11] Gilliot, P.-A. & Gorochowski, T. E. Transfer learning for cross-context prediction of protein expression from 5'UTR sequence (2023).

[12] Cuperus, J. T. *et al.* Deep learning of the regulatory grammar of yeast $5'$ untranslated regions from 500,000 random sequences. *Genome Research* **27**, 2015–2024 (2017).

[13] Nikolados, E.-M., Wongprommoon, A., Aodha, O. M., Cambray, G. & Oyarzún, D. A. Accuracy and data efficiency in deep learning models of protein expression. *Nature Communications* **13**, 7755 (2022).

[14] Zrimec, J. *et al.* Controlling gene expression with deep generative design of regulatory DNA. *Nature Communications* **13**, 5099 (2022).

[15] Linder, J., Bogard, N., Rosenberg, A. B. & Seelig, G. A Generative Neural Network for Maximizing Fitness and Diversity of Synthetic DNA and Protein Sequences. *Cell Systems* **11**, 49–62.e16 (2020).

[16] Wang, Y. *et al.* Synthetic promoter design in Escherichia coli based on a deep generative network. *Nucleic Acids Research* **48**, 6403–6412 (2020).

[17] Barazandeh, S., Ozden, F., Hincer, A., Seker, U. O. S. & Cicek, A. E. UTRGAN: Learning to Generate 5' UTR Sequences for Optimized Translation Efficiency and Gene Expression. *bioRxiv* **2023.01.30.526198** (2023).

[18] Rafi, A. M. *et al.* Evaluation and optimization of sequence-based gene regulatory deep learning models. *bioRxiv* **2023.04.26.538471** (2023).

[19] Nikolados, E.-M. & Oyarzún, D. A. Deep learning for optimization of protein expression. *Current Opinion in Biotechnology* **81**, 102941 (2023).

[20] Huang, T. *et al.* Analysis and Prediction of Translation Rate Based on Sequence and Functional Features of the mRNA. *PLOS ONE* **6**, e16036 (2011).

[21] Kelsic, E. D. *et al.* RNA Structural Determinants of Optimal Codons Revealed by MAGE-Seq. *Cell Systems* **3**, 563–571.e6 (2016).

[22] Cambray, G., Guimaraes, J. C. & Arkin, A. P. Evaluation of 244,000 synthetic sequences reveals design principles to optimize translation in Escherichia coli. *Nature Biotechnology* **36**, 1005–1015 (2018).

[23] Kudla, G., Murray, A. W., Tollervey, D. & Plotkin, J. B. Coding-Sequence Determinants of Gene Expression in Escherichia coli. *Science* **324**, 255–258 (2009).

[24] Quax, T. E., Claassens, N. J., Söll, D. & van der Oost, J. Codon Bias as a Means to Fine-Tune Gene Expression. *Molecular Cell* **59**, 149–161 (2015).

[25] Brandis, G. & Hughes, D. The Selective Advantage of Synonymous Codon Usage Bias in Salmonella. *PLOS Genetics* **12**, e1005926 (2016).

[26] Knöppel, A., Näsvall, J. & Andersson, D. I. Compensating the Fitness Costs of Synonymous Mutations. *Molecular Biology and Evolution* **33**, 1461–1477 (2016).

[27] Mittal, P., Brindle, J., Stephen, J., Plotkin, J. B. & Kudla, G. Codon usage influences fitness through RNA toxicity. *Proceedings of the National Academy of Sciences* **115**, 8639–8644 (2018).

[28] Jia, Q. *et al.* A "GC-rich" method for mammalian gene expression: A dominant role of non-coding DNA GC content in regulation of mammalian gene expression. *Science China Life Sciences* **53**, 94–100 (2010).

[29] Raghavan, R., Kelkar, Y. D. & Ochman, H. A selective force favoring increased G+C content in bacterial genes. *Proceedings of the National Academy of Sciences* **109**, 14504–14507 (2012).

[30] Frumkin, I. *et al.* Gene Architectures that Minimize Cost of Gene Expression. *Molecular Cell* **65**, 142–153 (2017).

[31] Angenent-Mari, N. M., Garruss, A. S., Soenksen, L. R., Church, G. & Collins, J. J. A deep learning approach to programmable RNA switches. *Nature Communications* **11**, 5057 (2020).

[32] Nieuwkoop, T. *et al.* Revealing determinants of translation efficiency via whole-gene codon randomization and machine learning. *Nucleic Acids Research* **51**, 2363–2376 (2023).

[33] Sasse, A. *et al.* Benchmarking of deep neural networks for predicting personal gene expression from DNA sequence highlights shortcomings. *bioRxiv* **2023.03.16.532969** (2023).

[34] Karollus, A., Mauermeier, T. & Gagneur, J. Current sequence-based models capture gene expression determinants in promoters but mostly ignore distal enhancers. *Genome Biology* **24**, 56 (2023).

[35] McInnes, L., Healy, J. & Melville, J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv* **1802.03426** (2020).

[36] Luecken, M. D. & Theis, F. J. Current best practices in single-cell RNA-seq analysis: a tutorial. *Molecular Systems Biology* **15**, e8746 (2019).

[37] Hastie, T., Tibshirani, R. & Friedman, J. *The Elements of Statistical Learning* (2009).

[38] Nguyen, E. *et al.* HyenaDNA: Long-Range Genomic Sequence Modeling at Single Nucleotide Resolution. *arXiv* **2306.15794** (2023).

[39] Chen, J. *et al.* Interpretable RNA Foundation Model from Unannotated Data for Highly Accurate RNA Structure and Function Predictions. *arXiv* **2204.00300** (2022).

[40] Sagi, O. & Rokach, L. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **8**, e1249 (2018).

[41] Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A. & Vandergheynst, P. Geometric deep learning: going beyond Euclidean data. *IEEE Signal Processing Magazine* **34**, 18–42 (2017).

[42] Wu, L. *et al.* Graph Neural Networks for Natural Language Processing: A Survey. *arXiv* **2106.06090** (2022).

[43] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O. & Dahl, G. E. Neural Message Passing for Quantum Chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, 1263–1272 (2017).

[44] Smer-Barreto, V. *et al.* Discovery of senolytics using machine learning. *Nature Communications* **14**, 3445 (2023).

[45] Lam, R. *et al.* Learning skillful medium-range global weather forecasting. *Science* **382.6677**, 1416–1421 (2023).

[46] Kipf, T. N. & Welling, M. Semi-Supervised Classification with Graph Convolutional Networks (2017).

[47] Kingma, D. P. & Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **1412.6980** (2017).

[48] Fey, M. & Lenssen, J. E. Fast Graph Representation Learning with PyTorch Geometric. *arXiv* **1903.02428** (2019).

[49] He, K., Zhang, X., Ren, S. & Sun, J. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778 (2016).

[50] Nikolados, E.-M., Wongprommoon, A., Mac Aodha, O., Cambray, G. & Oyarzún, D. A. Code and data for "Accuracy and data efficiency in deep learning models of protein expression". *Zenodo https://doi.org/10.5281/zenodo.7273952* (2022).

Figures

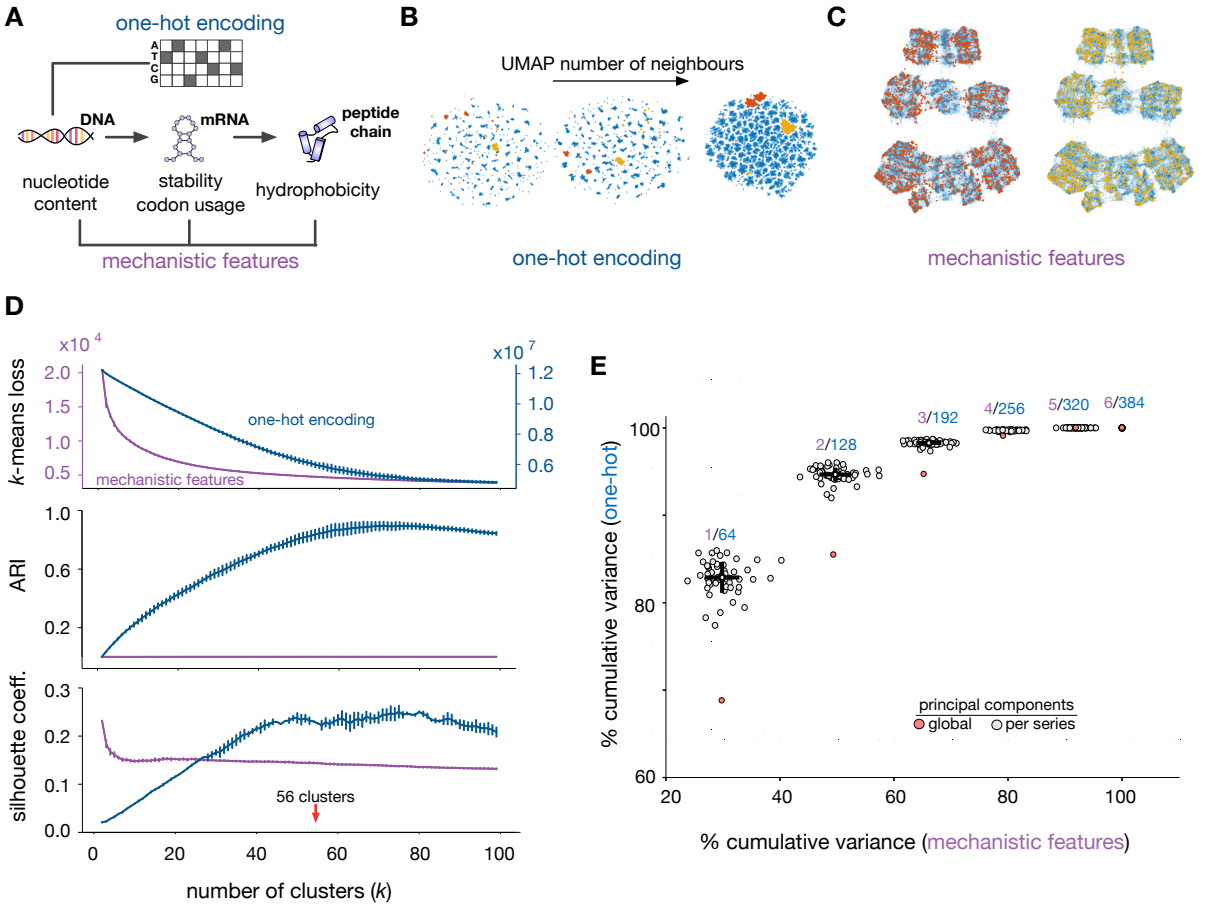

Figure 1: **Shape and coverage of a DNA variant library in different sequence representation spaces.** (**A**) We focus on DNA sequences described by one-hot encodings and different mechanistic sequence features such as nucleotide content, stability of mRNA secondary structure, codon bias and hydrophobicity of the peptide chain. (**B**) Two dimensional UMAP [35] visualization of the whole library from Cambray *et al* [22]. A total of N=175,672 sequences of length 96nt are represented by one-hot encodings of dimension 384. Two mutational series are highlighted in red and yellow, suggesting some degree of clustering in the one-hot encoded feature space. (**C**) Two dimensional UMAP visualization of the same library as in panel A, but using the 6-dimensional mechanistic feature space. This representation suggests a qualitatively different clustering structure, and the selected mutational series (red, yellow) appear homogeneously distributed in the mechanistic feature space; UMAP visualizations for other combinations of hyperparameters can be found in Supplementary Figure S1–S2. (**D**) Top: Clustering of the library using the k-means algorithm; plot shows the optimal k-means score for increasing number of clusters. Middle: Adjusted Rand index (ARI) between k-means clusters and the label of the mutational series, computed for all clustering repeats. Plots for k-means loss and ARI show mean $\pm$ s.d. computed across 25 repeats with different random seeds. Bottom: Silhouette scores showing how similar each variant is to its own cluster as compared to other clusters; silhouette scores are shown as mean $\pm$ s.d. computed across 10 repeats with different random seeds. (**E**) Representation power of one-hot encodings and mechanistic features. We computed the explained variance using Principal Component Analysis (PCA) applied to the whole dataset (global) and each individual mutational series. Circles denote the cumulative variance explained by an increasing number of PCs; for comparison, the cumulative variance of one-hot encodings were computed every 64 PCs.

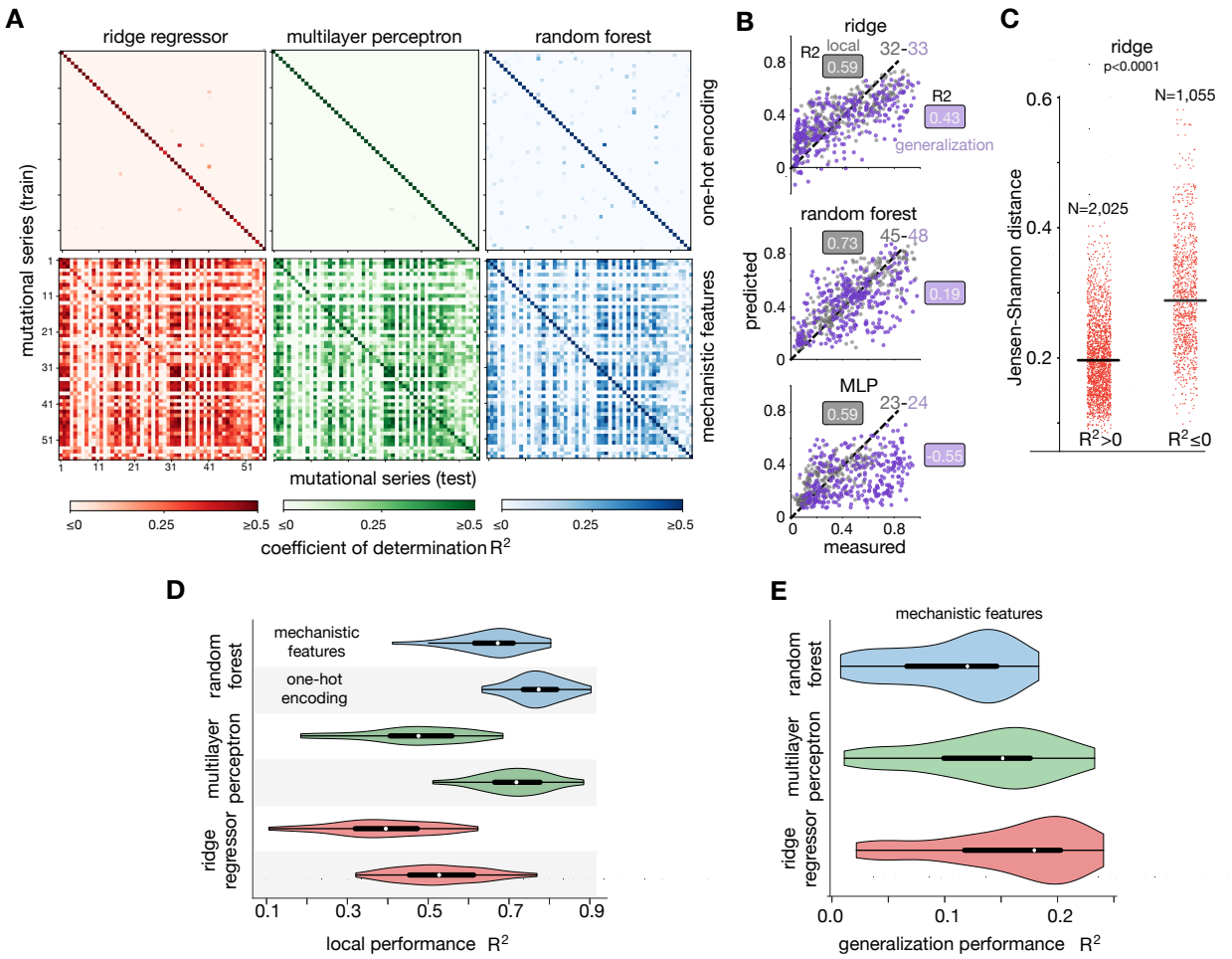

Figure 2: **Prediction accuracy of models trained on one-hot encodings and mechanistic features.** (**A**) Generalization performance computed via cross-series testing. Three different models (ridge regression, multilayer perceptron and random forests) are trained on each mutational series, and tested on all other mutational series. Heatmaps show the coefficient of determination (R^2) between ground truth sfGFP fluorescence measurements and model predictions. Negative R^2 scores have been zeroed out for clarity and represent an unsuitable model structure with worse performance than a naïve regressor that predicts the average expression for all samples. (**B**) Representative models with strong local predictions and varying generalization performance. The grey and purple dots show measured and predicted protein expression in a held-out test set take from same and different mutational series, respectively. Grey boxes show local performance R^2 score computed on the same mutational series employed for training (series label in grey). Purple boxes show the generalization performance R^2 score on another mutational series not employed for training (series label in purple). (**C**) Jensen-Shannon distance between sfGFP distributions across all pairs of mutational series, stratified according to the R^2 scores. Models with negative R^2 scores are associated with larger phenotypic differences between the sequence space employed for training and testing; Mann-Whitney U-test p = 6.3×10^{-186}. (**D**) Local performance of different models trained on mechanistic features and one-hot encoding; violins show the values in the diagonal of the heatmaps in panel A. (**E**) Generalization performance of models trained on mechanistic features, computed as the row-wise average of the heatmaps in panel A excluding the diagonal entry.

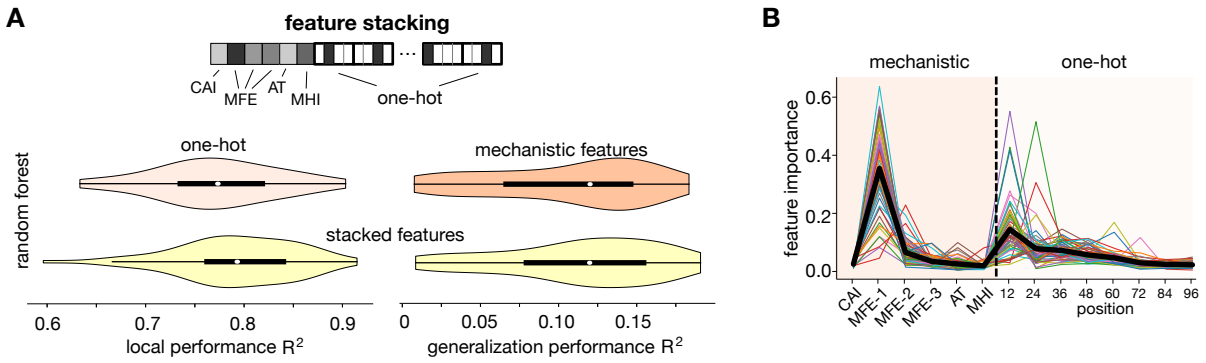

Figure 3: **Integration of sequence representations with feature stacking.** (**A**) For feature stacking, the 6-dimensional mechanistic features were merged with a 384-dimensional one-hot encoded binary vector. A random forest model (RF is trained on stacked feature vectors for each mutational series. Violin plots show the local performance and generalization performance averaged across the 55 mutational series not employed for training, both compared to the performance in Figure 2D–E when trained mechanistic feature or one-hot encodings in isolation. (**B**) Feature importance scores of RF models trained on stacked features. For clarity, scores for sequence nucleotides have been aggregated in windows of 12 nucleotide positions (48 one-hot encoding positions). Each colored line shows the feature importance of one model out of the total 56 models (one per mutational series), and the black thick line shows the mean feature importance across all models at each position. Details on the computation of importance scores can be found in the Methods.

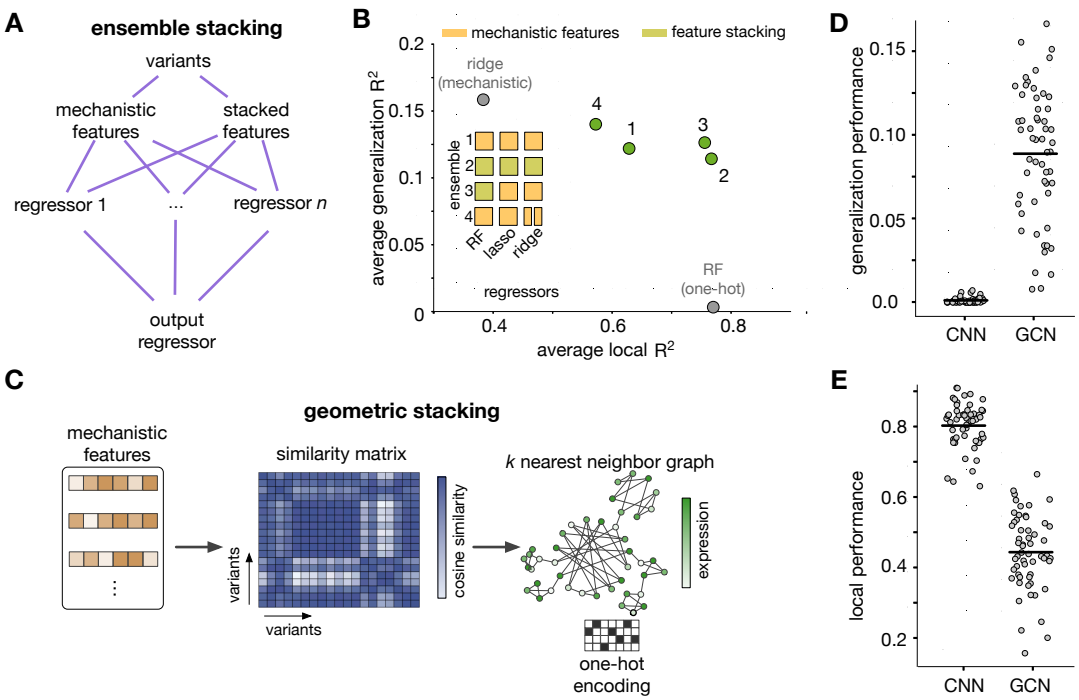

Figure 4: **Integration of DNA sequence representations with ensemble and geometric stacking.** (**A**) Ensemble stacking combines weak models trained on different feature sets into a single predictor; the final output is produced by a separate output regressor. (**B**) Local and generalization performance of four representative ensembles models, in comparison to the best performers in Figure 2D–E (gray). Local R^2 scores have been averaged across the 56 models; generalization R^2 scores have been averaged across the 55 test sets (as in Figure 2E) and the 56 models. The inset shows the four representative ensembles; in all cases the output regressor is a Gradient Boosting model. (**C**) Geometric stacking using graph convolutional neural networks (GCN [46]). From a library of variants, a graph is built with nodes representing sequence variants and edges weighed by the cosine similarity between variants in the mechanistic feature space. We employed one-hot encoded binary matrices as node features, and label each variant node with the measured sfGFP fluorescence. For training, the connectivity of the graph was sparsified via a k-nearest neighbours graph. (**D**,**E**) The GCN shows improved generalization performance as compared to another popular deep learning architecture (convolutional neural network, CNN) trained on stacked features, at the cost of a poorer local predictions. Panels D and E show the local and generalization R^2 scores for each of the 56 models (one per mutational series); black line denotes the average across all models. Details on model training and hyperparameters can be found in Methods.