Deep Fitness Inference for Drug Discovery with Directed Evolution

Nathaniel Diamant,[1] **Ziqing Lu,**[1] **Christina Helmling,**[2] **Kangway Chuang,**[1]
Christian Cunningham,[2] **Tommaso Biancalani,**[1] **Gabriele Scalia,**[1†*] **Max Shen**[1†*]
[1]Department of Artificial Intelligence and Machine Learning
[2]Department of Peptide Therapeutics,
Genentech Research and Early Development
[†]{scalia.gabriele,shen.max}@gene.com

Abstract

Directed evolution, with iterated mutation and human-designed selection, is a powerful approach for drug discovery. Here, we establish a fitness inference problem given time series DNA sequencing data. We describe maximum likelihood solutions for the nonlinear dynamical system induced by fitness-based competition. Our approach learns from multiple time series rounds in a principled manner, in contrast to prior work focused on two-round enrichment prediction. While fitness inference does not require deep learning in principle, we show that inferring fitness while jointly learning a sequence-to-fitness transformer (DeepFitness) improves performance over a non-deep baseline, and a two-round enrichment baseline. Finally, we highlight how DeepFitness can improve the diversity of the discovered hits in a directed evolution experiment.

1 Introduction

Directed evolution can discover drug-like biomolecules with novel activities of interest. There is substantial interest in augmenting laboratory directed evolution with machine learning to improve the activity and diversity of discovered hits. In many directed evolution experiments, the main readout is DNA sequencing data which measures the frequency of competing biomolecules in the evolving population. However, ***biomolecule frequency* is not equal to *biological activity***, which is what we actually care about: to discover hits, we would like to sort biomolecules by their biological activity. This poses an inference problem: given biomolecule frequencies, infer their biological activities.

In our DNA-encoded library (DEL) setting, a DNA sequence (alphabet size = 4) is translated into a peptide, or small protein (alphabet size = 20), using a known codon table. Peptides are physically connected to their DNA sequences by a linker. Each time step, round, or generation of directed evolution comprises:

- **Activity-based selection.** A set of peptide-linker-DNA compounds, typically more than 10^{13}–10^{14}, are challenged to bind to target proteins immobilized on a solid support. Washing steps remove peptide-linker-DNA compounds with weak binding strength. Survivors are extracted.
- **Mutation and propagation.** The DNA sequence, and thus peptide identity, of binding compounds are determined by DNA sequencing. Peptide-linker-DNA compounds are propagated by polymerase chain reaction (PCR), a multi-step process that doubles DNA molecules each step by duplication. PCR introduces 1×10^{-5} mutations per nucleotide per step.

[*]Equally contributed, authors acknowledge any reordering of * symbol'd authors

NeurIPS 2021 AI for Science Workshop.

2 A Model of Evolutionary Dynamics

Here, we introduce a model of evolutionary dynamics to motivate our fitness inference task.

Absolute fitness. Let G be the number of unique genotypes in a population. At time or round t, denote $\mathbf{n}_t \in \mathbb{N}^G$ as the vector of the number of each genotype, and $N_t = \sum_{i=1}^{G} n_{t,i}$ as the total count. We use repeated subscripts for indexing: $n_{t,i}$ is the count of the i-th genotype at time t. In population genetics, absolute fitness $\mathbf{W} \in (\mathbb{R}^+)^G$ is defined as

$$n_{t+1,i} = W_i n_{t,i}. \tag{1}$$

In our setting, $\mathbf{W}$ describes the change in genotype counts due to differences in binding strength.

Towards fitness inference. We aim to infer absolute fitness from DNA sequencing time series data, but this is not quite feasible. Denote $\mathbf{m}_t$ as the count of sequencing reads, and $M_t = \sum_i m_{t,i}$ as the total read depth. DNA sequencing can be described with $\mathbf{m}_t \sim \texttt{Multinomial}\left(\mathbf{n}_t/N_t, M_t\right)$ so we can only estimate $\mathbf{p}_t = \mathbf{n}_t/N_t$ which are genotype frequencies. To proceed, we note that 1 can be rewritten as (Proposition A1):

$$p_{t+1,i} = \frac{W_i}{\sum_j W_j p_{t,j}} p_{t,i}. \tag{2}$$

Fitness inference. Due to the unidentifiability of absolute fitness (Proposition A2), we set up our inference problem as: given $\mathbf{m}_t$ for $t = 0, 1, \ldots, T$, infer *relative fitness* $\mathbf{w} = c\mathbf{W}$ where $c > 0$ is unknown. Eq. 2 describes a non-linear dynamical system which can be simulated forward in time given an initial $\mathbf{p}_0$ and relative fitness $\mathbf{w}$. This simple model ignores genetic drift and lacks a mutational process. Rewriting Eq. 2 using $\mathbf{w}$ in matrix notation, we define a fitness model's predictions for round $t+1$ given previous round frequencies $\mathbf{p}_t$ as

$$\hat{\mathbf{p}}_{t+1} = \frac{\mathbf{w}}{\mathbf{w} \cdot \mathbf{p}_t} \odot \mathbf{p}_t \tag{3}$$

where $\odot$ denotes pointwise multiplication. Our method optimizes over $\mathbf{w}$ so that $\hat{\mathbf{p}}_{t+1}$ reflects $\mathbf{p}_{t+1}$ according to different noise models.

Related work. Many prior works learn enrichment scores [1; 2; 3; 4], which is equivalent to relative fitness with two time points, but not with > 2 time points (Proposition A3). In contrast, our work presents a principled method for datasets with > 2 time points. Ma et al. [5] train a neural network on a single round of data using a negative binomial loss, whereas our work uses fitness and multivariate losses to take into account the competition between genotypes. For hit prioritization, biologists conventionally rank compounds by frequency in the last time point [6]. However, the motto is "survival of the fittest", not "survival of the most frequent" — ranking by fitness can identify low-frequency, high-fitness "rising stars" otherwise missed.

Shen et al. [6] uses the fitness inference framework with a single fitness parameter per genotype. The fitness parameters are trained using the KL divergence between true and predicted frequencies and a regularization term that encourages unobserved frequencies to match those resultant from natural selection. We extend the fitness inference component by parameterizing fitness using a neural network enabling predictions on unseen genotypes. We also replace the KL divergence with likelihood based losses, which lead to improved performance in our experiments.

3 Methodology

Fitness inference can be split into two components:

1. A ***differentiable parameterization*** of $\mathbf{w}$ to map from genotype to fitness.
2. A differentiable ***loss function*** used to optimize how well the predicted frequencies, $\hat{\mathbf{p}}_{t+1}$ from Eq. 3, match the observed frequencies $\mathbf{p}_{t+1}$.

Given these two components, fitness can be inferred using first order optimization.

Loss functions. We developed two probabilistic loss functions based on a multinomial (MN) and a Dirichlet-multinomial (DM) model of $\mathbf{m}_{t+1}$. The MN loss is the negative log-likelihood of

$$\mathbf{m}_{t+1} \sim \text{MN}(\mathbf{p} = \hat{\mathbf{p}}_{t+1}, C = M_{t+1})$$

where $\mathbf{p}$ are the event probabilities and C is the total event count. The DM loss is an over-dispersed version of the MN that allows increased uncertainty when there are fewer read counts in round t. The DM loss is the negative log-likelihood of

$$\mathbf{m}_{t+1} \sim \text{DM}(\boldsymbol{\alpha} = M_t \hat{\mathbf{p}}_{t+1}, C = M_{t+1})$$

where $\boldsymbol{\alpha}$ are the concentration parameters. For more discussion of the DM loss see Appendix B.

Fitness parameterizations We experimented with two strategies for parameterizing fitness. The first approach was to define one log relative fitness ($\log w$) value per genotype, which we call *per-genotype fitness*. The second approach was to use a neural network to map from genotype to $\log w$, which we call *DeepFitness*. We converted each amino acid sequence into a sequence of Morgan fingerprints [7] with substructure counts so that similar amino acids were mapped to similar embeddings. We used a transformer-like encoder [8] with multi-head attention (details in Appendix C) to map from the embedded amino acid sequences to $\log w$.

4 Results

The directed evolution experiment analyzed in the following includes seven rounds (the first round is the random library), and three off-target rounds following rounds five, six and seven. The directed evolution experiment selects for activity for the target, and the library includes 8-mers (cyclic peptides). For a description of the data preparation, see Appendix E.

Baseline models. We compared our fitness inference approach to three baselines. *Prior round frequency* assumes that enrichment is equal to the frequency in the final time point of the training data. *Prior round enrichment* assumes that the heldout round's enrichment is equal to the penultimate round's enrichment. *Enrichment regression* seeks to minimize the MSE between predicted and actual enrichment and works with the per-genotype and deep parameterizations.

Generalization to heldout rounds and genotypes. We evaluated all models ability to predict the enrichment of the heldout final round T trained on time points $1, \ldots, T-1$, for the training sequences (Table 1, *Seen Molecules*) and the ability of DeepFitness to predict the enrichment in round T on heldout genotypes (Table 1, *Unseen Molecules*). Pearson-r is reported to measure the agreement between actual and predicted enrichment.

Table 1: Enrichment prediction Pearson correlation (*r*). Bold indicates the difference from the best results was not statistically significant with three random seeds.

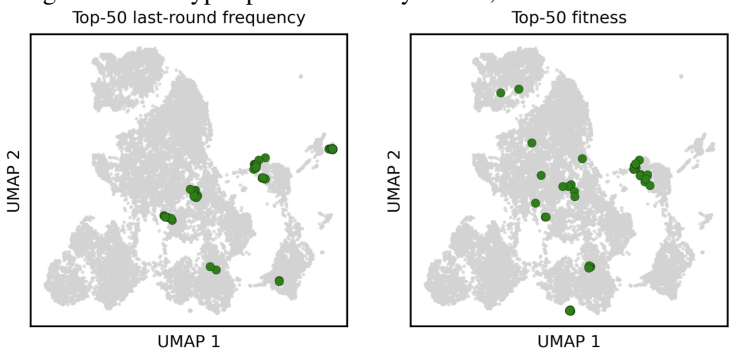
Parameterization	r Seen molecules (↑)	r Unseen molecules (↑)
Prior round frequency	0.03	-
Prior round enrichment	0.29	-
Per-genotype (Enrichment regression)	$0.09 \pm < 0.01$	-
Per-genotype (MN)	$\mathbf{0.33 \pm 0.05}$	-
Per-genotype (DM)	$\mathbf{0.39 \pm 0.01}$	-
DeepFitness (Enrichment regression)	0.25 ± 0.03	0.22 ± 0.04
DeepFitness (MN)	0.16 ± 0.01	0.10 ± 0.02
DeepFitness (DM)	$\mathbf{0.40 \pm 0.01}$	$\mathbf{0.45 \pm 0.04}$

The results show the benefits of using the fitness inference framework compared to all the baselines. The prior-round frequency and enrichment baselines predicted enrichment poorly compared to our fitness based methods, demonstrating that enrichment is driven by the most fit accounting for all rounds, not just the last round. The strength of the prior round enrichment baseline is explained by the fact that when the average fitness is similar from one round to the next (as it can be in later

rounds), the enrichment in the prior round is likely to be similar to enrichment in the following round. Fitness inference with the DM loss also outperformed enrichment regression when using either parameterization, which shows the benefits of probabilistic losses built on the fitness framework. Between the two fitness based losses, the DM loss yielded improved results over the multinomial loss, which struggled with the DeepFitness parameterization. This indicates that accounting for prior round read counts improves model performance. The DeepFitness DM model had higher average performance on test data than training data, which could indicate under-fitting, but is most likely due to the high variance of the metric on the test data.

Discovery of novel genotypes. We assessed the usefulness of the estimated fitness to highlight novel and diverse genotypes not easily detected through the standard last-round frequency pipeline, wherein the genotypes with high frequency in the last selection round are considered to be the most fit ([9; 10; 11; 12; 13; 14; 15]). We embedded genotypes in a meaningful "genotype space" learned in an unsupervised fashion through a VAE and embedded with UMAP (Appendix C) and compared the high-frequency and high-fitness regions in this space (Fig. 1). As shown, while several areas share high frequency and high fitness values, we can also identify regions of genotypes characterized by high fitness but low frequency (and vice-versa). Such regions can help improve the number and diversity of the identified biomolecules. We observe that the inferred fitness helps selecting genotypes distant (i.e., less similar) to those selected through frequency.

Figure 1: Genotype space colored by fitness, and fitness winners.



5 Discussion

We introduced fitness inference for directed evolution time series data with on and off-target selection rounds. We developed and compared two parameterizations of fitness, per-genotype and DeepFitness, with two different probabilistic loss functions. With both parameterizations, fitness inference with the DM loss significantly improved results compared to baselines that do not account for the competition induced in a multi-round selection experiment. Compared to the per-genotype parameterization, DeepFitness enabled fitness prediction on unseen genotypes learning a genotype to fitness mapping.

We assessed the diversity of the genotypes with the highest fitness compared to those with the highest last-round frequency. We found that, while the fitness-based selection re-discovers genotypes similar to those selected solely through frequency, it also highlights genotypes in unexplored regions, thus improving the overall diversity. As each directed evolution experiment is heavily influenced by the target, the number of rounds, and the covariates, future work should evaluate our approach in a broader range of settings. Additionally, evaluation is currently limited to an inherently noisy measurement, enrichment, as ground truth. Future work should include more precise protein assays and evaluate the broader impact of fitness-based selection on different properties. We envision using DeepFitness as a more general and robust proxy of binding strength to better identify and optimize biomolecules starting from directed evolution experiments.

Acknowledgments and Disclosure of Funding

The authors thank Kam Hon Hoi for his valuable feedback on the results and the analyses.

Funding: The authors are Roche shareholders.

References

[1] Kevin McCloskey, Eric A Sigel, Steven Kearnes, Ling Xue, Xia Tian, Dennis Moccia, Diana Gikunju, Sana Bazzaz, Betty Chan, Matthew A Clark, et al. Machine learning on dna-encoded libraries: a new paradigm for hit finding. *Journal of Medicinal Chemistry*, 63(16):8857–8866, 2020.

[2] Kehang Han, Steven Kearnes, Jin Xu, Wen Torng, and JW Feng. Improving hit-finding: Multilabel neural architecture with del. In *NeurIPS 2021 AI for Science Workshop*, 2021.

[3] Ge Liu, Haoyang Zeng, Jonas Mueller, Brandon Carter, Ziheng Wang, Jonas Schilz, Geraldine Horny, Michael E Birnbaum, Stefan Ewert, and David K Gifford. Antibody complementarity determining region design using high-capacity machine learning. *Bioinformatics*, 36(7):2126–2133, 11 2019. ISSN 1367-4803. doi: 10.1093/bioinformatics/btz895. URL `https://doi.org/10.1093/bioinformatics/btz895`.

[4] Katherine S Lim, Andrew G Reidenbach, Bruce K Hua, Jeremy W Mason, Christopher J Gerry, Paul A Clemons, and Connor W Coley. Machine learning on dna-encoded library count data using an uncertainty-aware probabilistic loss function. *Journal of Chemical Information and Modeling*, 2022.

[5] Ralph Ma, Gabriel HS Dreiman, Fiorella Ruggiu, Adam Joseph Riesselman, Bowen Liu, Keith James, Mohammad Sultan, and Daphne Koller. Regression modeling on dna encoded libraries. In *NeurIPS 2021 AI for Science Workshop*, 2021.

[6] Max W Shen, Kevin T Zhao, and David R Liu. Reconstruction of evolving gene variants and fitness from short sequencing reads. *Nat. Chem. Biol.*, 17(11):1188–1198, November 2021.

[7] H. L. Morgan. The generation of a unique machine description for chemical structures-a technique developed at chemical abstracts service. *Journal of Chemical Documentation*, 5(2): 107–113, 1965. doi: 10.1021/c160017a018.

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[9] Benjamin W. Thuronyi, Luke W. Koblan, Jonathan M. Levy, Wei-Hsi Yeh, Christine Zheng, Gregory A. Newby, Christopher Wilson, Mantu Bhaumik, Olga Shubina-Oleinik, Jeffrey R. Holt, and David R. Liu. Continuous evolution of base editors with expanded target compatibility and improved activity. *Nature Biotechnology*, 37(9):1070–1079, September 2019. ISSN 1546-1696. doi: 10.1038/s41587-019-0193-0.

[10] Michelle F. Richter, Kevin T. Zhao, Elliot Eton, Audrone Lapinaite, Gregory A. Newby, Benjamin W. Thuronyi, Christopher Wilson, Luke W. Koblan, Jing Zeng, Daniel E. Bauer, Jennifer A. Doudna, and David R. Liu. Phage-assisted evolution of an adenine base editor with improved Cas domain compatibility and activity. *Nature biotechnology*, 38(7):883–891, July 2020. ISSN 1087-0156. doi: 10.1038/s41587-020-0453-z. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7357821/`.

[11] Shannon M. Miller, Tina Wang, Peyton B. Randolph, Mandana Arbab, Max W. Shen, Tony P. Huang, Zaneta Matuszek, Gregory A. Newby, Holly A. Rees, and David R. Liu. Continuous evolution of SpCas9 variants compatible with non-G PAMs. *Nature biotechnology*, 38(4): 471–481, April 2020. ISSN 1087-0156. doi: 10.1038/s41587-020-0412-8. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7145744/`.

[12] Johnny H. Hu, Shannon M. Miller, Maarten H. Geurts, Weixin Tang, Liwei Chen, Ning Sun, Christina M. Zeina, Xue Gao, Holly A. Rees, Zhi Lin, and David R. Liu. Evolved Cas9 variants with broad PAM compatibility and high DNA specificity. *Nature*, 556(7699):57–63, April 2018. ISSN 0028-0836. doi: 10.1038/nature26155. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5951633/`.

[13] Arjun Ravikumar, Garri A. Arzumanyan, Muaeen K.A. Obadi, Alex A. Javanpour, and Chang C. Liu. Scalable, continuous evolution of genes at mutation rates above genomic error thresholds. *Cell*, 175(7):1946–1957.e13, December 2018. ISSN 0092-8674. doi: 10.1016/j.cell.2018.10.021. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6343851/`.

[14] Kevin M. Esvelt, Jacob C. Carlson, and David R. Liu. A System for the Continuous Directed Evolution of Biomolecules. *Nature*, 472(7344):499–503, April 2011. ISSN 0028-0836. doi: 10.1038/nature09929. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3084352/`.

[15] Ahmed H. Badran, Victor M. Guzov, Qing Huai, Melissa M. Kemp, Prashanth Vishwanath, Wendy Kain, Autumn M. Nance, Artem Evdokimov, Farhad Moshiri, Keith H. Turner, Ping Wang, Thomas Malvar, and David R. Liu. Continuous evolution of B. thuringiensis toxins overcomes insect resistance. *Nature*, 533(7601):58–63, May 2016. ISSN 0028-0836. doi: 10.1038/nature17938. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4865400/`.

[16] Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D. Goodman. Pyro: Deep Universal Probabilistic Programming. *Journal of Machine Learning Research*, 2018.

A Proofs

Proposition 1 (Relative fitness)**.** *Suppose* $n_{t+1,i} = W_i n_{t,i}$*. Then,*

$$p_{t+1,i} = \frac{W_i}{\sum_j W_j p_{t,j}} p_{t,i} \tag{4}$$

Proof. Using $N_{t+1} = \sum_i n_{t+1,i} = \sum_i W_i n_{t,i}$, we have

$$\begin{aligned}
n_{t+1,i} &= W_i n_{t,i} \\
\frac{n_{t+1,i}}{N_{t+1}} &= \frac{W_i}{N_{t+1}} n_{t,i} && \text{(divide by } N_{t+1}\text{)} \\
p_{t+1,i} &= \frac{W_i}{\sum_j W_j n_{t,j}} n_{t,i} && \text{(replace with definitions)} \\
p_{t+1,i} &= \frac{N_t W_i}{\sum_j W_j n_{t,j}} \frac{n_{t,i}}{N_t} && \text{(multiply by } N_t/N_t\text{)} \\
p_{t+1,i} &= \frac{W_i}{\sum_j W_j \frac{n_{t,j}}{N_t}} p_{t,i} && \text{(move } N_t \text{ through fraction)} \\
p_{t+1,i} &= \frac{W_i}{\sum_j W_j p_{t,j}} p_{t,i} && \text{(replace with definitions)}
\end{aligned}$$

□

Proposition 2 (Identifiability up to proportionality)**.** *Suppose*

$$p_{t+1,i} = \frac{W_i}{\sum_j W_j p_{t,j}} p_{t,i}. \tag{5}$$

Then, for any positive scalar c,

$$p_{t+1,i} = \frac{cW_i}{\sum_j cW_j p_{t,j}} p_{t,i}. \tag{6}$$

Proof.

$$\begin{aligned}
p_{t+1,i} &= \frac{W_i}{\sum_j W_j p_{t,j}} p_{t,i} \\
p_{t+1,i} &= \frac{cW_i}{c\sum_j W_j p_{t,j}} p_{t,i} \\
p_{t+1,i} &= \frac{cW_i}{\sum_j cW_j p_{t,j}} p_{t,i}
\end{aligned}$$

□

Proposition 3 (Enrichment is not equivalent to relative fitness with > 2 time points)**.** *Suppose*

$$p_{t+1,i} = \frac{W_i}{\sum_j W_j p_{t,j}} p_{t,i}. \tag{7}$$

for some fitness $\boldsymbol{W}$ *holds for* $\boldsymbol{p}_0 \neq \boldsymbol{p}_1 \neq \boldsymbol{p}_2$*. Then, enrichments* $\boldsymbol{p}_2/\boldsymbol{p}_1 \neq \boldsymbol{p}_1/\boldsymbol{p}_0$*.*

Proof. Rearranging the relative fitness equation, each enrichment is equal to a fraction with $\mathbf{W}$ in the numerator, and $\sum_j W_j p_{t,j}$ in the denominator. The numerator is the same for $\mathbf{p}_2/\mathbf{p}_1, \mathbf{p}_1/\mathbf{p}_0$, but the denominator is not the same. So, the enrichments are not equal. □

Remark Enrichment is equivalent to relative fitness with 2 time points. Specifically, algebraic rewriting of equation 2 shows that enrichment is proportional to absolute fitness with 2 time points. Relative fitness is also proportional to absolute fitness by definitions in section 2.

Remark. When there are more than 2 time points where each pair of time points are consistent with the same $\mathbf{W}$, computing enrichment will yield different enrichment values for each pair of time points. Thus, enrichment cannot be equivalent to relative fitness.

Proposition 4 (Upper bound for relative off-target activity)**.** *Suppose*

$$\underbrace{c_1\mathbf{W}_{\texttt{total}}}_{\textit{available}} = \underbrace{c_1\mathbf{W}_{\texttt{on}}}_{\textit{goal}} + \underbrace{\frac{c_1}{c_2}}_{\textit{unknown}} \underbrace{c_2\mathbf{W}_{\texttt{off}}}_{\textit{available}}. \tag{8}$$

Then,

$$\frac{\min \mathbf{w}_{\textit{total}}}{\max \mathbf{w}_{\textit{off}}} = \frac{\min c_1\mathbf{W}_{\texttt{total}}}{\max c_2\mathbf{W}_{\texttt{off}}} \geq \frac{c_1}{c_2}. \tag{9}$$

Proof. Using the property that absolute fitness must be non-negative,

$$\begin{aligned} c_1\mathbf{W}_{\texttt{total}} &= c_1\mathbf{W}_{\texttt{on}} + \frac{c_1}{c_2}c_2\mathbf{W}_{\texttt{off}} \\ c_1\mathbf{W}_{\texttt{total}} &\geq 0 + \frac{c_1}{c_2}c_2\mathbf{W}_{\texttt{off}} \\ \frac{c_1\mathbf{W}_{\texttt{total}}}{c_2\mathbf{W}_{\texttt{off}}} &\geq \frac{c_1}{c_2}. \end{aligned} \tag{10}$$

This relationship holds for each entry in the vector.

□

B Dirichlet-multinomial (DM) loss

By construction, relative fitness (Eq. 2) only accounts for the frequencies and not the read counts at round t. However, in practice, the intrinsic high assay noise makes the measured enrichment less reliable for lower counts. The multinomial distribution previously described does not account for prior round counts, preventing modeling of this critical aspect. Therefore, we introduce an over-dispersed multinomial distribution, the Dirichlet-multinomial (DM), to account for the increased difficulty (i.e., lower confidence) of predicting $\mathbf{p}_{t+1}$ from $\mathbf{p}_t$ when the total read counts are lower.

The DM distribution is the posterior predictive distribution of a Dirichlet prior on a multinomial model of count data. When updating a Dirichlet prior with concentration $\boldsymbol{\alpha}$ with observed counts per category $\mathbf{k}$, the posterior is Dirichlet distributed with concentration $\boldsymbol{\alpha} + \mathbf{k}$. The total concentration increases from $\sum_i \alpha_i$ to $\sum_i(\alpha_i + k_i)$. Based on this observation, we consider a DM likelihood with $\boldsymbol{\alpha} = M_t\hat{\mathbf{p}}_{t+1}$, with $\hat{\mathbf{p}}_{t+1}$ computed as in Eq. 3, and M_t being the total read counts. Note that the expected frequencies of this distribution are $\hat{\mathbf{p}}_{t+1}$ (as in the multinomial distribution), but the total concentration is M_t, thus accounting for the total counts. This yields the log-likelihood for $\mathbf{m}_{t+1}$:

$$\log p(\mathbf{m}_{t+1}|\mathbf{m}_t,\ \mathbf{w}) = \log \text{DM}[\boldsymbol{\alpha} = M_t\hat{\mathbf{p}}_{t+1}, C = M_{t+1}](\mathbf{m}_{t+1}) \tag{11}$$

where $\text{DM}[\boldsymbol{\alpha}, C](\mathbf{m}_{t+1})$ is the Dirichlet-multinomial likelihood of counts $\mathbf{m}_{t+1}$ with concentration parameters $\boldsymbol{\alpha}$ and total count C. We used Pyro [16] to differentiably calculate the DM loss as the negative DM log-likelihood.

C Model Architecture and Hyperparameters

The DeepFitness transformer architecture had three eight-head attention layers with 64 hidden dimensions each with a dropout rate of 10%. The attention layers were followed by an multi-layer perceptron (MLP) with two layers of size 64 and then one for the output. The MLP layers were linear followed by batch norm followed by ReLU then 10% dropout.

We optimized hyperparameters for DeepFitness with grid search. For on+off fitness models, we first optimized the off-target model, and then the on-target model using the frozen off-target model previously optimized. The following grid of hyperparameters has been considered:

- Learning rate: $[10^{-4}, 10^{-3}, \ldots, 10^{-2}]$.
- Weight decay: $[0, 10^{-4}, 10^{-3.5}, 10^{-3}, \ldots, 10^{-1}]$.
- Batch size: $[64, 128, 256]$.

The weights from each training run were taken using early stopping on the validation loss. Ten percent of training data were randomly selected as validation data for DeepFitness model selection.

To learn the unsupervised genotype space, we trained a VAE using the same encoder and featurization described for DeepFitness. A fully connected decoder reconstructs both the peptide sequence and the embedding of each amino acid, thus learning amino acid similarity on top of sequence similarity. The final loss is therefore the sum of the KL-divergence, the sequence reconstruction, and the amino acid reconstruction. Final representations are obtained from the latent space, and shown after 2D projection with UMAP.

D DeepFitness Batch Sampling

Eq. 3 only enables predictions between two consecutive rounds, so each minibatch must only contain two consecutive rounds with the same subset of genotypes. To ensure the loss is defined for each batch, we use a simple batch sampling algorithm shown in Listing D.

```
def sample_batch(
    batch_size: int,
    genotypes: np.ndarray,  # G x genotype embedding dimension
    count_matrix: np.ndarray,  # G x T
) -> tuple(np.ndarray, np.ndarray):  # X, y
    T = count_matrix.shape[1]
    t = np.random.randint(0, T - 1)
    log_enrichment_defined_mask = (
        (count_matrix[:, t] > 0) & (count_matrix[:, t + 1] > 0)
    )
    log_enrichment_defined_idx = np.where(
        log_enrichment_defined_mask,
    )[0]
    batch_idx = np.random.choice(
        log_enrichment_defined_idx,
        batch_size,
    )
    X = genotypes[batch_idx]
    y = count_matrix[batch_idx, [t, t + 1]]
    return X, y
```

E Data Description and Preparation

The directed evolution experiment analyzed in the following includes seven rounds (the first round is the random library), and three off-target rounds following rounds five, six and seven. The directed evolution experiment selects for activity for target the target, and the library includes 8-mers (cyclic peptides). We split the data by genotype into a training and a test set, and heldout out round seven and off-target seven to use as holdout rounds. We then filtered the training and test sets so that each genotype had a count of at least 10 in one on-target round and appeared in at least two consecutive

rounds. The final number of training/test genotypes is summarized in Table 2. The distribution of the enrichment for each pair of rounds is shown in Fig. 2.

Table 2: Genotype counts by time point and cohort.

Cohort	M_1	M_2	M_3	M_4	M_5	M_6	M_7
train	2180	14765	17419	18233	18476	16844	-
test	233	1598	1926	2025	2054	1878	1673
train (off-target)	-	-	-	-	15285	11834	-
test (off-target)	-	-	-	-	1675	1300	375

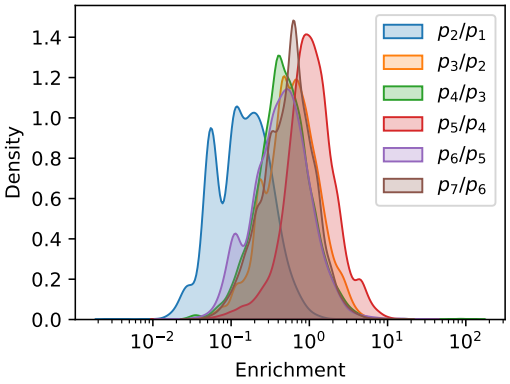



Figure 2: Kernel density plot of log enrichment ratios of the filtered genotypes by pair of time points.