Engineering highly active and diverse nuclease enzymes by combining machine learning and ultra-high-throughput screening

Neil Thomas + David Belanger
EvolutionaryScale + Google Deepmind

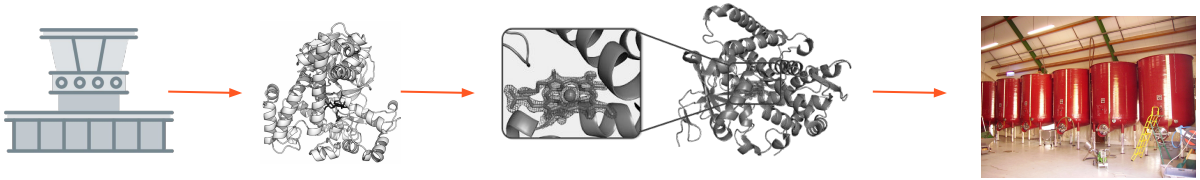
biorxiv.org/content/10.1101/2024.03.21.585615
github.com/google-deepmind/nuclease_design

Talk Roadmap

- Project Goals + Structure of Campaign
- Methods
  - ML Library Design
  - High Throughput Screening + Data Collection
- Results
  - Top Variants
  - Overall Library
  - Zero-shot
- Discussion

Project Goals + Structure

Stages of enzyme engineering



Target finding

Identify desired catalytic activity and usage requirements (e.g., pH and temperature).

Discovery

Identify a small number of natural or engineered sequences with non-zero activity.

Optimization

Find variants that improve activity, thermostability, solubility, expression, etc. over the backbones

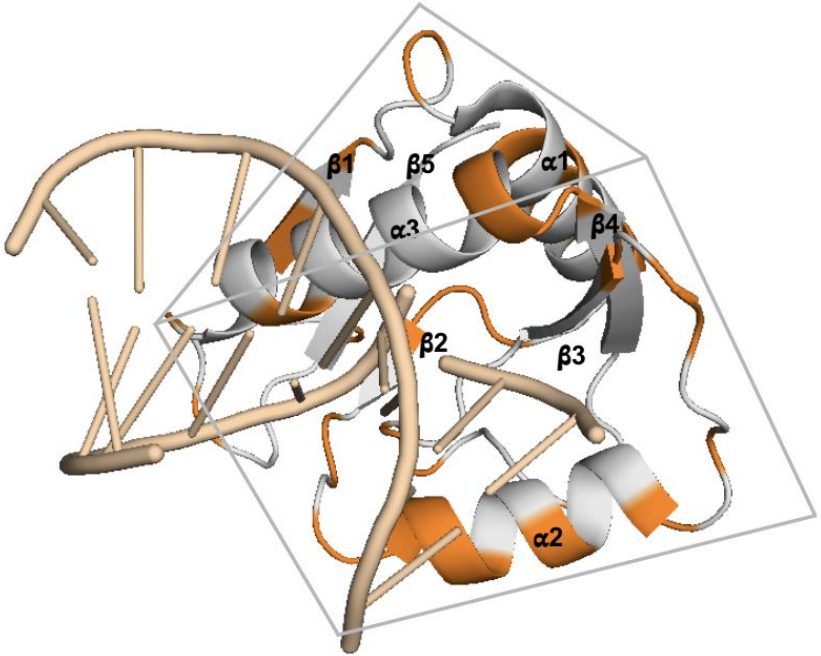
Scale up

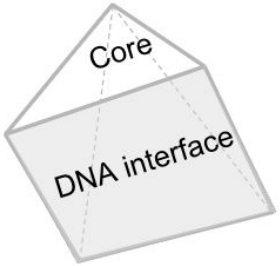
Engineer production conditions (typically fermentation) to produce the target enzyme at large scale.

Focus of this seminar series: using ML to improve both discovery and optimization

This talk: a deep dive about an optimization project

NucB - a nonspecific endonuclease





- hydrolyzes both single- and double-stranded DNA substrates (light orange)
- Isolated from Bacillus Licheniformis
- Optimal pH 9

Basle et al., 2018 "Crystal structure of NucB, a biofilm-degrading endonuclease"

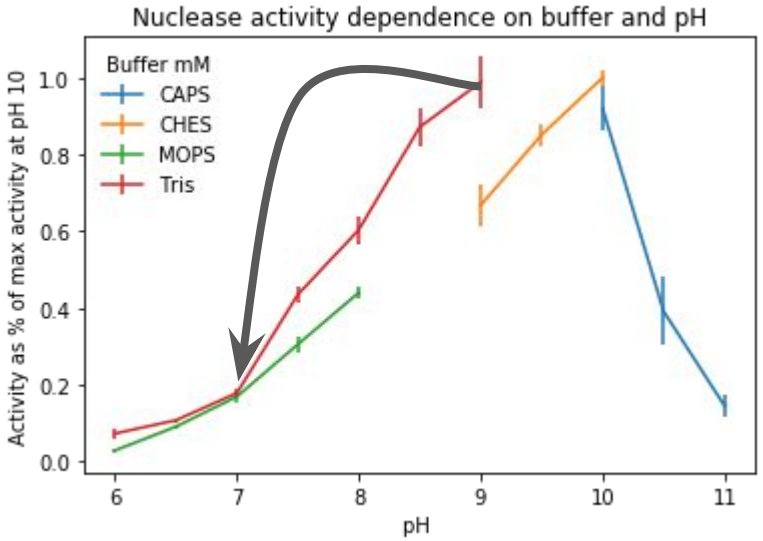
Goal of the optimization campaign: restore and improve NucB activity to unlock uses as a therapeutic

Target clinical application

Degrade biofilms that accumulate on chronic wounds

Challenge

- 80% reduced activity at pH 7 (therapeutic pH)



Therapeutic pH ← Wildtype pH

80% reduction in enzyme activity

Protein optimization goal

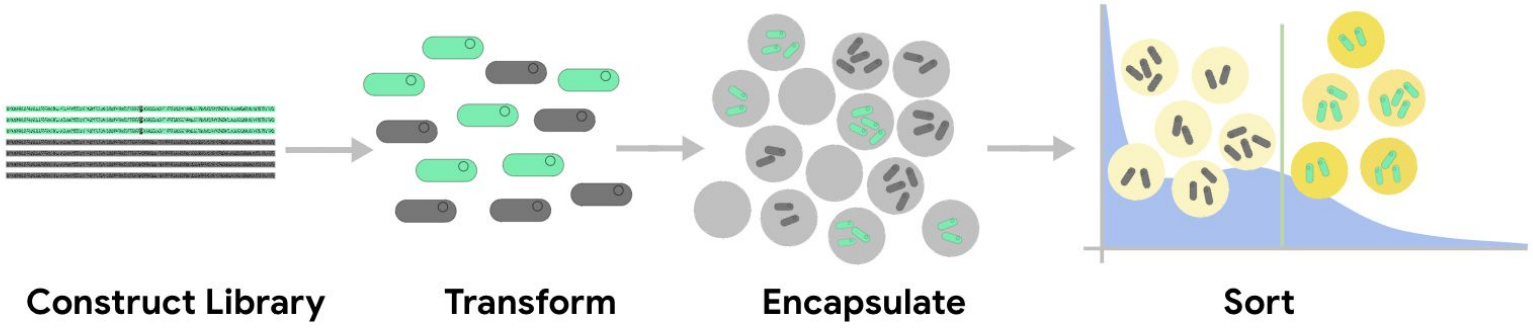
Improve the catalytic activity of NucB at pH 7.

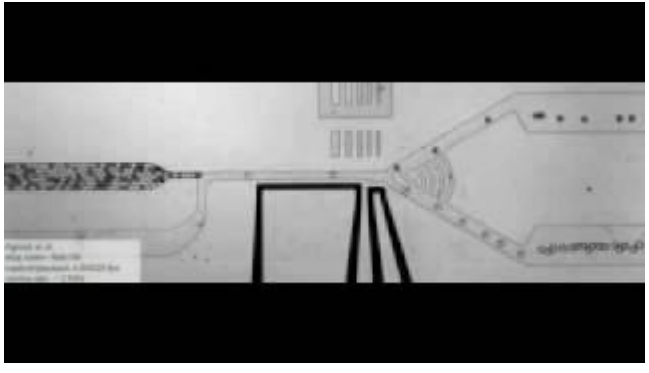
Methods research goal

Demonstrate that ML-guided protein design can improve over directed evolution when both use extremely high throughput experiments.

Experimental Platform - Ultra-high-throughput screening

Cell Sorting



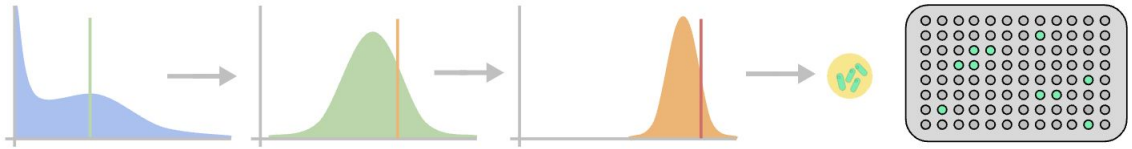


Thousands of droplets per second!

Triplebar

The two ways that we used cell sorting

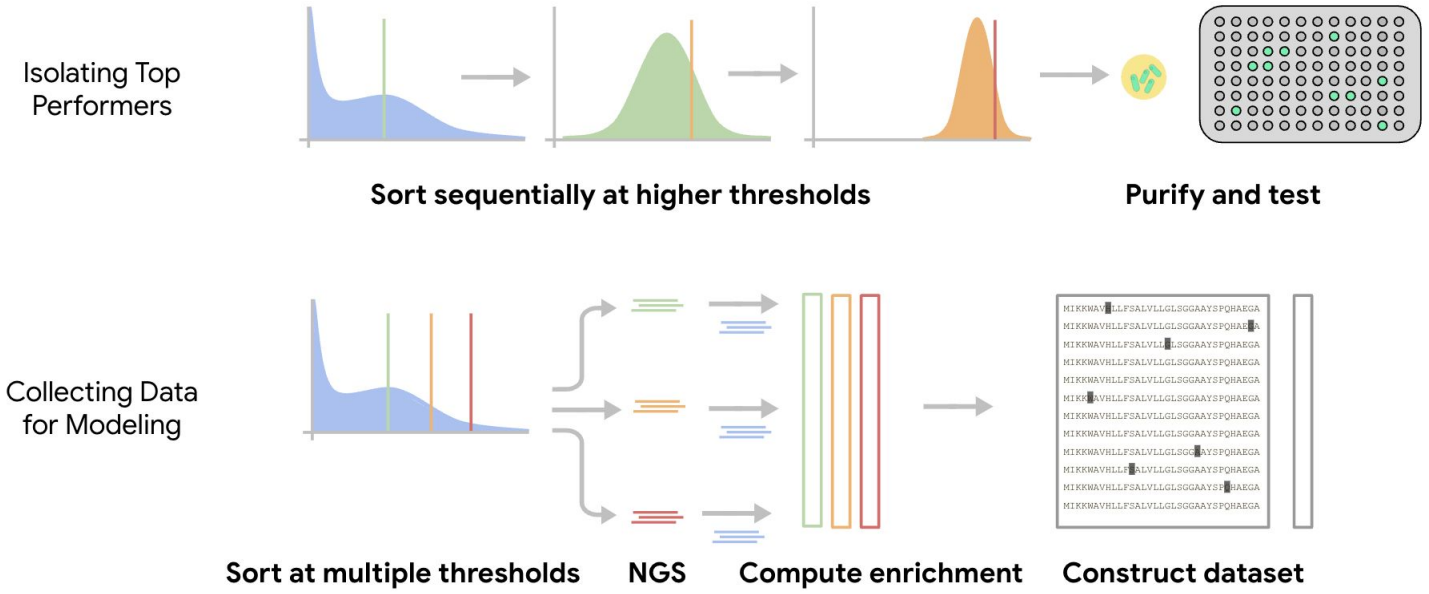
Isolating Top Performers



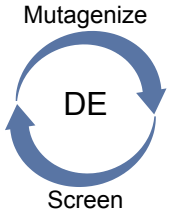
Sort sequentially at higher thresholds

Purify and test

The two ways that we used cell sorting

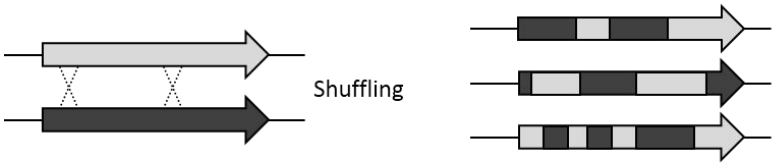


Baseline directed evolution techniques



Directed Evolution - DE

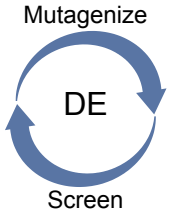
- Fully *in-vitro*
- Independent campaign
- Mutagenesis followed by screening
- Mutagenesis:
  - Error-prone PCR
  - Recombination (shuffling)



Hit Recombination - HR

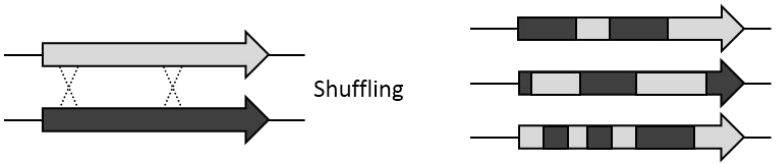
- Designed *in-silico*
- Model-free
- Screened in parallel with our designed libraries
- If A and B are both good, design A+B for the subsequent round

Baseline directed evolution techniques

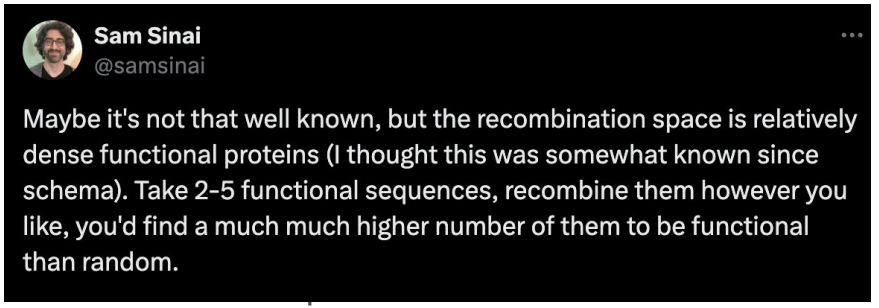


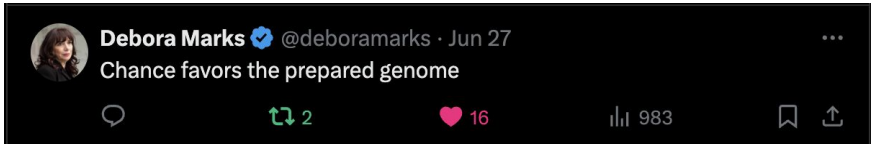
Directed Evolution - DE

- Fully *in-vitro*
- Independent campaign
- Mutagenesis followed by screening
- Mutagenesis:
  - Error-prone PCR
  - Recombination (shuffling)



Hit Recombination - HR

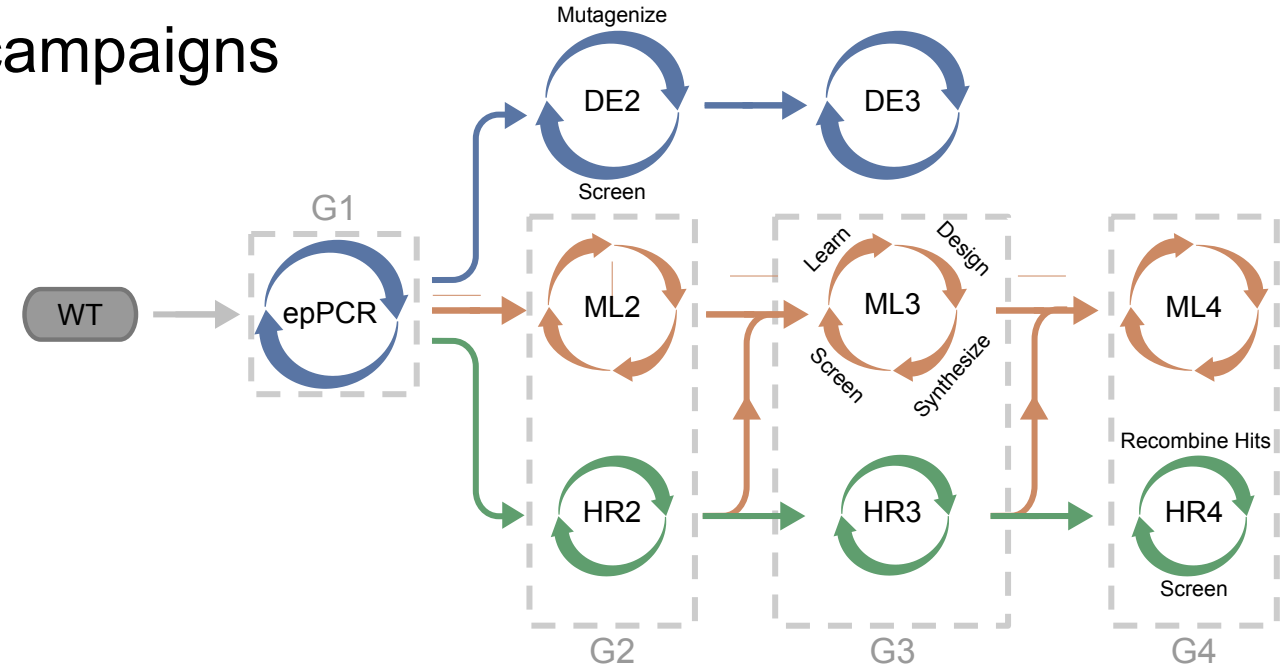




These are very successful techniques!

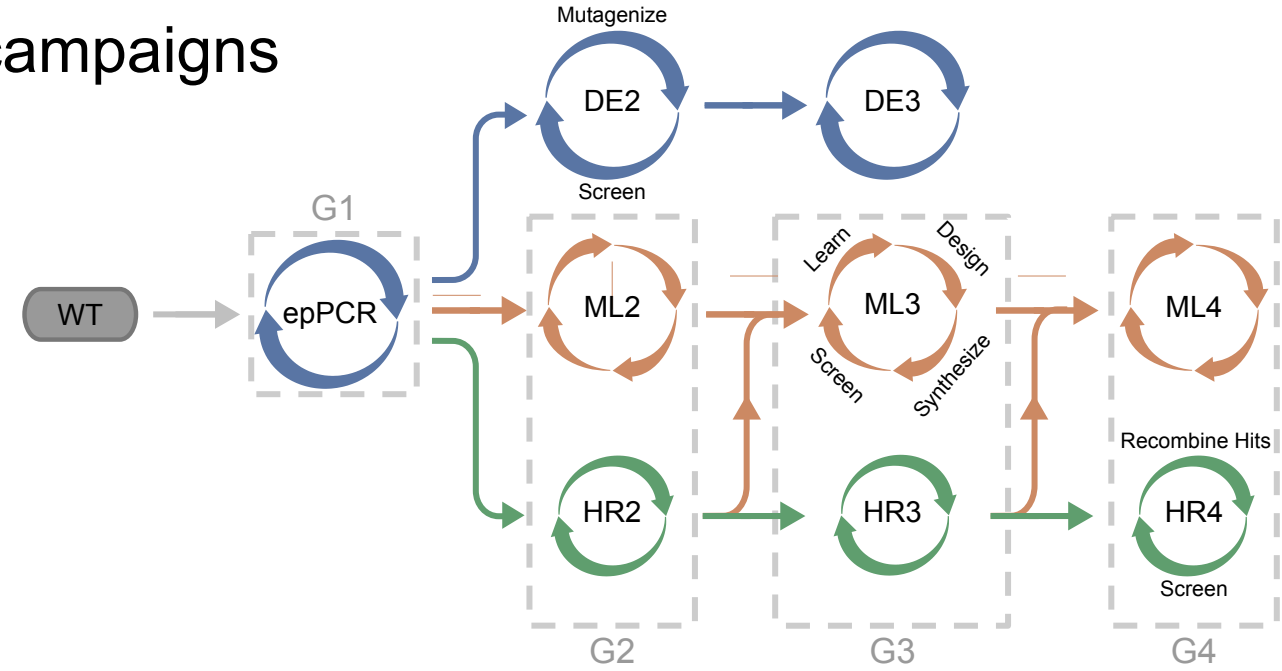
The optimization campaigns

- Starting point = wildtype (WT)
- 4 Rounds
- Initial G1 library generated by error-prone PCR
- DE run independently
- HR and ML screened in parallel



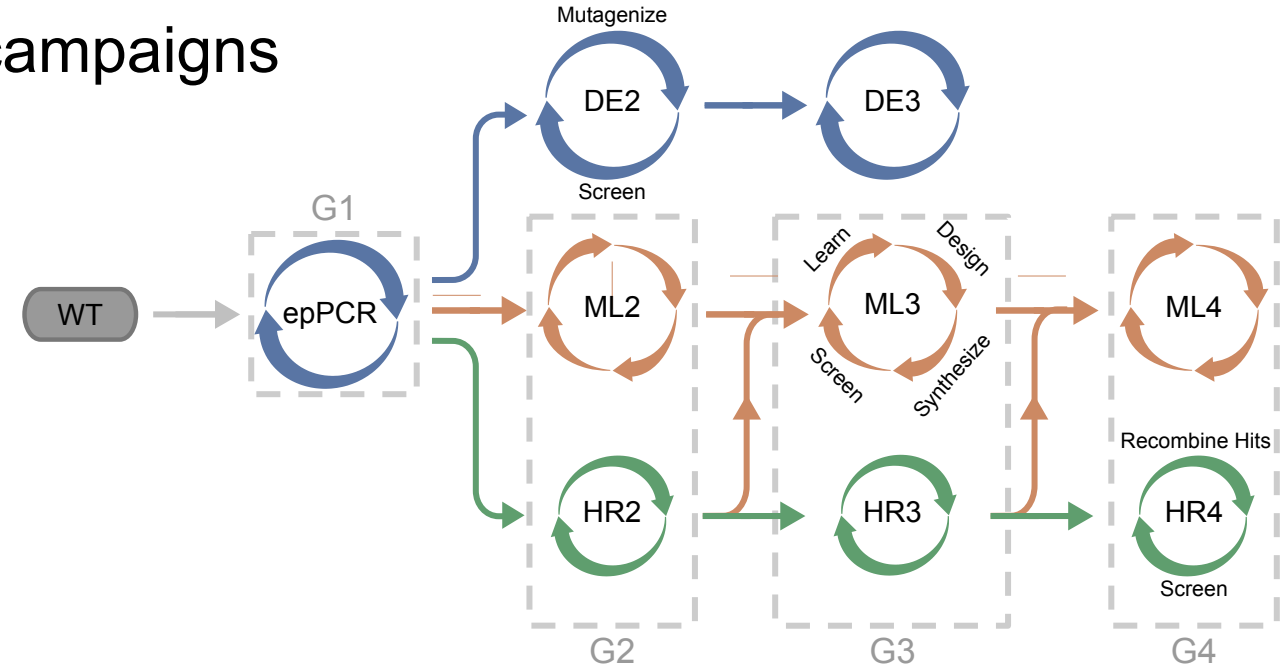
The optimization campaigns

- 4 Rounds



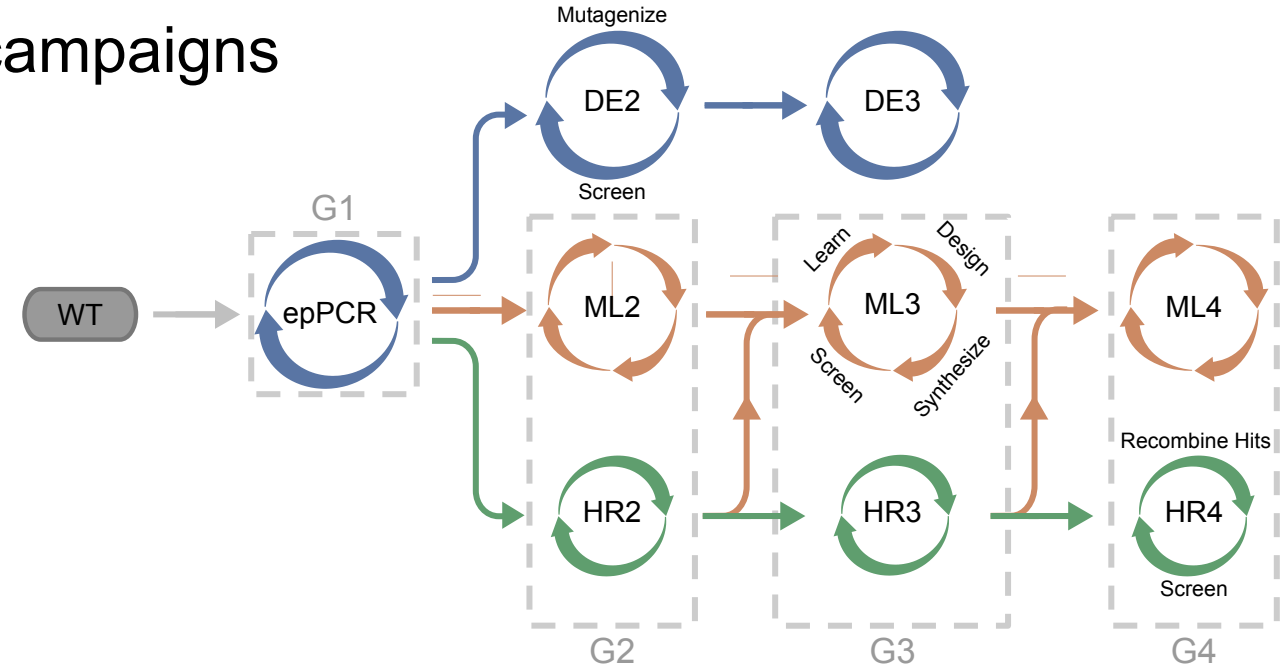
The optimization campaigns

- 4 Rounds
- Starting point = wildtype (WT)
- Initial G1 library generated by error-prone PCR

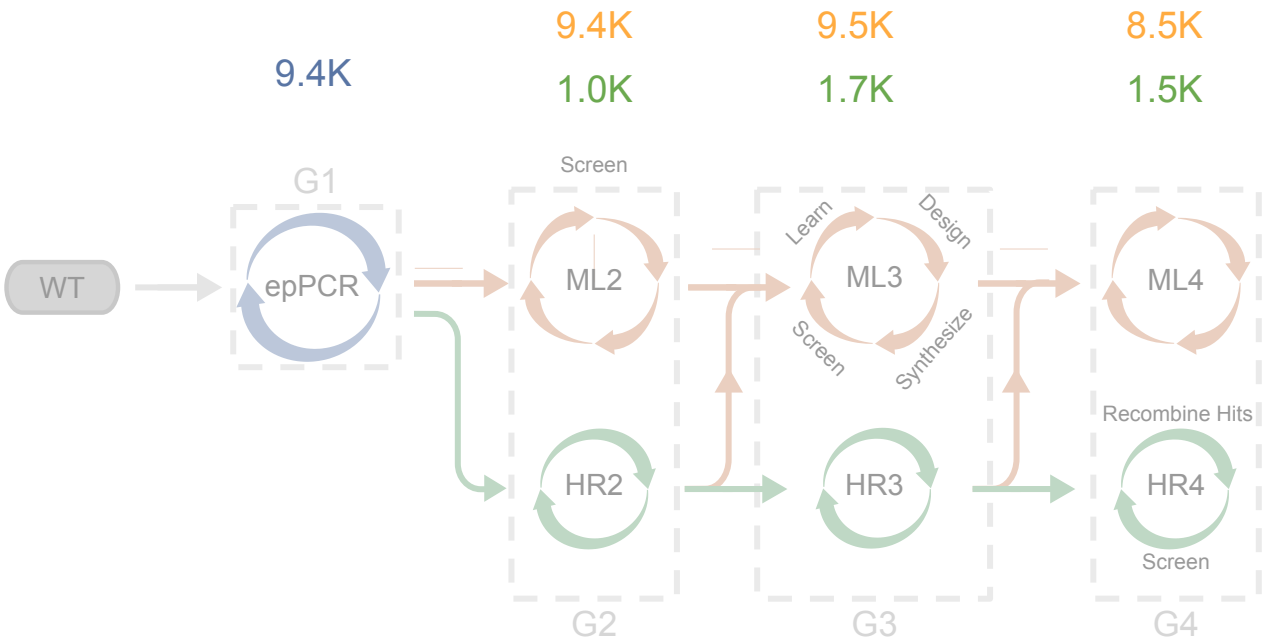


The optimization campaigns

- 4 Rounds
- Starting point = wildtype (WT)
- Initial G1 library generated by error-prone PCR
- DE run independently
- HR and ML screened in parallel



Campaign sizes
~10K per round

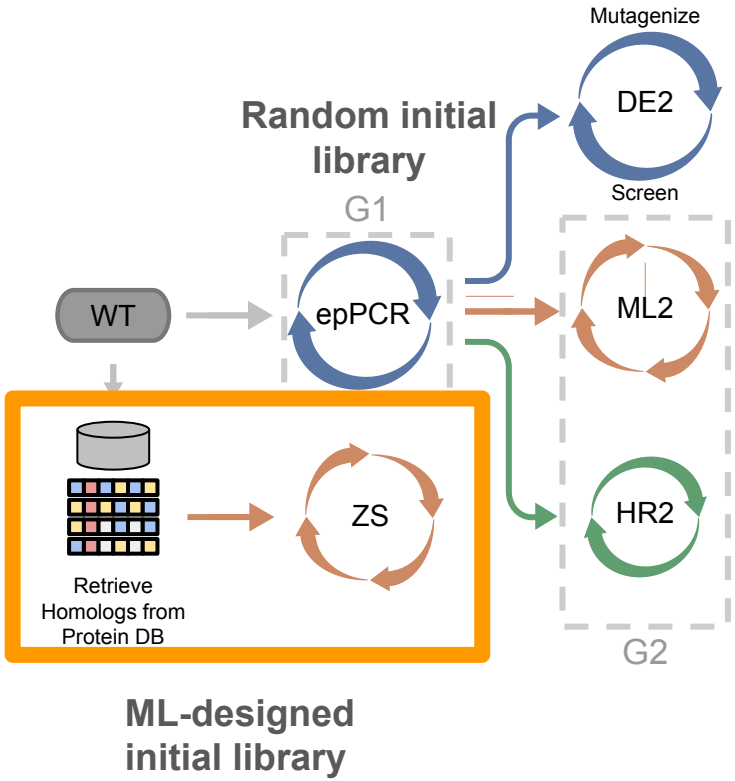


Zero-shot design:
Could we have obtained a better initial Library than error-prone PCR?

What we did

Generate a library using no experimental data for model training.

Compare the library to epPCR.



Methods

ML Library Design Methods

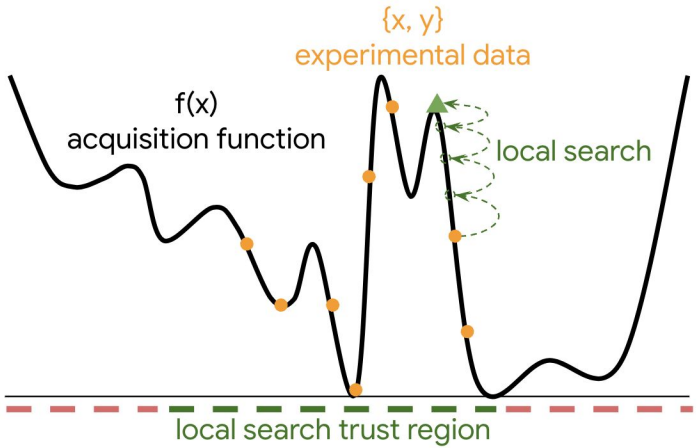
TeleProt: our library design framework

Search space:
consider substitutions (no indels) to the WT

Acquisition function:
use a model f(seq) to predict enzyme activity

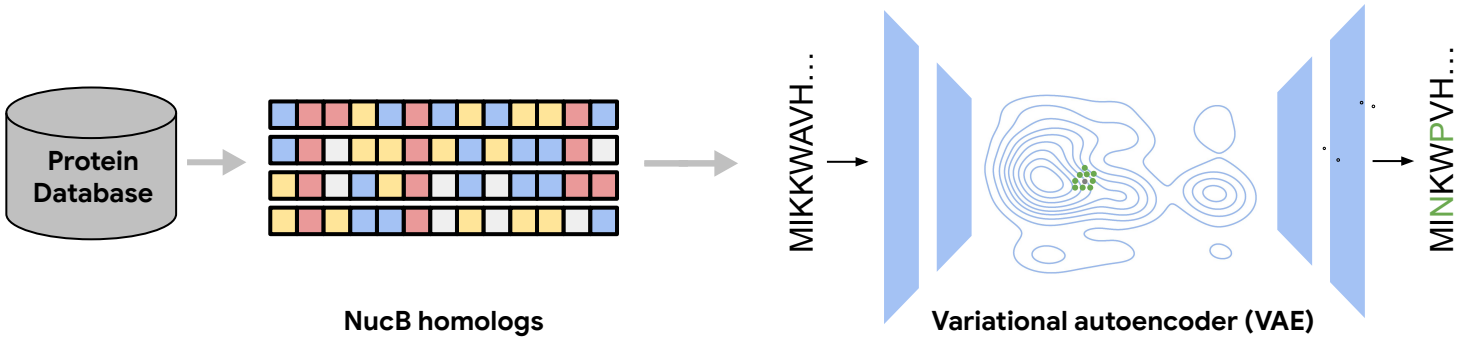
Candidate generation:
find new sequences with high f(seq)

Batch selection:
select a diverse subset of candidates



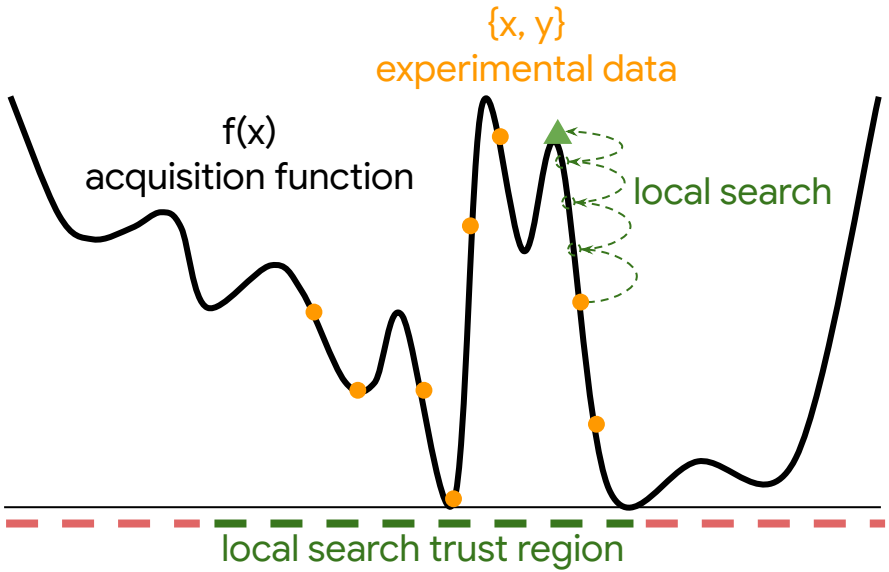
Supervised Model Fitting

Unsupervised Model Fitting



Similar model architecture as Riesselman et al., 2018

Candidate Generation #1: Local Search



Goal:

Find variants with high acquisition function score that are in regions close to the training data.

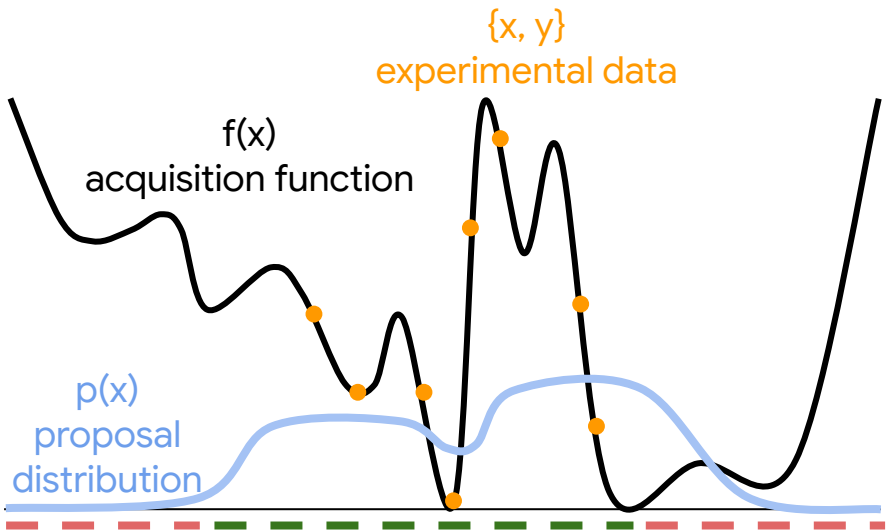
Techniques used:

Initialize the search at the WT and at hits from prior rounds.

Evolve a population of sequences towards those with high score.

Use an ensemble of different non-model-based methods for mutating high-scoring sequences.

Candidate Generation #2: Proposal Distribution



Goal:

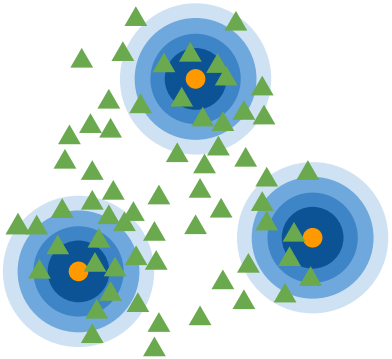
Sample variants that are likely to be functional and also in regions where the acquisition function is reliable.

Techniques used:

- VAE: Sample from a VAE trained on a combination of homologs and hits from prior rounds.
- ProSAR: Estimate the effect of each mutation using an additive model. Sample combinations of the top-scoring mutations (Fox et al. 2007).

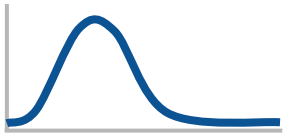
Batch Selection

i



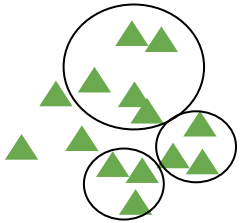
Assign each candidate (green) an 'extrapolation score': min distance from a hit in the training data (orange).

ii



Specify a target distribution over extrapolation scores

iii



Select a subset of the candidates that satisfy the extrapolation score distribution and also do not over-use individual mutations.

Why is this necessary?

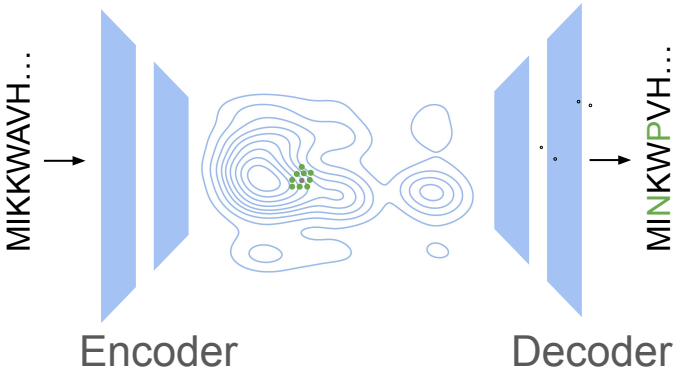
Simply selecting the top-scoring sequences leads to a low diversity library and doesn't provide a controllable explore-exploit tradeoff.

Sampling variants from a VAE

VAE (Kingma et al., 2014)

Generative model: z ~ Normal, x ~ Decoder(z)

Inference: z ~ Encoder(x)



Sampling WT neighbors (Giessel et al., 2022)

x ~ Decoder(Encoder(WT))

Reject any x with too many mutations or gaps.

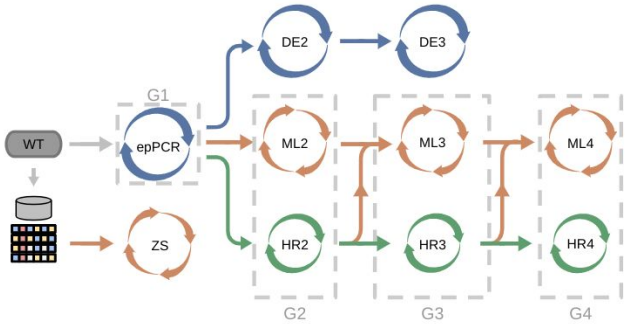
TeleProt Systems

Method Name	Acquisition Function	Candidate Generation	Round
Zero-shot	None	Neighbor sampling with VAE[100]	ZS
MBO-DNN	CNN classifier	Randomized local search	ML2, ML3, ML4
Prosar+Screen	VAE likelihood	Combinatorial library from ProSAR[61]	ML2, ML3
Sample+Screen	CNN classifier	Neighbor sampling with semi-supervised VAE	ML4

Data source	Evolution	Assay	Both

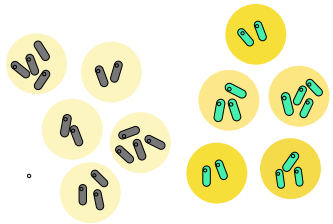
Key idea

As data accumulated, we transitioned from depending on evolutionary data to assay-labeled data.

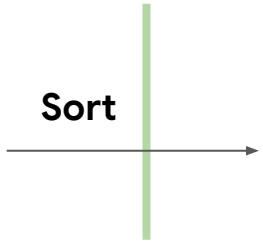


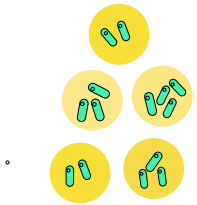
Data Collection

Key idea: Enrichment factors



A: 5 | 0.5
B: 5 | 0.5



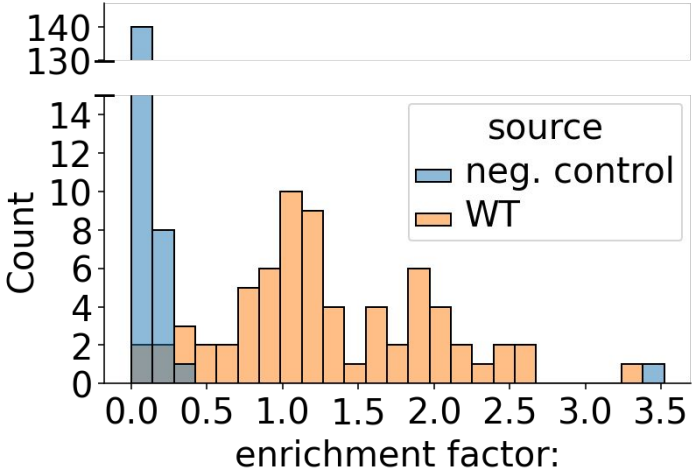


A: 0 | 0.0
B: 5 | 1.0

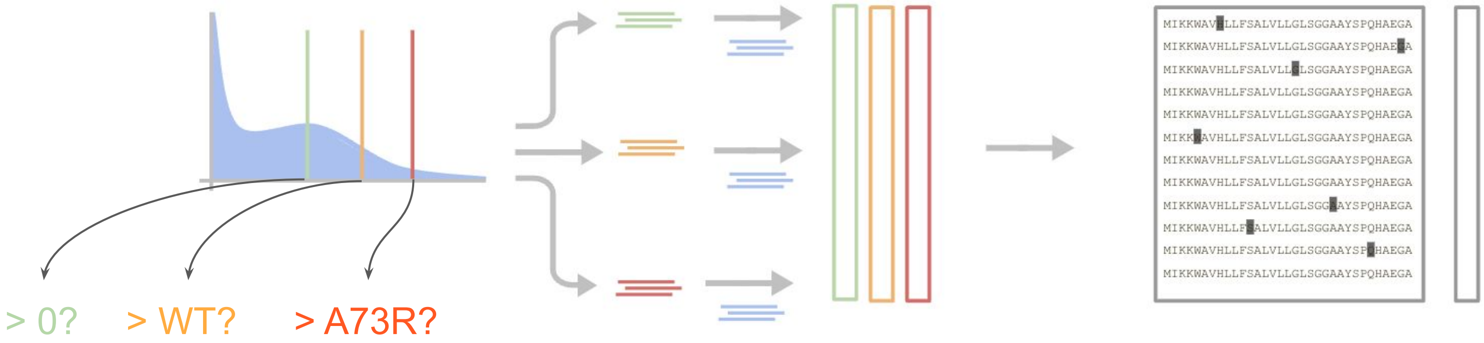
Enrichment Factor:
A: 0
B: 2

Key idea: Use **fiducial** sequences to calibrate hit-calling

- Fiducial has known activity
- Multiple replicates of a fiducial using **synonymous codons** to serve as a null distribution
- For a new variant EF: assign p-value with **right-sided t-test** compared to fiducial
- Call a “hit” if p-value is significant after **FDR correction**



Sorting at multiple thresholds gives data with intermediate activity resolution



Sort at multiple thresholds

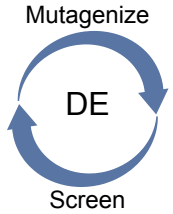
Compute enrichment

Resolve labels and construct dataset

Results

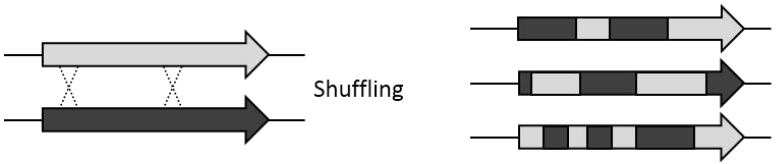
Reminder: Campaign

Reminder: Baseline DE techniques



Directed Evolution - DE

- Fully *in-vitro*
- Independent campaign
- Mutagenesis followed by screening
- Mutagenesis:
  - Error-prone PCR
  - Recombination (shuffling)

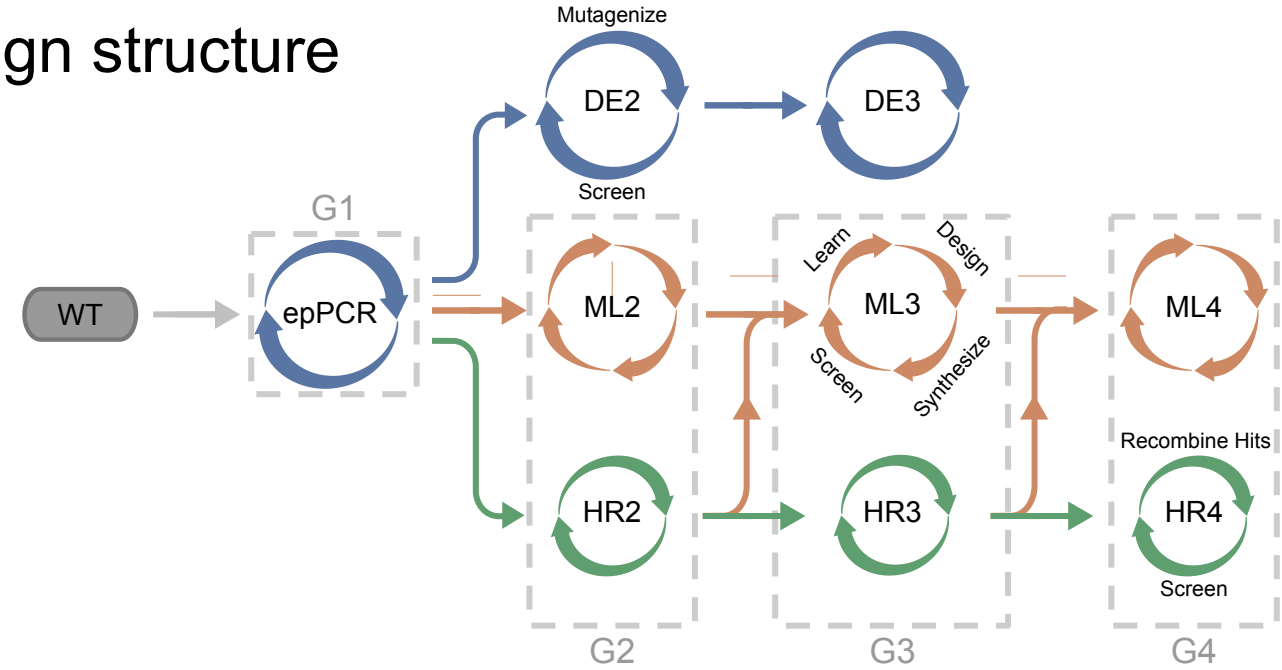


Hit Recombination - HR

- Designed *in-silico*
- Model-free
- Screened in parallel with our designed libraries
- If A and B are both good, design A+B for the subsequent round

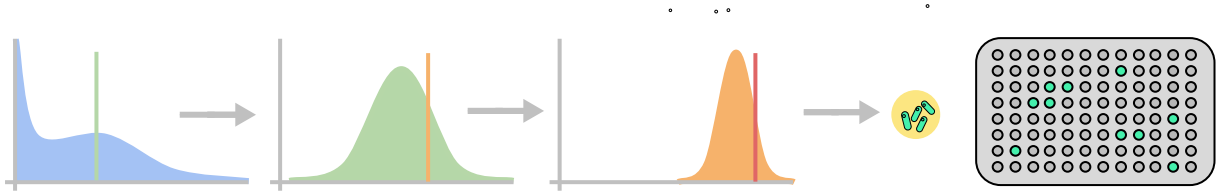
Reminder: campaign structure

- Starting point = wildtype (WT)
- 4 Rounds
- Initial G1 library generated by error-prone PCR
- DE run independently
- HR and ML screened in parallel



Activity of the top-performing variants

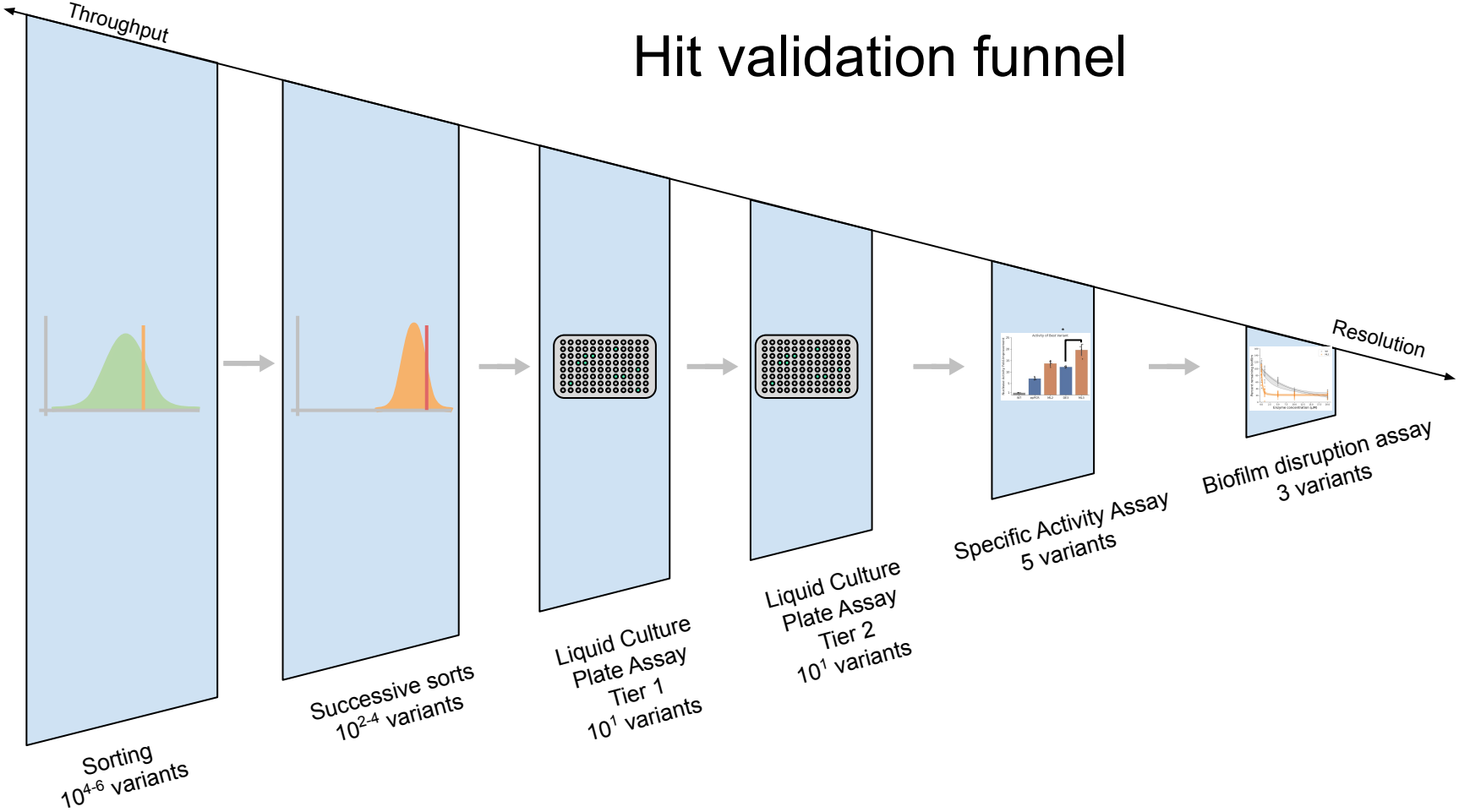
Isolating Top Performers



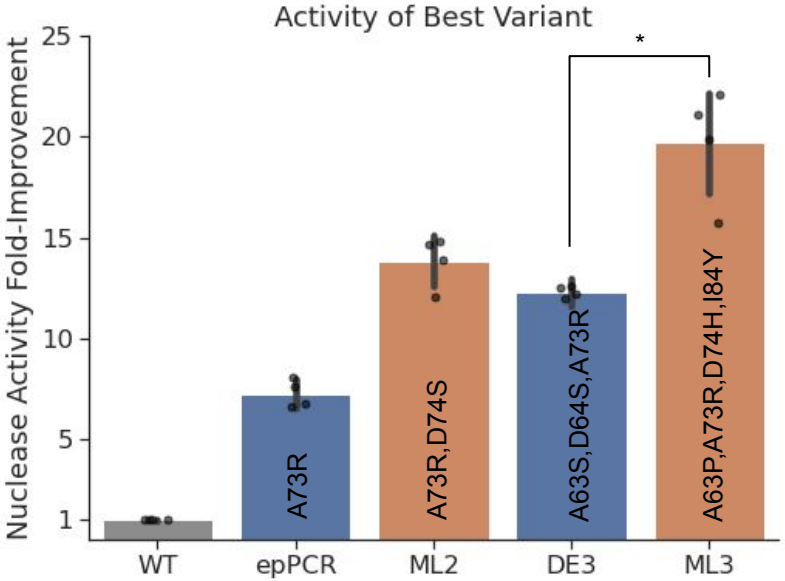
Sort sequentially at higher thresholds

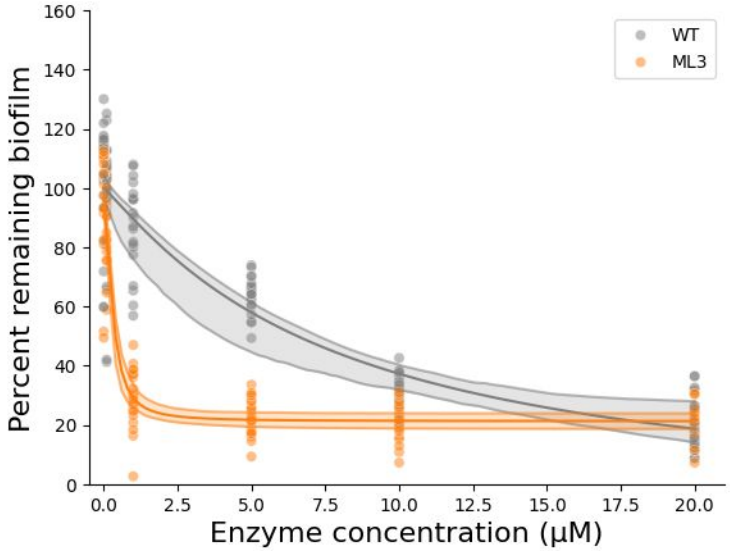
Purify and test

Hit validation funnel



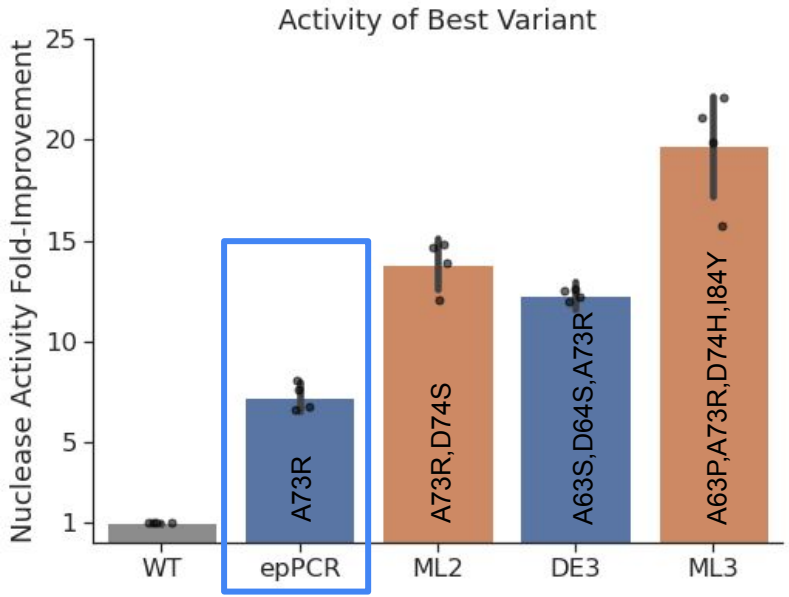
Top ML variant: 19x. Top DE variant: 12x.





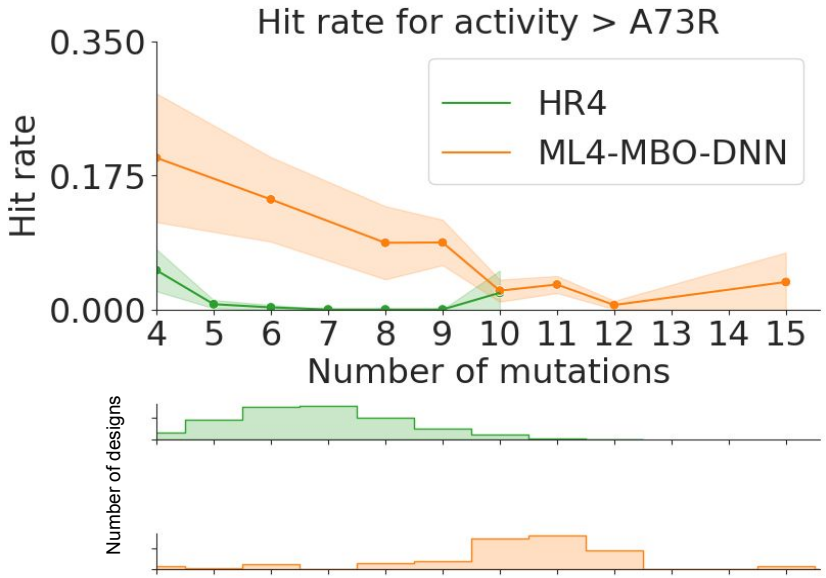
- Purified enzyme activity assessed at 4 concentrations
- Top hit validated for biofilm degradation

Note: A73R ~8x improvement



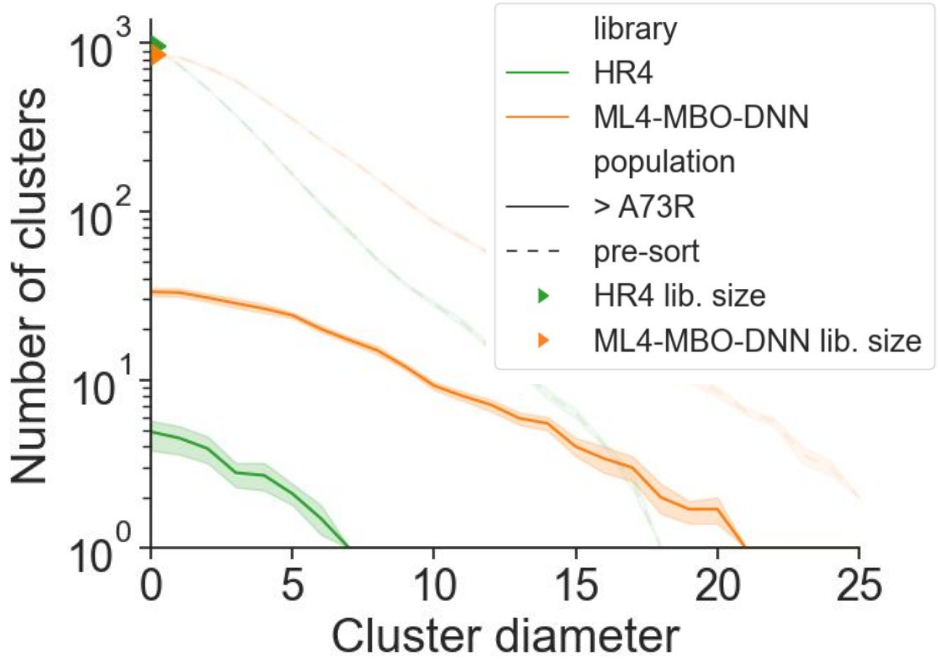
Assessing the Overall Composition of the Libraries

ML produced a much higher rate of hits than HR



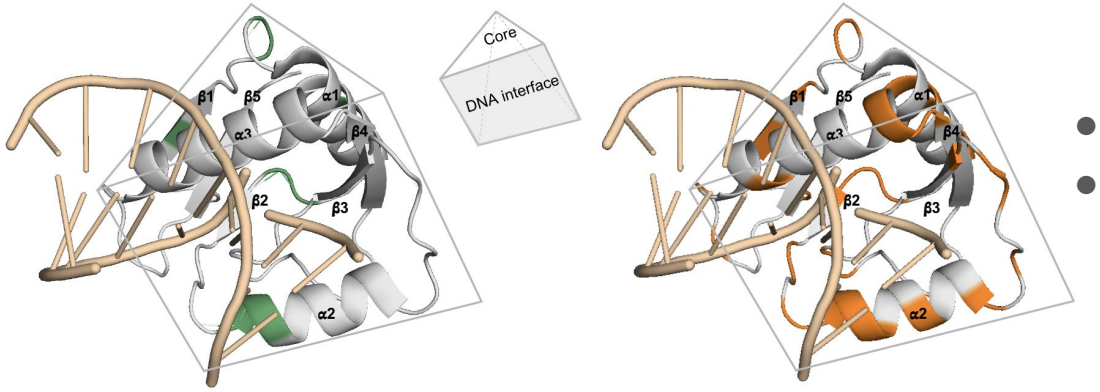
ML4 maintained high activity (>A73R) while designing out to 15 mutations

ML designs were substantially more diverse than HR designs



- Cluster diameter: maximum Hamming distance between sequences in the same cluster.
- Similar pre-sort library sizes

Designs exhibit Structural Diversity

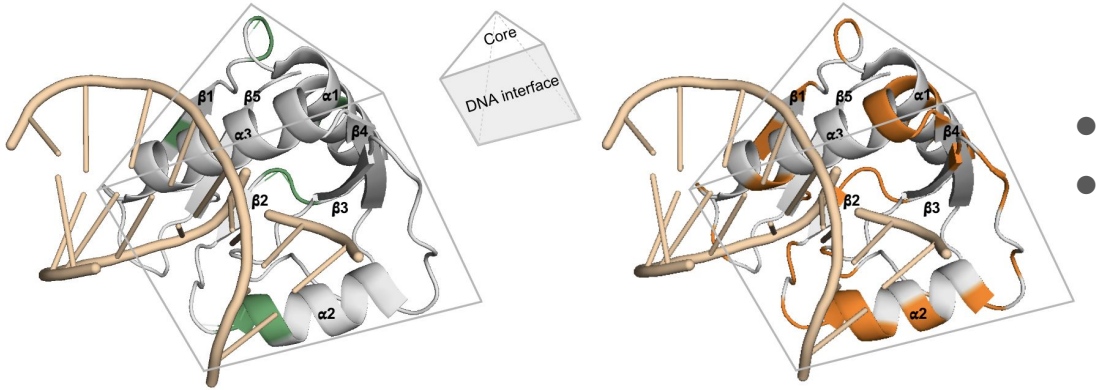


HR

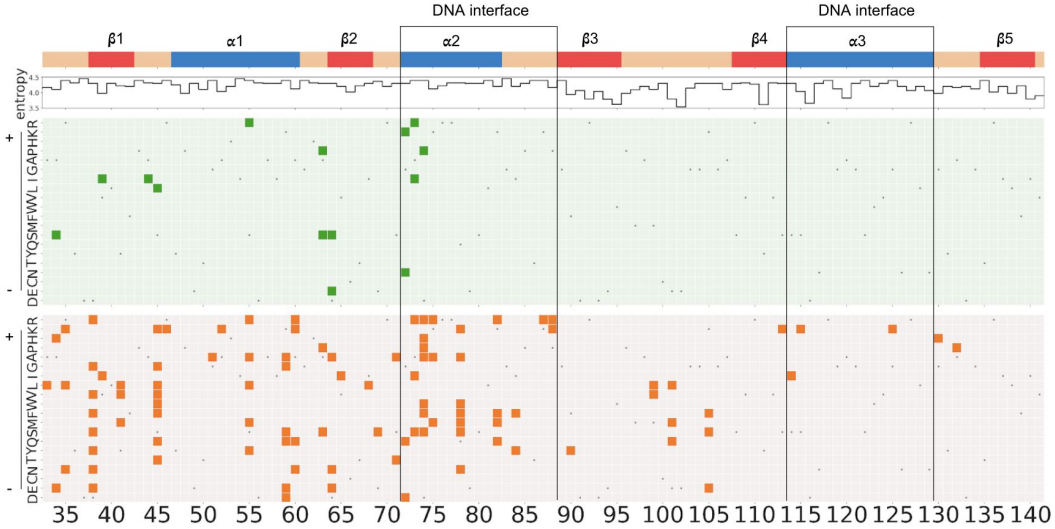
ML

- Active designs span many positions
- Span many functional domains

Designs exhibit Structural Diversity



- Active designs span many positions
- Span many functional domains

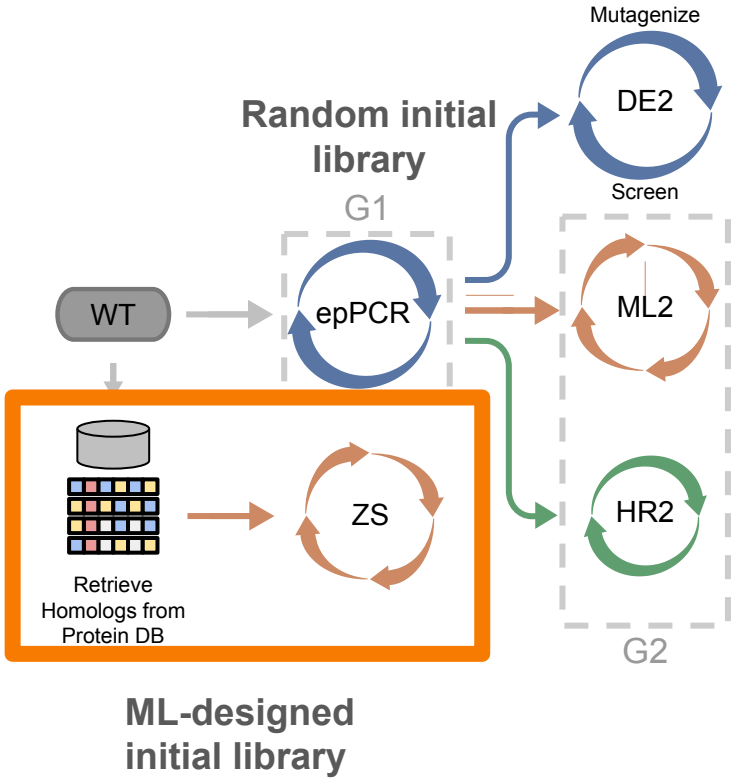


HR

ML

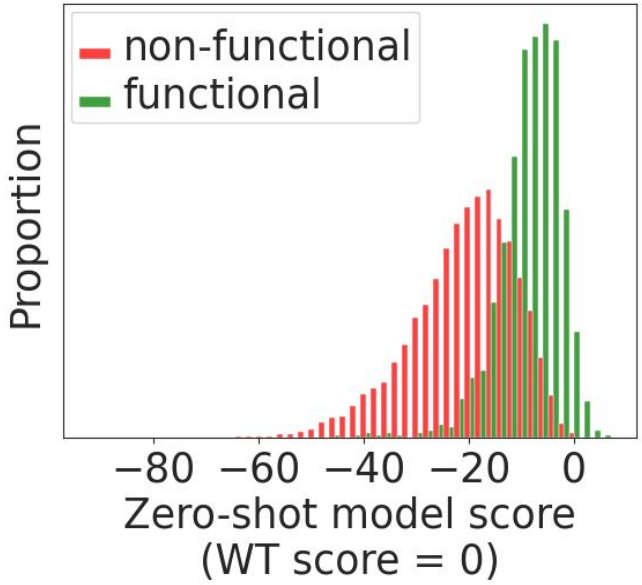
Zero-Shot Initial Library Design

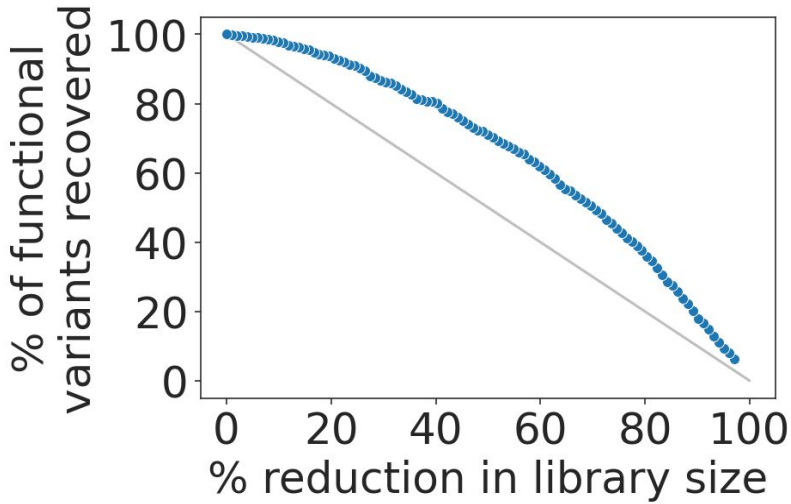
Reminder: zero-shot design



Why did we pursue this investigation?

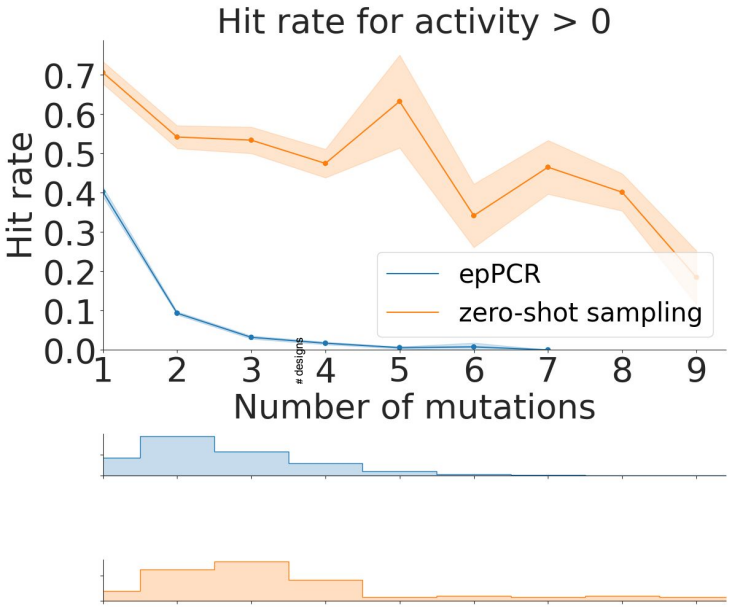
Retrospective analysis on the G1 data showed that a zero-shot model could be used to enrich for functional variants.



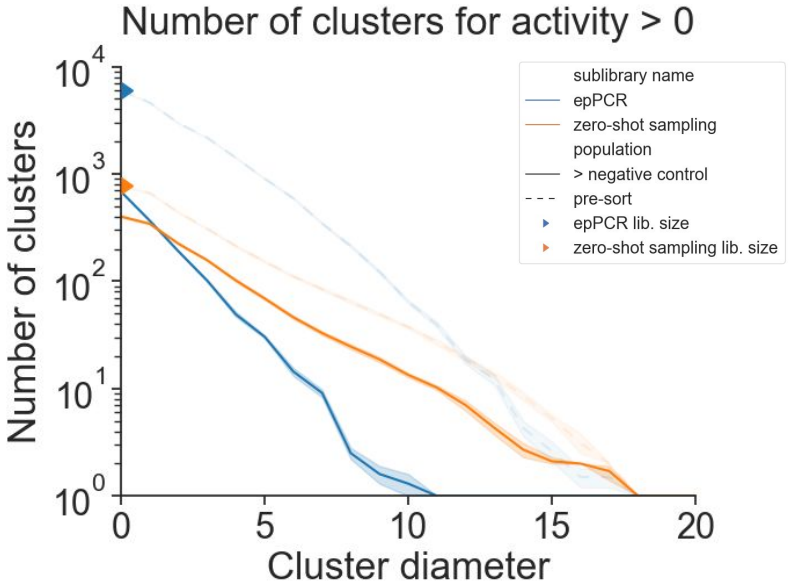


We could have reduced the library by 50% while keeping 75% of the functional variants

Finding enzyme variants with non-zero activity

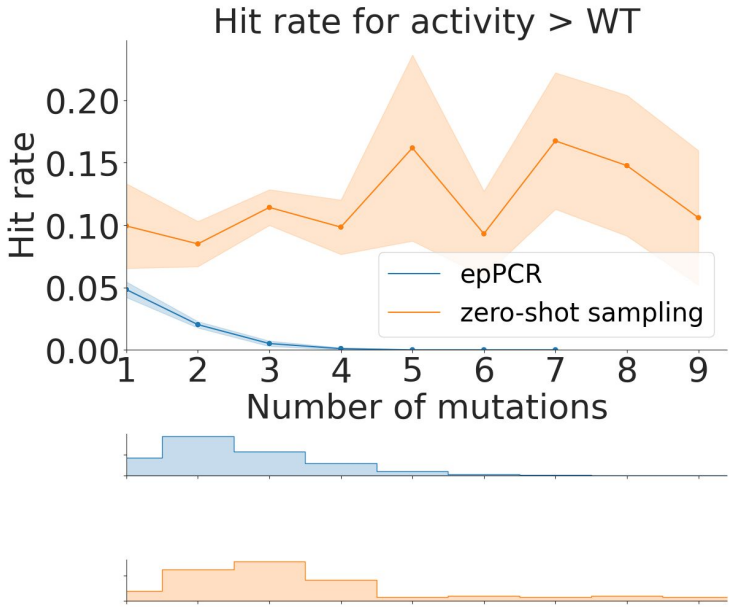


zero-shot design has a better hit rate

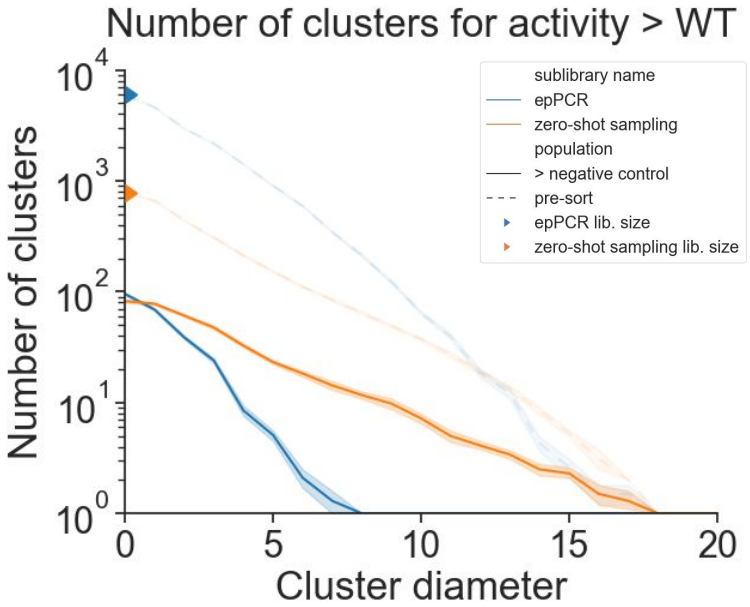


zero-shot hits are more diverse

Finding enzyme variants that are better than the WT



zero-shot design has a better hit rate

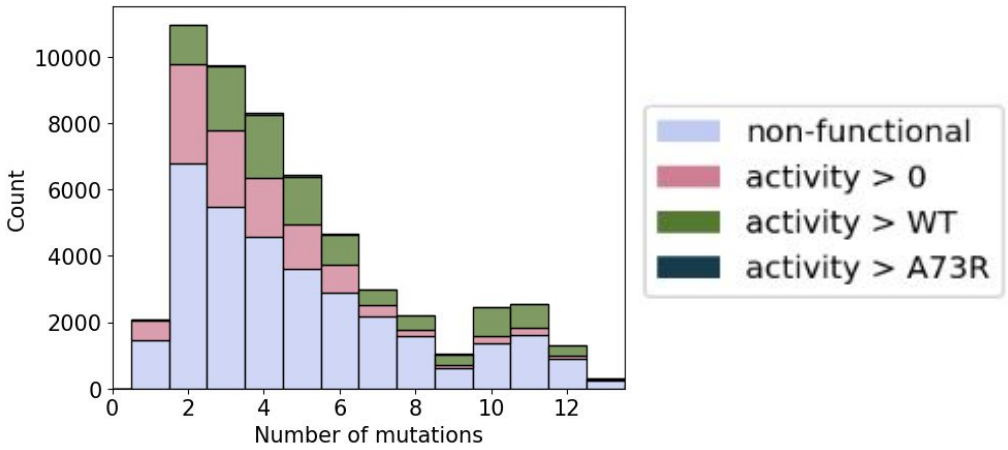


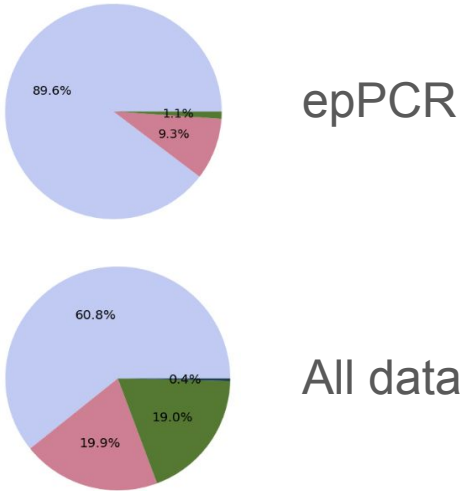
zero-shot hits are more diverse

Our Enzyme Activity Dataset

github.com/google-deepmind/nuclease_design

Our open-source enzyme fitness landscape - 56K variants!





- Active variants out to >13 mutations
- Four discrete activity levels

- Many more active variants than epPCR alone

github.com/google-deepmind/nuclease_design

Discussion

Future work

- Improving modeling with, e.g., representations from protein language models
- Leveraging structure-conditioned models for zero shot design
- Avoiding bottlenecks of DNA synthesis costs using randomized DNA synthesis protocols
- Incorporating experimental uncertainty from sequencing data

Summary of our findings

- MLDE outperformed DE when compared head-to-head
- TeleProt is a flexible framework for balancing evolutionary and assay-labeled data when designing libraries.
- MSAs are powerful for zero shot design. We didn’t use structure or large-scale pretraining!
- Using high-throughput experiments enabled us to employ a large, diverse portfolio of sequence design approaches

Acknowledgements

Google

Maria Chavarha

Lucy Colwell

Charlie Emrich

Jun Kim

Abi Ramanan

Triplebar

Jeremy Agresti

Lucas Frenz

Kathleen Hirano

Kevin Hoff

Kosuke Iwai

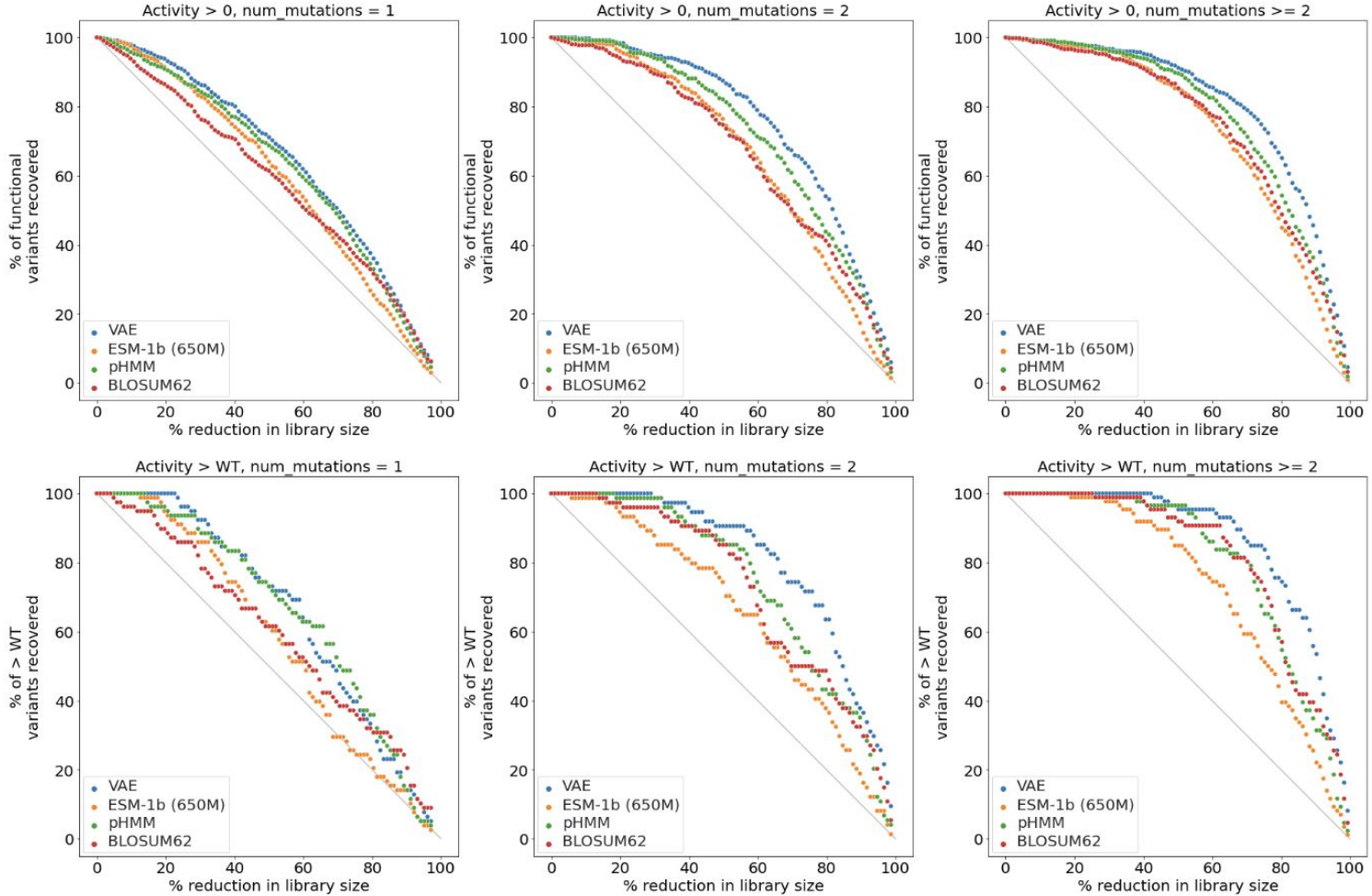
Hanson Lee

Kendra Nyberg

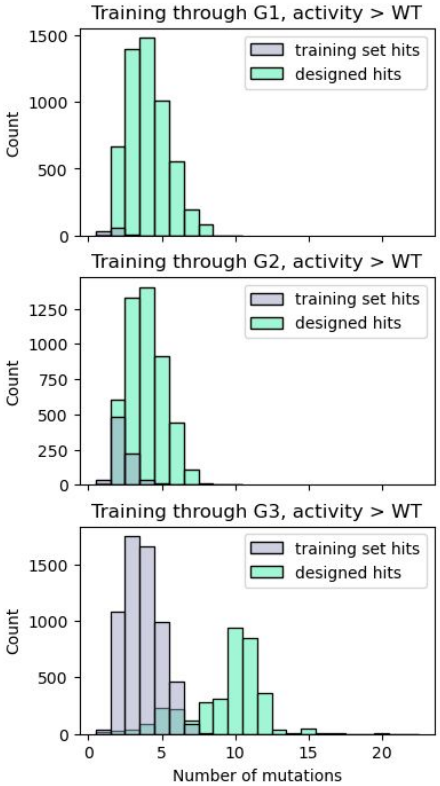
Vanja Polic

Chenling Xu

Additional Info

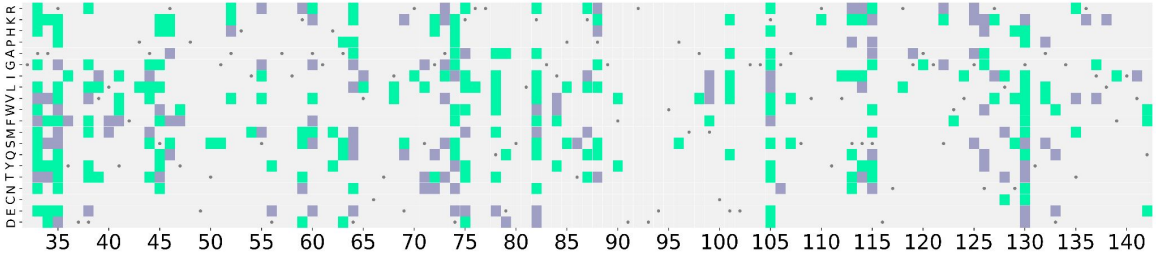

Activity > 0, num_mutations = 1
Activity > 0, num_mutations = 2
Activity > 0, num_mutations >= 2
Activity > WT, num_mutations = 1
Activity > WT, num_mutations = 2
Activity > WT, num_mutations >= 2
% of functional variants recovered
% of > WT variants recovered
% reduction in library size
VAE
ESM-1b (650M)
pHMM
BLOSUM62

ML methods extrapolated beyond their training set

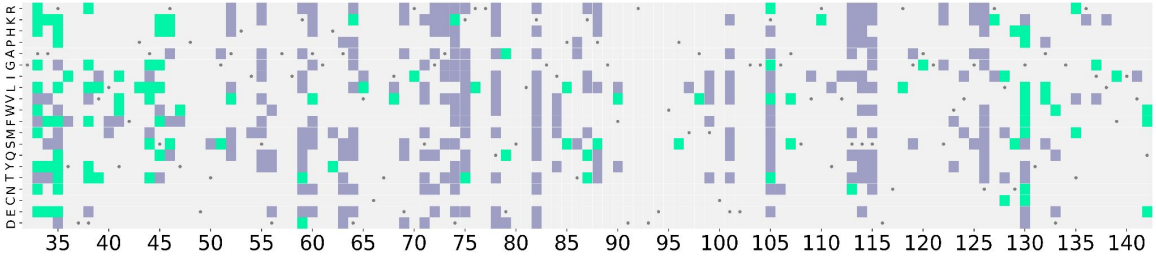


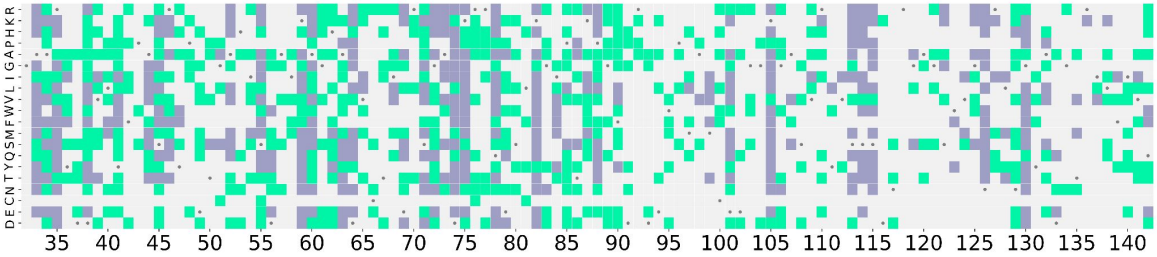
ML methods extrapolated beyond their training set

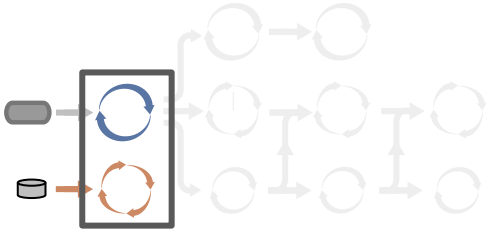
G1:

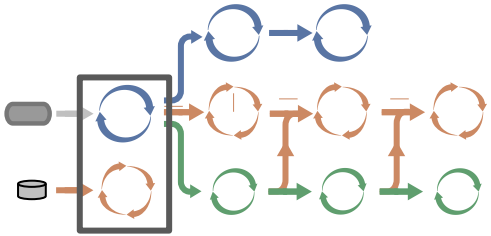


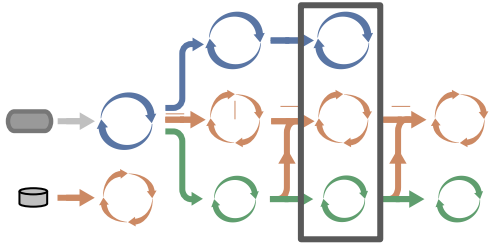
G2:

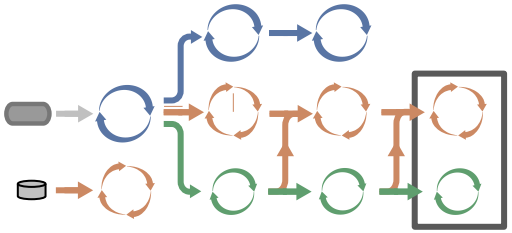




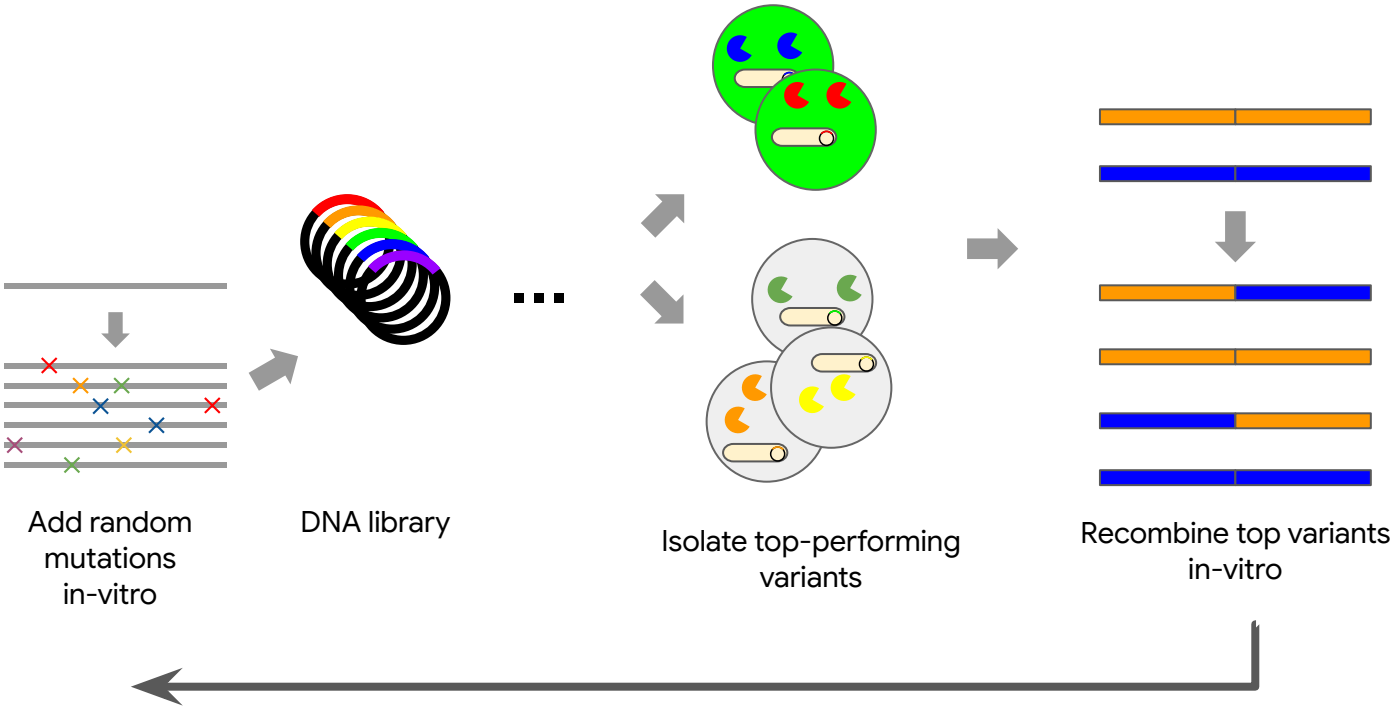




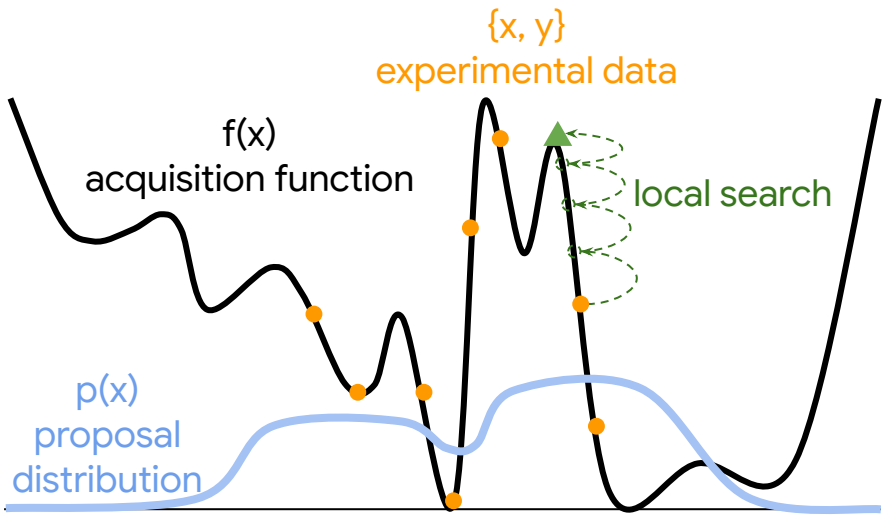




Traditional directed evolution



Candidate Generation #2: Proposal Distribution



Goal:

Sample variants that are likely to be functional and also in regions where the acquisition function is reliable.

Techniques used:

- Sample from a VAE trained on a combination of homologs and hits from prior rounds.
- Estimate the effect of each mutation using an additive model. Sample combinations of the top-scoring mutations (ProSAR; Fox et al. 2007).

Project goals + structure: Neil

ML methods: David

Data collection / processing: Neil

Results: Neil

Zero-Shot results: David

Dataset: David

Discussion: Neil