Computational Methods in Drug Discovery: From Molecular Modeling To Library Design

Zhang, Chris

UNIVERSITY OF CALIFORNIA,
IRVINE

Computational Methods in Drug Discovery: From Molecular Modeling To Library Design

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Chemistry

by

Chris Zhang

Dissertation Committee:
Professor David Mobley, Chair
Professor Kieron Burke
Professor Brian Paegel

2024

TABLE OF CONTENTS

LIST OF FIGURES

Page

LIST OF TABLES

Page

ACKNOWLEDGMENTS

First and most importantly, I'd like to thank my advisor David Mobley. David's patience and guidance these past five years is without a doubt what has allowed me to get to where I am today. I am incredibly grateful to him for allowing me to explore different career interests throughout my time in grad school and developing his students not only as scientists, but as people. It's been an absolute privilege to learn from David.

I would also like to thank and acknowledge the other members of my thesis committee. I'd like to thank Professor Brian Paegel; co-authoring a paper with Brian and members of his lab showed me how much science can also be a medium of expression and creativity. I also thank Professor Kieron Burke for teaching me in many of the classes I took during graduate school and always challenging me to improve.

I also thank the Mobley Lab and all the friends I have made at UCI over the years. In particular I thank Hannah Baumann, Pavan Behara, Swapnil Wagle and Oanh Tran for all those lunchtime conversations. I thank Mary Pitman, Meghan Osato, Anjali Dixit and Patrick Fitzgerald for being wonderful collaborators. I also thank lab alums Nathan Lim, Victoria Lim, Mary Pitman and David Wych for being great mentors.

I thank my family for all their support and all the friends-turned-family I've made along the way. To Hannah Baumann, Edward Chen, Emily Chen, Shane Flynn, Joyce Kang, Michael Lim, Davina Pham, Aoon Rizvi, Moises Romero, Patrick Tan, Cynthia Wong, Meagan Wong, Shana Yang – we did it!

Finally, I acknowledge financial support through grants from the National Institute of Health that has made it possible to conduct my research.

Chapter 4 of this dissertation is reprinted with permission from the American Chemical Society regarding the published work: Zhang, C.; Pitman, M.; Dixit, A.; Leelananda, S.; Palacci, H.; Lawler, M.; Belyanskaya, S.; Grady, L.; Franklin, J.; Tilmans, N.; Mobley, D. L. "Building-Block Based Binding Predictions for DNA-Encoded Libraries." *J. Chem. Inf. Model.*, **2023**, 63, 16, 5120-5132. I acknowledge and thank my co-authors: Anjali Dixit and David Mobley (UCI), Mary Pitman (SandboxAQ), Henri Palacci (DE Shaw Research), Sumudu Leelananda, Meghan Lawler, Svetlana Belyanskaya, LeShadric Grady, Joe Franklin and Nicolas Tilmans (Anagenex).

Parts of Chapters 3 and 5 are currently in review regarding the submitted works: Fitzgerald, P.; Dixit, A.; Zhang, C.; Mobley, D. L.; Paegel, B. "A Building Block Centric Approach to DNA-Encoded Library Design." and Zhang, C.; Osato, M.; Mobley, D. L.; "Characterizing Discrete Binding Conformations of T4 L99A via Markov State Modeling", respectively. I acknowledge and thank my UCI co-authors Anjali Dixit, Patrick Fitzgerald, Meghan Osato, David Mobley and Brian Paegel.

VITA

Chris Zhang

Education

Doctor of Philosophy in Chemistry **2024**
University of California, Irvine *Irvine, California*

Bachelor of Arts in Chemistry **2018**
Harvard University *Cambridge, Massachusetts*

Professional Experience

Graduate Student Researcher **Sep 2018 – Mar 2024**
University of California, Irvine *Irvine, California*

CompChem/ML Intern **Jun 2023 – Sept 2023**
Janssen Pharmaceuticals *San Diego, California*

Data Scientist Intern, Analytics **Jun 2022 – Sept 2022**
Meta (formerly Facebook) *Menlo Park, California*

Pharmacokinetics Intern **May 2018 – Aug 2018**
Merck Research Laboratories *Boston, Massachusetts*

Undergraduate Researcher Assistant **Jun 2016 – May 2018**
Harvard University *Cambridge, Massachusetts*

Teaching Experience

Graduate Teaching Assistant **2018 – 2023**
General Chemistry, University of California, Irvine *Irvine, California*

Undergraduate Course Assistant **2017 – 2018**
Linear Algebra and Physics, Harvard University *Cambridge, Massachusetts*

Publications

1. **Zhang, C.**; Pitman, M.; Dixit, A.; Leelananda, S.; Palacci, H.; Lawler, M.; Belyanskaya, S.; Grady, L.; Franklin, J.; Tilmans, N.; Mobley, D. L. "Building-Block Based Binding Predictions for DNA-Encoded Libraries." *J. Chem. Inf. Model.*, **2023**, 63, 16, 5120-5132.

2. **Zhang, C.**; Osato, M.; Mobley, D. L. "Characterizing Discrete Binding Conformations of T4 L99A via Markov State Modeling." *in review*

3. Fitzgerald, P; Dixit, A.; **Zhang, C.**; Mobley, D. L., Paegel, B. "A Building Block Centric Approach to DNA-Encoded Library Design" *in review*

Talks and Presentations

1. "Building-Block Based Binding Predictions for DNA-Encoded Libraries", November 2023, MOMA Therapeutics (virtual)

2. "Building-Block Based Binding Predictions for DNA-Encoded Libraries", April 2023, Relay Therapeutics (virtual)

3. "Similarity Searching of Building Blocks in DNA-Encoded Libraries", March 2023, Pittcon, Philadelphia, PA

4. "Similarity Searching of Building Blocks in DNA-Encoded Libraries ", poster at OpenEye CUP, March 2023, Santa Fe, NM

5. "Discovering Active Building Blocks in DNA-Encoded Libraries using 2D and 3D Similarity Scoring", poster at OpenEye CUP, March 2022, Santa Fe, NM

6. "Using Machine Learning to Predict Building Block Yields for the Development of DNA Encoded Libraries", UCI Physical Sciences Machine Learning Nexus, March 2021

ABSTRACT OF THE DISSERTATION

Computational Methods in Drug Discovery: From Molecular Modeling To Library Design

By

Chris Zhang

Doctor of Philosophy in Chemistry

University of California, Irvine, 2024

Professor David Mobley, Chair

Advancements in computational methods have significantly impacted the field of drug discovery by enabling the exploration of complex molecular interactions and the design of diverse chemical libraries. This dissertation presents a study into various computational approaches aimed at enhancing the efficiency and efficacy of early stage drug discovery. In Chapter 2, we explore how machine learning methods can be used to more efficiently select compounds within large chemical databases. We demonstrate how active learning approaches identify promising drug candidates with reduced computational cost and how machine learning (ML) models can be used to filter large chemical databases. We then shift our focus to characterizing discrete binding conformations of T4 L99A using Markov state models (MSMs) in Chapter 3. Using MSMs, we characterize the dynamic behavior of protein-ligand interactions and provide insights into the binding mechanisms crucial to rational drug design that need to be addressed in future studies. Chapters 4 and 5 delve into strategies for building block selection in DNA-encoded library (DEL) design. Leveraging building block-centric approaches, we provide guidelines to construct libraries under specific design constraints and develop predictive models to inform prior additional computational and experimental follow-up. Collectively, we discuss a diverse set of computational techniques which we hope will lead to more efficient and effective strategies for drug design and library construction in the future.

Chapter 1

Introduction

Reliable candidate discovery is a major bottleneck in the drug development pipeline. Identification and validation of compounds in pre-clinical trials span many years with thousands of potential candidates whittled down to a single approved drug [**?** 57]. It has been estimated that in current (at the time of writing) drug discovery efforts, the average cost of developing a drug is around $2 billion USD [71, 29]. Modern developments to *in silico* methods seek to remedy the time and cost drain of early stage discovery, providing information on potential therapeutic targets and ligand properties without the need for time- and resource-intensive experiments [91, 33].

The rapidly growing size of chemical libraries in recent years has been spurred on by the improvement in both experimental and computational methods. Namely, among large virtual chemical libraries, the Enamine database boasts over 6B readily synthesizable compounds for purchase [35]. While the growing size of virtual databases provides more potential for identifying promising drug candidates in early stage screening, managing and navigating these libraries becomes increasingly challenging [48, 75, 46, 62]. In Chapter 2, we discuss the growing adoption of machine learning (ML) methods for efficient sampling of large virtual screening databases. We focus on a procedure known as active learning, whereby carefully selected examples are provided to an ML model to allow for accelerated sampling of an underlying chemical landscape [129, 136, 118, 103]. We show the synergy between active learning and current computational tools and demonstrate the relative ease of implementing an active learning procedure using open-source methods. We also discuss areas in virtual screening where industry is developing tools and showcase some results as part of work with Janssen Pharmaceuticals. Lastly, we conclude by outlining practical considerations and challenges when running an active learning protocol.

Typically, the goal of virtual screening is to identify small molecules that have high binding affinity to a target protein. However, identifying compounds that bind favorably to a target is no simple feat. As dynamic structures, proteins are constantly changing shape. This is an important consideration for current computational methods which initialize proteins from static starting structures. In particular, binding free energy calculations have been widely utilized due to their ability to provide thermodynamic information on potential drug candidates while only requiring simulation data generated from molecular dynamics (MD) [23, 91, 33]. These calculations serve as a reasonably cheap method to compare the viability of different ligands to a protein target and have become a staple in computational drug discovery [89]. However, current implementations fail to account for changes in protein conformation upon binding to different small molecules, leading to convergence issues in binding free energy calculations for even the simplest of systems [73, 90]. In Chapter 3, we investigate the conformational changes in the binding pocket of the L99A mutant of T4 lysozyme, a model system for binding free energy calculations. We find that we can resolve states consistent with experiment [87] using the slow dynamics of the system, providing us a new way to define these conformational states. We estimate that these conformational changes are far slower than the typical simulation timescales used for binding studies, motivating the need to further develop and employ enhanced sampling methods to deal with protein conformational changes in different systems.

The improvement of high-throughput experimental techniques has necessitated the development of improved computational models. Specifically, DNA-encoded libraries (DEL) are a rapidly advancing technology with rich opportunities for computational analysis. In DEL, researchers iteratively combine small molecule building blocks (BBs) encoded with short DNA sequences together to rapidly build up libraries. Each resultant library product is attached to a unique DNA barcode, allowing for rapid decoding of products powered by next-generation sequencing methods [13, 20]. As a combinatorial method, DELs are capable of generating high volumes of binding data in a single experiment. In Chapter 4, we describe how DEL data can be analyzed at the building block level to inform further library design. Using a combination of chemical similarity scoring, dimensionality reduction and clustering, we show that chemically similar building blocks have similar probabilities of forming compounds that bind to a target. We find that even using simple ML models can vastly improve our ability to predict whether new compounds are likely to bind to a target. Lastly, we discuss how we believe informatics-based approaches can be used to improve DEL design in the future.

Building upon the idea of using computational techniques to guide library design, Chapter 5 showcases how cheminformatics can be used to improve building block selection prior to synthesizing DELs. Using publicly available building block catalogs from Enamine, we enumerate various DELs constructed through different design decisions. We introduce different selection rules that can be used to sample BBs. Furthermore, we vary parameters such as BB cost and molecular weight to evaluate their impact on products of the resultant library. We conclude by evaluating our different selection rules on a dataset containing experimental binding information and show how increased diversity of BBs leads to greater likelihood of forming products that bind to a target.

In the final chapter, Chapter 6, we conclude by discussing future work. As the DEL field continues to progress and new methods for analyzing DEL data are being developed, we describe how establishing a DEL benchmark would benefit the field. We outline a proposal for standardizing how to evaluate new DEL x ML-based methods, drawn heavily from MoleculeNet [133], an open-source tool for benchmarking machine learning on chemical datasets. Moreover, we provide an overview of currently available public DEL datasets and describe the need for more open-access DEL datasets in the future. Lastly, we outline opportunities to improve ML predictions on DEL data, namely by incorporating more chemically-relevant data such as 3D docking information [116].

Chapter 2

Efficient Selection of Compounds in Large Virtual Screening Libraries using Active Learning

Modern chemical databases are vast in size and scope, spurred on by the improvement in both experimental and computational methods. This has driven forward the need to develop methods that efficiently screen through large volumes of data and identify promising candidates for drug discovery. In this chapter, we discuss the growing adoption of machine learning (ML) methods for efficient sampling of large virtual screening databases. We outline active learning procedures which allow for accelerated sampling of an underlying chemical landscape and discuss tools in industry that are being developed in tandem with medicinal chemistry efforts to drive forward discovery.

2.1 Introduction

Large virtual screening libraries are valuable for finding starting points for drug development. In recent years, the size and scope of make-on-demand libraries has grown to encompass billions of compounds, enabling computationalists to sift through increasingly large, diverse databases of compounds and propose the most promising candidates for experimental follow-up [75, 46, 62]. Among large chemical databases, the Enamine REAL database is one of the most widely used and currently encompasses over 6 billion compounds (at time of writing) [35]. Apart from spanning diverse regions of chemical space, compounds in the Enamine REAL database can be synthesized on-demand and all adhere to physicochemical property (PCP) filters, with the least stringent adhering to Lipinski's Rule of Five (Ro5) [74] and Veber criteria [127]. Filtering on properties such as molecular weight, SLogP (a measure of hydrophobicity) [132], the number of hydrogen bond donors/acceptors and the total polar surface area of a compound are thought to lead one into regions of chemical space that are more "drug-like" [74, 11]. Thus, the Enamine REAL database offers advantages in that it not only boasts a vast selection of compounds that can be feasibly synthesized experimentally, but that the available compounds are more likely to be pharmaceutically relevant based on their physicochemical properties.

With the growing size of chemical databases, practically screening through them has become more computationally difficult. Active learning has been proposed as a way to tackle this issue [129, 136, 118, 103]. In active learning, researchers strategically select informative examples from a larger dataset in order to train a machine learning model to serve as a surrogate for gathering information. This is especially useful when methods of labeling the data, such as computational techniques or wet lab experiments, may be too impractical or costly [114, 30]. By learning from carefully chosen examples from a broader dataset, an ML model acquires the ability to navigate the information landscape more effectively than brute force approaches. The typical process for active learning is as follows (Figure 2.1):

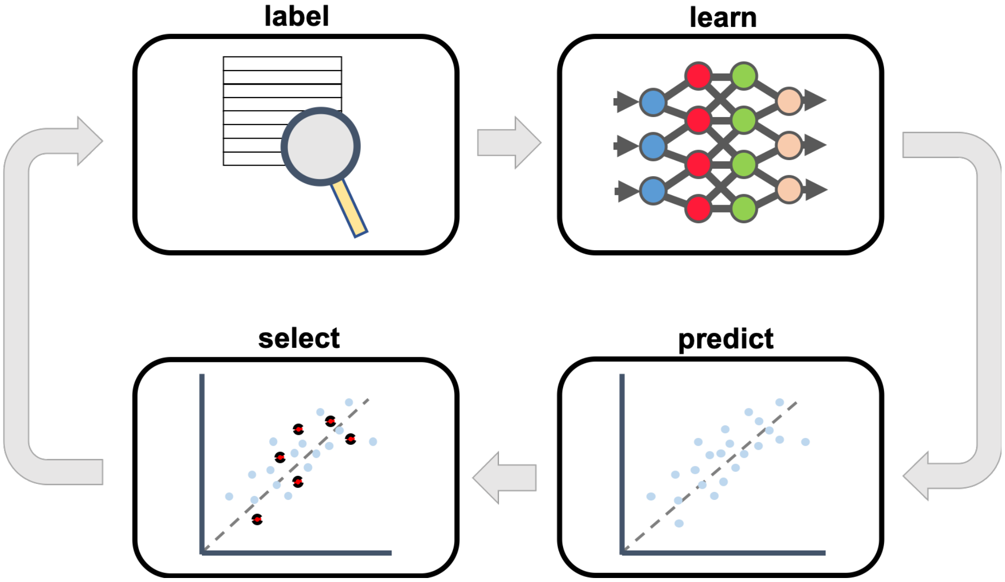


Figure 2.1: Schematic of active learning for drug discovery. In our active learning protocol, we cycle through the following steps: label, learn, predict and select. In the label step, we use an oracle to retrieve labels for some amount of data. This could be by performing an experiment or using a computational method such as docking or binding free energy calculations. Then in the learn stage, we pass our labeled data to a machine learning model so that it can be trained to approximate the oracle. We apply our ML model on the remainder of the dataset in the predict stage, in order to estimate how the oracle would label each point. Finally, we select new points using different selection strategies to pass to the oracle. The process is typically repeated for several cycles.

(1) We label some amount of data using the method we wish to approximate. Whether the method is an experimental or computational technique, we assume the method to be a source of ground truth and refer to it as the **oracle**. The choice of what the oracle could be varies for different research goals, but studies in the literature include using molecular docking [135, 48] or binding free energy calculations [65, 123] as the oracle.

(2) Once the oracle labels the data, we pass the data to a machine learning (ML) model for training. During this learning step, the ML model is provided more information to improve its predictions. The goal is that with enough informative examples, the ML predictions are ultimately able to serve as a reasonable substitute for the oracle on new data points.

(3) After training, we apply the ML model on all the remaining data to generate predictions.

Once ML models have been trained, we can generate predictions on a large number of data points with relative ease [134].

(4) Based on the ML model's predictions, we apply various **selection rules** to select the next batch of data to pass to the oracle for labeling. We then add the newly labeled data points to the previous set of training data to repeat the cycle.

In the literature, it has been reported that choice of selection strategy is one of the most critical factors impacting active learning performance [125, 65]. Depending on specific research goals, different selection strategies have their respective merits. Broadly speaking, selection rules tend to fall into one of two categories: exploration and exploitation. In the exploration-based approaches, the active learning protocol prioritizes selecting data points that the model is most uncertain about or most dissimilar to what the model has already seen. The rationale behind exploration-based approaches is to create a more robust model that better understands characteristics of an entire dataset [103]. On the other hand, exploitation-based approaches hone in on the top predictions made by the ML model and strive to identify more instances of similar data points with high performance. While this approach has been demonstrated to yield high recovery of top compounds with a large reduction in sampling, it has also been found that the value add of exploitation-based approaches diminishes over many active learning iterations [65]. Several works in the literature have also reported balancing both exploratory and exploitative schemes to achieve greater success [123, 49].

In this chapter, we assess how different selection strategies affect how an ML model learns. We explore the merits of various selection strategies, comparing exploration-focused, exploitation-focused and balanced schemes in both their ability to improve recall of top docking hits and the correlation between ML score and docking score. Our study is based on benchmark study by Schrodinger in which they used their proprietary software known as Active Learning Glide [135]. We showcase two approaches to reproducing this study. Our first approach uses OpenEye tools for docking and the open-source Python package chemprop [134] for

ML training and prediction. In our second approach, we describe tools being developed at Janssen Pharmaceuticals in conjunction with active learning protocols to further enhance efficient selection of compounds from chemical libraries. Lastly, we conclude this chapter by discussing future outlooks on active learning in drug discovery and existing opportunities for expansion.

2.2 Methods

2.2.1 D4 Receptor Docking

We followed the procedure reported in benchmark by Schrodinger using their Active Learning Glide method [135] with some modifications. We selected random compounds from Enamine to dock to the human D4 dopamine receptor (PDB code: 5WIU) using different docking software. We stored all docking results in a csv file containing the SMILES and docking score of each compound which we used to create different training and test sets. We refer to this file as our **answer key**. In typical active learning challenges, we would not have access to an answer key because it is not feasible to submit the entire dataset to the oracle for labeling. However, this was a more computationally tractable experiment where we could pre-calculate all answers to better assess how well ML models performed. At steps where we would typically submit compounds to the oracle for labeling – equivalent here to running docking calculations for those compounds – we instead revealed the answer by looking up the docking score for those compounds in our pre-computed answer key. Following Schrodinger's terminology, we designated the top 0.1% of the compounds in the dataset by docking score as our **docking hits**. After each round of active learning, we calculated the percentage of docking hits we recovered as a function of the percentage of the database screened. We defined this as our **recall** at a percentage of the database screened. In addition, we also evaluated the correlation between docking score and ML-predicted score after each round of the active learning protocol.

Open-Source Implementation

We obtained a raw structure file for the human D4 dopamine receptor from the Protein Data Bank (PDB ID: 5WIU) and prepared the protein for docking using OpenEye's Spruce toolkit (v.2023.1.1) [97], first converting the structure into a design unit in order to make a receptor file. For our ligands, we used a set of 1M randomly selected ligands from Enamine provided through the "Evaluating Large Ligand Libraries with Active Learning Glide" tutorial from Schrodinger [115]. We used the OpenEye OMEGA application (v.4.2.2.0) to generate conformers for each of the 1M ligands. Having generated both a receptor and ligand file, we performed docking using OpenEye's HYBRID (v.4.2.1.0) to dock each ligand to the receptor and took the pose with the highest docking score as the label for each compound. Although we ran docking for 1M ligands, due to some complications with the protocol (outlined at the end of this chapter), we only conducted active learning for 200K docked compounds.

Industry Implementation

To prepare the target protein, we ran the Protein Preparation workflow on Maestro. We toggled settings to initialize the system with pH 7.4, sample water orientations, remove non-water solvents, flip side-chains to preserve H-bond networks and adjust for potential atom clashes. Following these adjustments, we generated a grid file that defined the shape and properties for the receptor necessary for docking. We used the ligand present in the 5WIU structure to define the region a docked molecule must occupy to satisfy the initial requirements of docking. We used Schrodinger's LigPrep to generate 3D conformations for each ligand and assign protonation states at pH 7.4. We opted to ignore any information about chirality reported in the original ligand file and instead determine chirality from the generated 3D structures.

2.2.2 Selection Rules

We implemented the same four different selection strategies as reported by Schrodinger, which we refer to as: `top`, `top_rand`, `uncert` and `top_uncert`. We considered the `top` and `top_rand` approaches to be exploitation-based selection rules, `uncert` to be an exploration-based scheme and `top_uncert` to be a balanced approach. In the `top` selection strategy, the compounds with predicted docking scores in the highest 0.1% of the data were selected to be evaluated by the oracle. With `top_rand`, a random 0.1% of the data was randomly selected from the top 10% of compounds with the highest ML predicted docking score. On the other hand, as an exploration-focused strategy, `uncert` selected the most uncertain 0.1% of the ML predictions. Using our ML tools, we quantified uncertainty as the standard deviation of the predicted docking scores for the base classifier models in our ensemble. Lastly, in the `top_uncert` selection rule, the most uncertain 0.1% of compounds was selected from the top 5% by ML predicted score. We also included an additional selection rule which chose a random 0.1% of compounds from the dataset regardless of ML predictions to serve as a null model. In both our active learning implementations, we wrote a series of scripts to select the next round of ligands given the docking scores and uncertainties for each compound in the previous round. We exported the results of ML predictions from each round into csv files and analyzed those using the pandas (v.2.0.1) [102] toolkit. We implemented each selection rule by sorting the data in a specific order and exporting the relevant structures into new output files.

2.2.3 Machine Learning Models

Open-Source Implementation

We used the chemprop package (v.1.6.1) [134] to build and train message passing neural networks for predicting docking scores. To build an initial model, we called the `chemprop_train` function, setting the dataset type to 'regression' and the number of training epochs to 100. Additionally, to get uncertainty estimates on our predictions, we opted for an ensemble of models. We set the ensemble size parameter to 5 as recommended by the authors. During training, chemprop identified an optimal set of hyperparameters for each model using bayesian optimization. Once training was completed, we called the `chemprop_predict` function to generate ML predictions. We referenced our previously trained model by providing the appropriate file path in the checkpoint directory argument. Furthermore, we set the uncertainty method to 'ensemble', which used the standard deviation among the ensemble model predictions as the uncertainty value of each ML prediction.

Industry Implementation

We had access to two different ML modeling tools: Schrodinger's DeepAutoQSAR and an internal JnJ method (Figure 2.2). Both tools combine a series of base classifier models in an ensemble to create a model with better performance than any model individually. Referred to as automated machine learning (AutoML), these methods formulate hyperparameter selection as an optimization problem in order to create off-the-shelf ML workflows [37]. To run DeepAutoQSAR, we provided an input file containing SMILES strings for each compound and its docking score to the receptor. We specified a training time of 4 hours as it was reported by Schrodinger that there were diminishing returns on model performance beyond this time span [135]. During this specified training time, DeepAutoQSAR iterated through combinations of both simple and graph-based architectures (deep neural networks, random forests and gradient boosted trees) and returned the ensemble model with the lowest error on a validation set. The final trained model was returned as a file that could be called for prediction. We provided a similar input for JnJ's internal AutoML tool, specifying a training time of 4 hours. AutoML is thought to be an improvement upon DeepAutoQSAR because of its featurization and final estimation steps (Figure 2.2B). In addition to standard molecular featurizations such as extended connectivity fingerprints (ECFP) [105] used in DeepAutoQSAR, AutoML includes an ADME featurization based on the Design Enablement models [67, 22, 126, 27, 121].

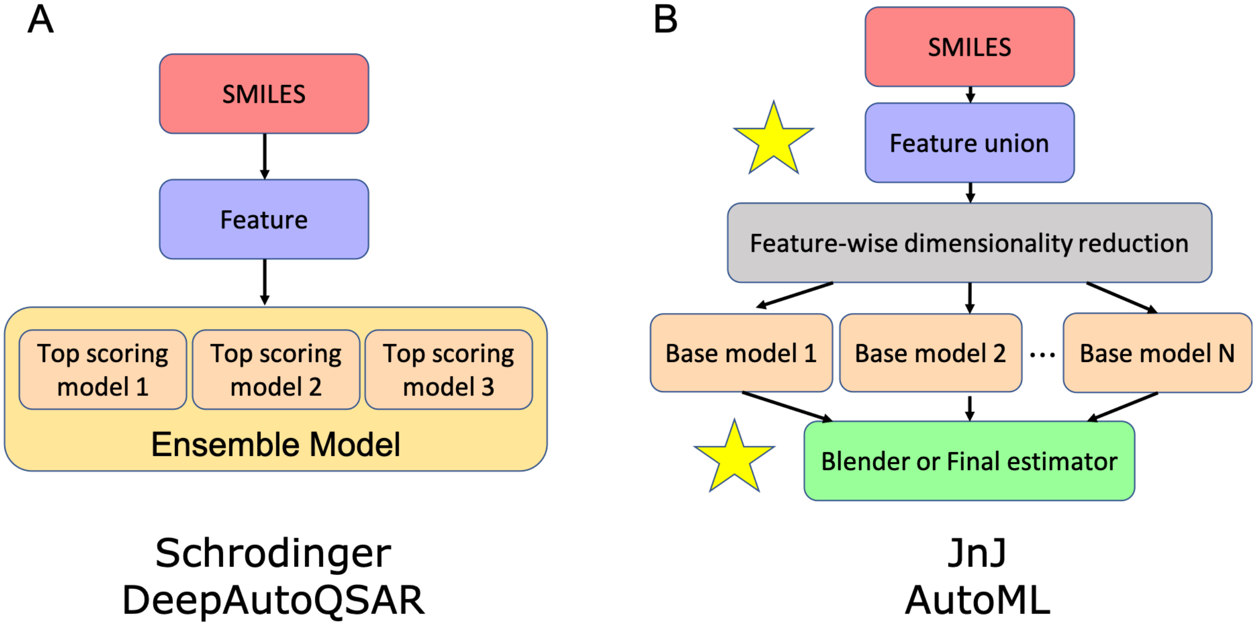


Figure 2.2: Model architectures for (A) DeepAutoQSAR and (B) AutoML. Both architectures read in compounds as SMILES and generate features for input into a number of base classifier models. Moreover, AutoML includes a set of more sophisticated features based on internal compound data and a final blending step to aggregate the prediction of the individual classifier models.

2.2.4 Janssen Proprietary Models

Both the features in AutoML and the Design Enablement models were based off underlying transformer models [67, 22, 126, 27, 121]. Briefly, transformers operate using an "attention" mechanism, whereby the model learns meaning and context of individual components of data by tracking the relationships between them [126]. In their commonly used application in text parsing, transformers learn the semantic context of a word by calculating weights for each word based on how it relates to every other word [27]. This application can be extended to drug discovery, where models are trained to learn SMILES syntax and subsequently predict molecular properties [63]. The Design Enablement models were trained on the absorption, distribution, metabolism and excretion (ADME) data for all compounds in the internal Johnson & Johnson (JnJ) databases [67].

2.3 Results

2.3.1 Active learning provides large improvement over brute force sampling

Open-Source Implementation

We find that regardless of choice of selection rule, active learning provides a significant advantage to brute force sampling of compounds. To begin the active learning protocol, we select a random 0.1% of compounds and their associated docking scores to pass to chemprop in order to build our first ML model. We then use that model to predict the docking scores on the remaining 99.9% of the dataset. We refer to this as round 0 of our active learning procedure, since we did not apply any selection rules to pick this initial set of compounds. However, even while using a completely random set of training data, we find that ML predictions far exceed what would be expected from random guessing. At 1% of the data sampled, we would expect a brute force procedure to identify 1% of the total docking hits in the data. However, we find that our ML model performs an order of magnitude better, recalling 12% of the docking hits (Figure 2.3).

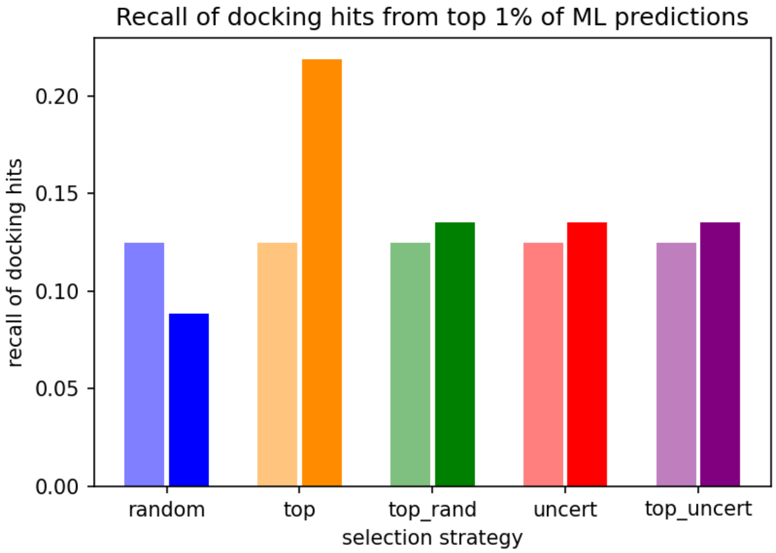


Figure 2.3: Recall of docking hits for different active learning selection strategies. Here, we show the improvement in the recall of docking hits identified after one round active learning. We initialize all models with the same random 0.1% of the data so the lighter shaded bar in each group, representing the results from round 0, have the same height. We represent the result after one round of active learning, round 1, with the darker shaded bar in each group. `Random` selection (blue) serves as a null model for reference. Both the `top` (orange) and `top_rand` (green) rules are exploitation-focused strategies whereas the `uncert` selection rule (red) is an exploration-based approach. The `top_uncert` strategy (purple) is a balance of both exploitation and exploration. After one round of active learning, the `top` selection strategy recalls the greatest number of docking hits with the top 1% of ML predictions.

We observe that differences in the selection rules start to appear following round 1. We apply our various selection strategies to the ML predictions from round 0 in order to identify new compounds to add to the existing training set. Since each selection rule is different, we end up with five distinct training sets for round 1. Each round 1 training set is the concatenation of the training set from round 0 and the new points chosen by the corresponding selection rule. Using chemprop, we are able to construct a new model using an existing model as the starting point, rather than training a new model from scratch. We create five new ML models, where each one is built off the round 0 model but trained with a different round 1 training set. We find that one of our exploitation-based approaches, `top`, experiences the most appreciable increase in the recall of docking hits, almost doubling from its value in the previous round to 22% (Figure 2.3). However, the other exploitation-focused approach, `top_rand`, does not result in a similarly large increase of recall, sharing the same increase as the exploration-based and balanced approaches `uncert` and `top_uncert` to 14%. Lastly, we find that the `random` selection rule actually results in a decrease in recall compared to its previous round performance, going from 12% down to 9% (Figure 2.3).

When repeating the same procedure for rounds 2 and 3, we find that all selection rules result in an increase in recall, but to different extents (Figure 2.4A). While we observe that on the whole every selection rule results in continued improvement over active learning cycles, the differences in which compounds are added to the training set result in different jumps in performance at each round. For example, we find that the `top`, `top_uncert` and `random` methods experience a modest increase in recall compared to their round 1 performances (4%, 3% and 5% respectively). On the other hand, the recall for the `top_rand` and `uncert` methods jump much more significantly compared to their previous round performance (7% and 11% respectively) (Figure 2.4A). Finally in the last round, we find that while all methods continue to result in an increase in recall, the `uncert` selection rule results in the greatest increase compared to its previous round (a jump of 8%) and results in the highest recall of docking hits among all the methods (Figure 2.4A).

When analyzing the changes in correlation between docking score and ML-predicted score across active learning rounds, we find far more variation. Interestingly, we observe that the `random` selection rule results in one of the most consistent increases in correlation as rounds progress, second only to the `uncert` method, which is focused on exploring the underlying chemical space (Figure 2.4B). However, none of the selection strategies result in strong correlation, with the highest performing `uncert` selection rule resulting in a Pearson correlation of only 0.41 after three rounds of AL (Figure 2.4B).

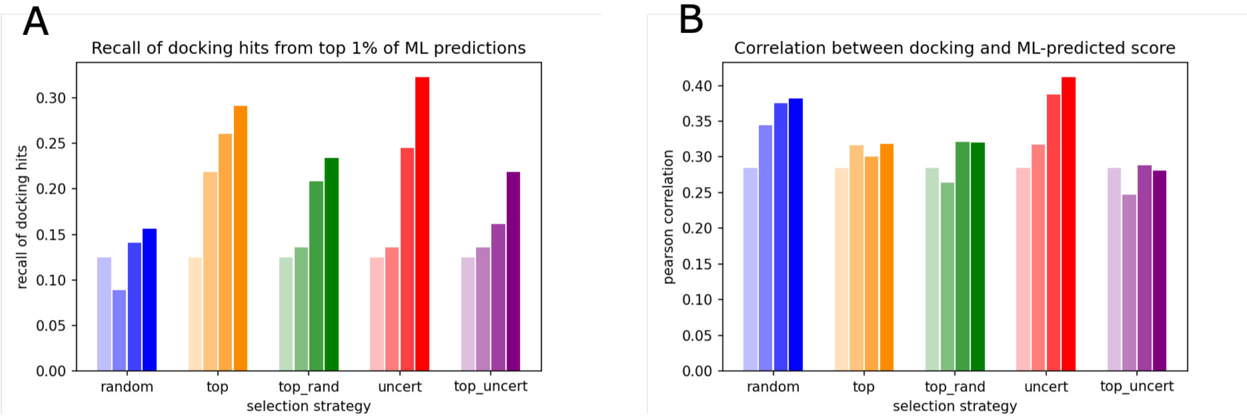


Figure 2.4: Recall of docking hits and correlation between docking and ML-predicted score over multiple active learning cycles. We assess both (A) the recall of docking hits as well as (B) the correlation between docking and ML-predicted docking scores across three rounds of active learning. For each selection rule, we show the results for successive rounds of active learning illustrated by bars of the same color with progressively darker shading. We observe that recall of top docking hits generally increases for each subsequent round, whereas correlation is more prone to fluctuation.

We believe the results from the data demonstrate the value add of active learning approaches. After an initialization step and three rounds of active learning, corresponding to 0.4% of the data sampled, we find that our best performing method is able to rank over 30% of the docking hits within the top 1% of its predictions. Moreover, even when randomly selecting compounds for multiple active learning cycles, we find that over 15% of the docking hits are within the top 1% of the ML predictions – an order of magnitude greater than what we would expect from a brute force approach (Figure 2.4A). Interestingly, we find that the `uncert` selection rule to be best at recalling docking hits, despite being an exploration-based method. We posit that this may be due to the `uncert` selection rule resulting in the best learning of the underlying landscape, as evidenced by its Pearson correlation score compared to other methods after three AL cycles (Figure 2.4B).

Given the weak correlation results yielded by every selection strategy, we believe our models may benefit from longer training times or more training data per cycle. Nonetheless, it is compelling that potentially under-trained models are still able to recall docking hits more than an order of magnitude better than brute force. This observation also reinforces the related but separate nature of exploitation and exploration-based approaches for active learning [103, 129]. Even while having low correlation between their predictions and docking scores, models are able to successfully recall many docking hits. Thus, the two tasks seem related but not dependent on each other. Being able to identify a docking hit does not require the ML model to perfectly predict docking scores, but being able to better predict docking scores improves the ability to recover docking hits.

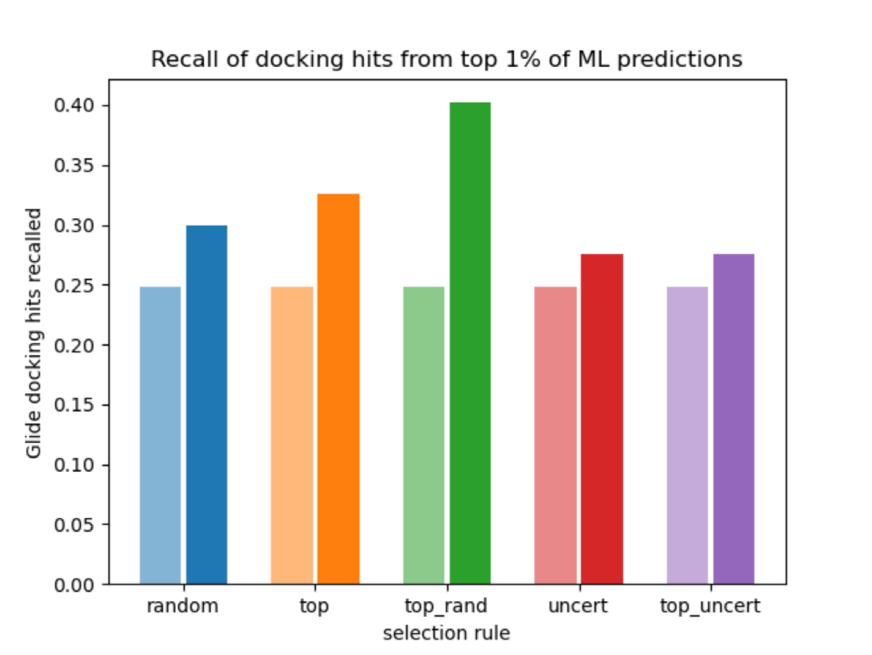


Figure 2.5: Assessment of different active learning selection rules. Here, we demonstrate the improvement in the fraction of Glide docking hits identified after one round of applying various selection rules. The lighter color bar for each selection rule indicates the same random initialization of 0.1% of the data. The two rules based on exploitative approaches, `top` and `top_rand`, result in greater recall of hits than random selection after one round of active learning. On the other hand, the exploration-based approaches, `uncert` and the balanced approach, `top_uncert`, result in lower recall than random after one round of active learning.

Industry Implementation Results

While our industry implementation follows a very similar cadence to the open-source procedure, we find that there are noticeable differences in the performance of AutoML compared to chemprop across active learning cycles.

In round 0, we select a random 0.1% of the compound SMILES and respective docking scores to feed to AutoML for training. After training, we predict the docking scores of all the remaining compounds in the test set. Even though we initialize with random compounds, we find that AutoML recalls 25% of the docking hits in the top 1% of its predictions (Figure 2.5), doubling that of chemprop (Figure 2.3). We subsequently apply our selection rules for round 1 of active learning. Similar to with chemprop, we find that the differences in selection rules emerge during round 1. We observe the exploitation-based strategies perform better at recovering top docking hits compared to the exploration-based and balanced strategies (Figure 2.5). The `random` selection rule results in a recovery of 30% of the docking hits, whereas the `top` and `top_rand` strategies recall 32% and 40%, respectively. On the other hand, the exploration-based and balanced strategies lag behind random selection, both only recalling 27% of the docking hits.

As we continue our procedure for a total of three rounds of active learning (excluding the initial random selection), we find that the recall of docking hits improves but with diminishing returns over AL cycles. As we would naively expect, both exploitation-based strategies result in more docking hits recall than random selection after three cycles. However, we also find that the exploration-based strategy performs worse than random (Figure 2.6A). Interestingly, we observe that the `top_uncert` approach improves *more* in recalling top docking hits at each subsequent active learning round with a significant jump between the second and third rounds.

In addition to looking at the recovery of docking hits, we also evaluate the correlation between calculated and ML-predicted docking scores. We posit that exploration-based approaches may provide more informative examples to help the ML model make generally more accurate predictions, which we thought would manifest in improved correlation. However, we do not find this to be the case. The relative performance of the various selection rules in terms of correlation is almost identical to their performance in terms of recall, with slight difference that the exploration-based approach outperforms random selection (Figure 2.6B). Consistent with what was reported by Schrodinger [135], we find the `top_uncert` to have the best performance both in terms of recall of top docking hits and correlation between calculated and ML-predicted docking scores after three rounds of active learning. We posit that this is consistent with the notion of the `top_uncert` approach being a balanced selection rule. After enabling the ML model to sufficiently explore the landscape by observing informative examples, the balanced selection rule allows the ML model to become increasingly capable of identifying top docking hits to the target. In support of this hypothesis, we find that the `top_uncert` method also results in the highest Pearson correlation between docking and ML-learned docking scores after three rounds of active learning (Figure 2.6B).

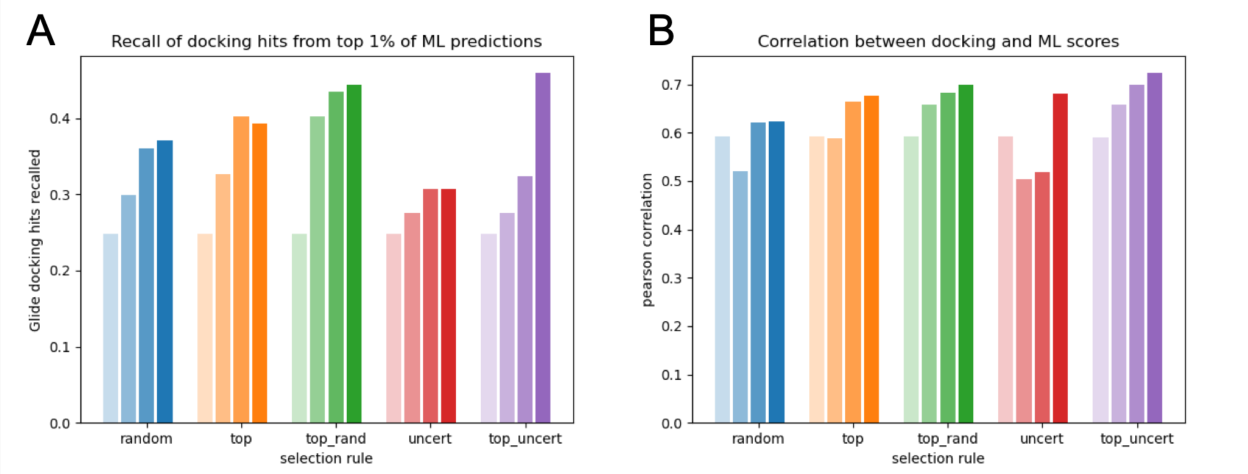


Figure 2.6: Results from active learning over multiple iterations using JnJ's AutoML. We assess both (A) the recall of docking hits as well as (B) the correlation between docking and ML-predicted docking scores across three rounds of active learning. For each selection rule, we show the results for successive rounds of active learning illustrated by bars of the same color that get progressively darker. Generally speaking, we observe diminishing returns for improvements in both recall and correlation over multiple active learning rounds.

The results from the open-source and industry active learning procedures share some similarities but also have major differences. The most notable similarity is the observation that regardless of selection rule, an ML model's ability to recall docking hits improves over successive active learning cycles. While this does not come as a surprising result, it serves as validation that the overall AL scheme is being designed correctly. However, the results between using chemprop versus AutoML is striking. Whereas we find the `uncert` selection rule results in the most successful model both in terms of recall and correlation using chemprop (Figure 2.4), it results in a low performing model when using AutoML (Figure 2.6). Conversely, we observe that the `top_uncert` selection rule has lackluster performance using chemprop (Figure 2.4) but results in the best performance for both recall and correlation using AutoML (Figure 2.6). While part of the discrepancy can be attributed to the difference in datasets for each protocol (200K with chemprop vs 1M with AutoML), we believe this should not impact the relative trends of the selection rules. Thus, we attribute the variation of results to the differences in the underlying ML modeling tools. In particular, AutoML benefits from features derived from underlying transformer models trained on absorption, distribution, metabolism and excretion (ADME) property data (see Methods for more details). We introduce and report on these transformer models, known as the Design Enablement (DE) models, in the following section. We showcase the potential for these models to filter the Enamine dataset (and subsequently other large chemical databases) by their predicted ADME properties.

2.3.2 Machine learning can be used to filter compounds beyond physicochemical properties

The Enamine REAL database can be divided into subsets, with increasingly stringent physicochemical property filters [35]. A stricter subset of the REAL space is the **lead-like** space (3.8B molecules), which enforces more constraints on molecular weight, SLogP, hydrogen bond acceptor (HBA) and ring count for compounds. The lead-like space can be further subset into the even more restrictive **350/3** space (620M molecules), with a more narrow range of allowed molecular weight and SLogP. In addition, the 350/3 space filters compounds by their heavy atom count and the number of aryl rings they contain (Table 2.1). While guidelines for physicochemical properties such as Lipinski's Rule of Five can help identify compounds with drug-like properties, many approved drugs also violate these rules [53]. Thus, despite the rationale that more restrictive filtering on physicochemical properties leads to more pharmaceutically relevant candidates, these filters do not paint a full picture of the viability of a compound for drug development.

Enamine subset	Property Filters							
	MW	SLogP	HBA	HBD	Rot. Bonds	Heavy Atoms	TPSA	Rings
REAL	≤ 500	≤ 5	≤ 10	≤ 5	≤ 10	N/A	≤ 140	N/A
Lead-Like	≤ 460	$-4 \leq x \leq 4.2$	≤ 9	≤ 5	≤ 10	N/A	≤ 140	rings ≤ 4
350/3	$270 \leq x \leq 350$	$-4 \leq x \leq 3$	≤ 9	≤ 5	≤ 10	$14 \leq x \leq 26$	≤ 140	rings ≤ 4; aryl rings ≤ 2

Table 2.1: Physicochemical property filters for various subsets of Enamine REAL

As part of work with Janssen, we assess whether internal tools can be used as a means of filtering the Enamine REAL dataset. Dubbed the Design Enablement (DE) models, these transformer networks are trained to predict a variety of absorption, distribution, metabolism and excretion (ADME) properties using assay data from the Johnson & Johnson internal database (see Methods) [67, 134, 22, 126, 27, 121]. We reason that if the DE models are sufficiently accurate, they can enable us to filter out compounds from Enamine that do not match the desired pharmocokinetic properties for hits to a target of interest. Typically, these properties cannot be ascertained until the compound is experimentally tested. Thus, using the DE models as a means to pre-filter compounds in Enamine would lower computational costs by reducing the number of compounds and subsequently focusing the search to compounds with more desirable property profiles.

We observe that the increasingly strict physicochemical property filtering implemented for the Enamine database and its subsets does not necessarily make for more pharmaceutically desirable compounds, thus motivating the need for accurate ML-based ADME predictions. As we move from the REAL to lead-like to 350/3 space, we observe a greater density of more soluble compounds, which is desirable for pharmaceutical candidates (Figure 2.7A). However, in the case of experimental permeability, we see that there is little difference in its distribution across the three Enamine subsets (Figure 2.7B). In other words, physicochemical property filtering alone does not guarantee a higher concentration of compounds with favorable ADME profiles. Otherwise we would find the highest concentration of experimentally permeable compounds in the 350/3 set. Therefore, the capacity to predict ADME properties prior to experimental validation would minimize the risk of selecting compounds that appear favorable *in silico* but flounder when tested experimentally. We validate the DE model predictions to assess whether they can provide an accurate enough means of predicting the ADME properties of untested compounds.

We find that the distribution of DE model predictions generally matches the distribution of experimental properties. Searching through the internal JnJ database for compounds with experimentally measured ADME data sourced from Enamine, we create an 8K compound validation set. This set of compounds provides us a means of comparison: for every compound in the validation set, there is an experimental measurement we can compare to our DE prediction. We generate DE predictions for both the solubility and permeability of the compounds in the validation set and compare to their experimental distributions. We find that for both properties, DE predictions generally match experimental distributions but are right-skewed (Figure 2.7C, D). In the case of solubility, the DE predictions capture the bimodal distribution of the experimental distributions and correctly identify that the 350/3

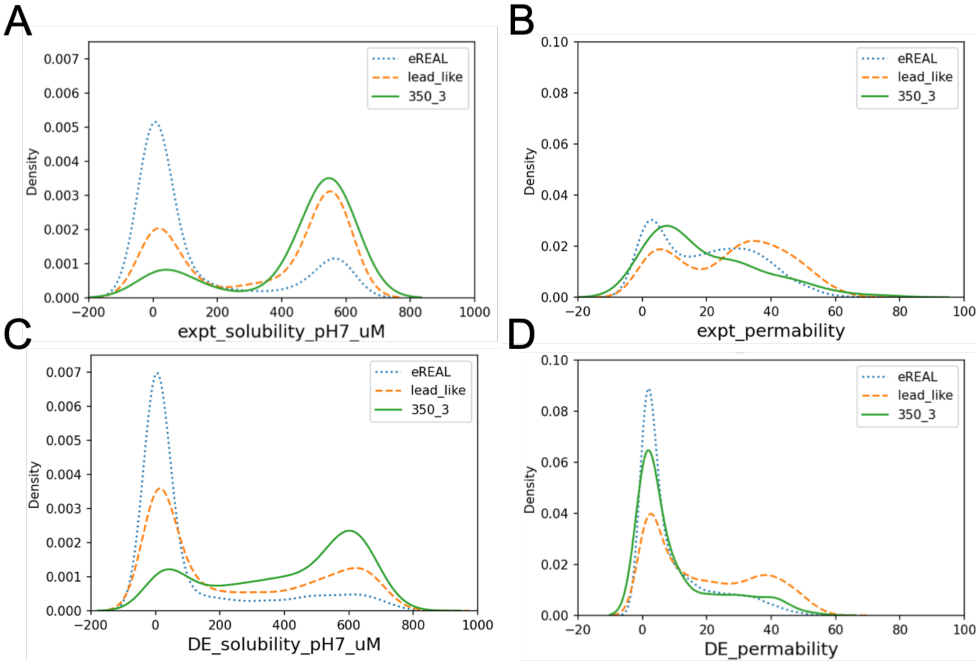


Figure 2.7: Distributions of experimental and DE predicted solubility and permeability on an 8K compound validation set. We divide compounds in the validation set based on which subset of Enamine they fall in to based on their physicochemical properties: (blue dotted line) the REAL space, (orange dashed lines) the lead-like space and (green solid lines) the 350/3 space. We show the distributions for experimental (A) solubility and (B) permeability for compounds in the validation set. This is contrasted with the distributions for DE-predicted (C) solubility and (D) permeability of compounds in the validation set. The distributions of DE model predictions generally match experimental distributions, but are more right-skewed.

set contains a greater concentration of more soluble compounds compared to the lead-like and full REAL sets. However, the model overestimates the number of insoluble compounds and underestimates the number of highly soluble ones (Figure 2.7C). When predicting permeability, the DE model has a more pronounced bias towards false negatives, estimating a higher percentage of compounds to be impermeable (Figure 2.7D).

To quantify the errors made by the DE model, we compare the experimental and DE predictions for the permeability of each compound. We observe a good amount of agreement between experiment and prediction evidenced by a Pearson correlation (r) of 0.83 and a coefficient of determination (R^2) of 0.64 (Figure 2.8). We then set a cutoff value for permeability

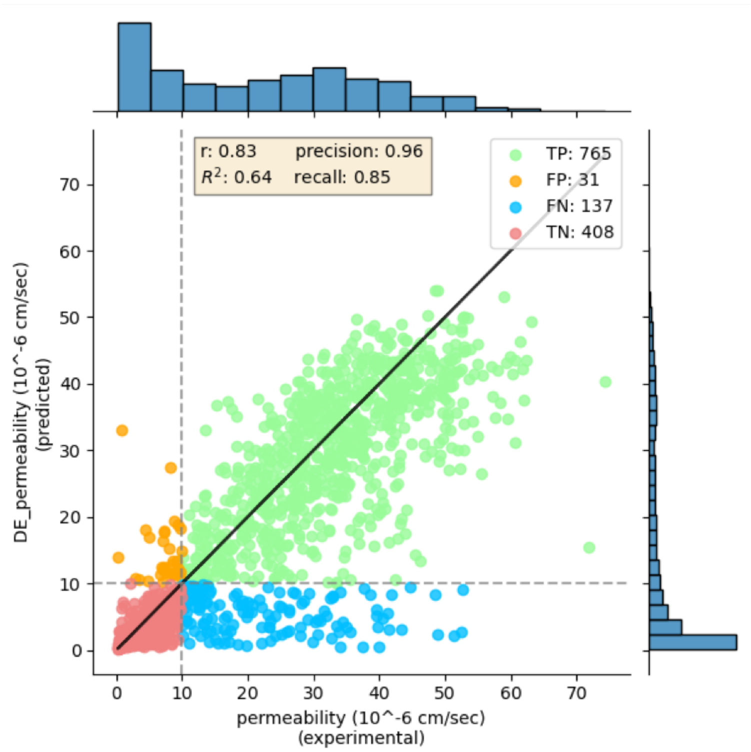


Figure 2.8: Binary classification of DE permeability estimates. We set a binary cutoff based on thresholds from previous internal JnJ projects to evaluate the quality of the DE model predictions of solubility. Compounds that are misclassified are predicted to be permeable when they are not experimentally (FP) and predicted to be impermeable when they are experimentally (FN). We observe more instances of the latter, consistent with the theory that the DE predictions make more conservative predictions.

based on thresholds set by medicinal chemists in previous internal projects to interpret the DE model predictions as a binary classification problem. We consider compounds with experimental permeability greater than 10×10^{-6} cm/sec to be "permeable" and those less than to be "impermeable". This allows us to calculate the number of true positives (TP), false positives (FP), false negatives (FN) and true negatives (TN) present. We find that the precision of the model – defined as $TP/(TP + FP)$ – is equal to 0.96, meaning that of all compounds predicted to be "permeable" by the DE model at the selected threshold, 96% of them have experimental permeability values $\geq 10 \times 10^{-6}$ cm/sec. The recall of the model – defined as $TP/(TP + FN)$ – is equal to 0.85, meaning that the DE model identifies 85% of all compounds with experimental values of permeability $\geq 10 \times 10^{-6}$ cm/sec as "permeable".

As we previously observed in the distribution of DE permeability predictions (Figure 2.7D), the DE model produces a higher number of false negatives, leading to a reduction in recall for increased precision (Figure 2.8).

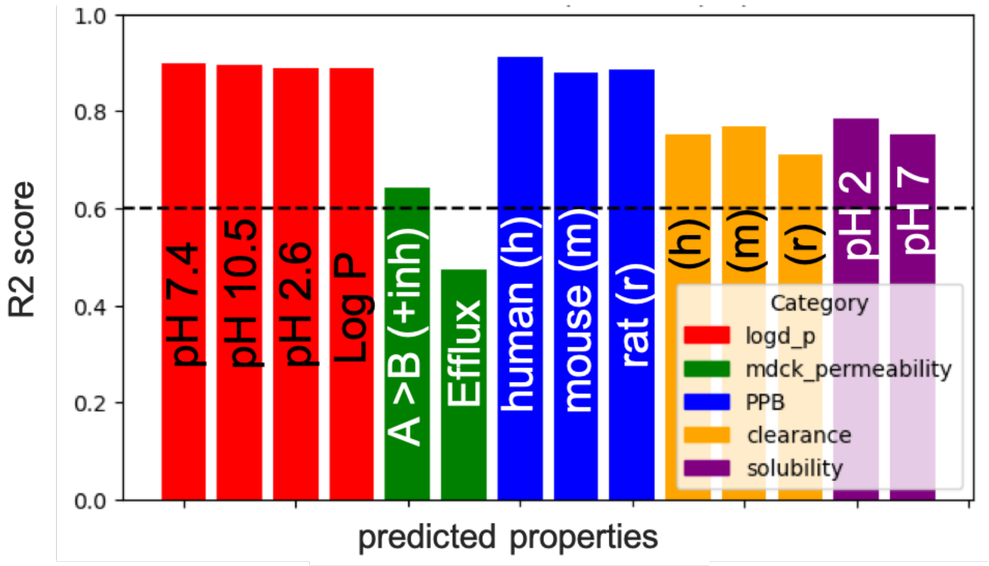


Figure 2.9: R2 scores for selected ADME properties. We evaluated the accuracy of the DE predictions on 14 different ADME properties, which can be broadly categorized into measure of partition coefficients (red), permeability (green), plasma-protein binding (blue), clearance (orange) and solubility (purple). We set a cutoff at a coefficient of determination (R2 score) of 0.6 or greater to consider that property to be well-predicted by the DE models. Apart from the efflux rate for MDCK cells, ADME properties are well-predicted by the DE models.

In addition to solubility and permeability, we use the DE model to predict 12 additional experimental assay values for compounds in the validation set. We group these properties into measures of logD/logP, MDCK permeability, plasma-protein binding (PPB), clearance and solubility [67]. Although every compound in the validation set has some experimental ADME data, not every compound has measurements for all ADME properties. Thus, the properties we choose to evaluate are also determined by the availability of experimental data. We ensure that for every property we choose, there is data for N > 30 compounds. For all properties, we analyze the agreement between DE predictions and experimental values by calculating the coefficient of agreement, R^2. We find that $R^2 > 0.6$ for all properties except efflux rate using MDCK cells [67, 28], suggesting strong agreement between DE predictions and experiment across a variety of ADME properties (Figure 2.9). Based on this finding, we believe the DE model predictions can be used to as a pre-filtering step to isolate compounds with specific ADME properties in the Enamine database. While we identify that the model produces many false negatives, we believe missing some true positives is not deleterious given the large size of the initial Enamine search spaces.

2.4 Discussion

2.4.1 Value Add of Active Learning for Early Stage Database Screening

In this chapter, we highlight the advantage of using *some* form of active learning versus brute force approaches. We find that even an active learning scheme using random selection of compounds outperforms brute force approaches by over an order of magnitude (Figure 2.4A, 2.6A). With our potentially under-trained chemprop implementation, we recover over 30% of our docking hits in the top 1% of ML predictions using the highest performing `uncert` strategy for three cycles of AL (Figure 2.4A). Using AutoML, we find that the highest performing `top_uncert` strategy recovers almost 50% of the docking hits after three cycles (Figure 2.6A). We again posit that the difference in performance is primarily powered by the ADME-based features in AutoML, which are developed using assay data from the Johnson & Johnson internal databases [67, 134, 22, 126, 27, 121]. While both chemprop and JnJ's AutoML are related architectures [134], the difference in performance clearly demonstrates that the "secret sauce" of many industry ML tools lies within the training data.

Although we find that the industry implementation has a noticeable improvement over the open-source tools, we argue that current open-source tools make it quite accessible to build out a simple active learning pipeline. In literature, studies have provided open-source tools to automate active learning pipelines [48, 62, 117] that are compatible with various docking software. We imagine future tools automating active learning workflows incorporating different oracles, such as binding free energy calculations [123, 65, 49].

2.4.2 Challenges in Active Learning

A slew of issues can be encountered while designing an active learning protocol. In the following section, we highlight three important challenges to consider: (1) issues with data set size, (2) choice of oracle and (3) ML model training and prediction costs.

Issues with data set size

Having large data sets to work with is almost necessarily an issue with active learning studies because the purpose of active learning is to efficiently sample from search spaces that cannot be explored via brute force. In our initial attempts to run an active learning study, our proposed dataset was the entirety of the Enamine REAL database, which at the time of writing was over 6 billion compounds [35]. For each of our proposed selection rules, selecting 0.1% of the data to sample and evaluate with the oracle would have required 6 million docking calculations per active learning cycle. Even at a large pharmaceutical company, this level of computational cost would be difficult to manage, and furthermore inefficient as a commercial tool. Our solution to this was to leverage existing internal models to pre-filter and reduce the original search space by approximately an order of magnitude. However, we acknowledge that this solution is not one that is typically available.

Even with this potential solution, there is still an issue with data storage. At 6 billion compounds, even a compressed file takes up multiple GB of storage. While this is not an unreasonable scale to deal with given advances in modern hardware, the active learning workflow requires generating new training and test sets at each round. Likely for reproducibility and tracking purposes, researchers would be interested in keeping all files rather than rewriting after each round. Since each training/test set combination is just a new permutation of the full dataset, this results in duplicating the original dataset by the number of active learning rounds multiplied by the number of selection rules being evaluated. These costs could quickly accrue and become troublesome. Furthermore, the size of the initial dataset directly impacts the computational cost of downstream operations in the workflow – the more data points we have, the more we feed to both the oracle and the ML model, which encounter their own issues with scaling.

Another consideration is the quality of sampling that can be achieved. This means striking a balance between pruning down data sets to reduce costs while highlighting more productive regions of chemical space and throwing out potentially useful data. In our benchmark, we observed a case of this when evaluating our `top_uncert` selection rule. While it appeared to underperform compared to many of other selection rules in earlier stages of active learning, it rapidly jumped up to become the best performing method after the third round (Figure 2.6A). We posit this is due to a ruggedness in the chemical space of the starting dataset; the model was suddenly provided very informative data points in the final round that significantly boosted its performance. Unexpected changes in performance can be symptoms of underlying quality issues in the original dataset.

Our recommendation to these problems is to encourage researchers to find methods to reduce the size of their starting datasets and cluster compounds to draw broader insights. This can be achieved in a number of ways and does not necessarily require the use of commercial tools. For example, one could cluster compounds based on their Bemis-Murcko scaffolds [8, 133, 134] or permanently prune poor-performing compounds from the starting dataset [47]. Yang et al. defines a metric known as "cluster head recovery", where docking hits are first grouped by Tanimoto similarity [135]. A cluster is successfully "recovered" if any ligand in that cluster is identified by the ML model to be a docking hit. Particularly in cases where the underlying chemical space is diverse and there are many singletons, cluster head recovery is a way of quantifying the diversity of chemotypes selected by ML. Lastly, there could potentially be ways to filter the starting data based on known information for the target of interest. For example, the D4 dopamine receptor is known to preferentially bind cationic molecules [135]. Similarly, a docking study on carbonic anhydrase [116] filtered molecules for sulfonamides, a feature present in many known D4 inhibitors [43, 14]. Based on the specific goals of the active learning study, existing knowledge of the system could be used to toss out certain ligands and focus the search space.

Choice of oracle

Depending on choice of oracle, computational costs could vastly differ. For example, while it would be feasible to dock thousands of compounds at each active learning round, it may not be possible to run free energy calculations at that scale. Studies in the literature have shown that providing a larger number of data points in earlier rounds of active learning improves results [123, 125]. Thus, it may be important to select an oracle that is not prohibitive to run at larger scales, in order to maximize the value add of the active learning protocol.

Furthermore, the crux of active learning is that with sufficient training data, an ML model will eventually be able to mimic the knowledge of the oracle. In most cases, an ML model is a much cheaper tool to use for prediction on large data sets. However, it is important to consider the quality of the underlying oracle predictions. In the case for a tool like docking, an ML model would only allow the researcher to identify top *docking* hits, which are not the same as experimental hits [75, 60, 106]. Thus, an interesting conundrum emerges: it is easier to create an ML model capable of replicating the predictions of cheaper methods, but these methods are typically less accurate leading to a final ML model that is not as useful.

ML model training and prediction costs

While using a trained ML model to predict on data is cheaper than almost any computational method, the costs can still accrue when predicting on large enough sets of data. An issue we initially encountered was that model training and prediction times were not trivial, even on a relatively small scale. We originally tried creating ensembles of base classifier models using sklearn, but found that sklearn models are only partially parallelizable and offer no GPU support [100]. We instead turned to chemprop [134], which is built off PyTorch and thus can be accelerated by GPUs. In performance benchmarks for chemprop, the authors report that on a dataset of 100K compounds, training should finish within an hour and inference, as a faster operation, should take less than a few minutes [55]. We found that for a dataset of 200K compounds, training finished within an hour but inference was much *slower*, taking around 15 minutes. Slow prediction times could lead to issues if applied on datasets several orders of magnitude larger for multiple active learning cycles. However, given there is such a large discrepancy between our reported prediction times and those found by the authors – and the fact that in general, inference should be faster than prediction – we acknowledge that we may need to look into our implementation further.

Lastly, we conclude with the perspective that within industry, active learning is being explored as a tool to accelerate early-stage discovery in conjunction with medicinal chemistry efforts. Thus, active learning cycles need to not only be fast enough to keep pace with rapidly changing ideas but accurate enough to properly inform experimental follow-up. The challenges we outline in this chapter mostly revolve around developing active learning into a tool that can be integrated in real-time early stage discovery efforts. While the emergence of automated workflows for active learning are big steps towards this goal, integrating these tools into real-world drug discovery workflows has yet to be seen. However, as a standalone tool, active learning has already demonstrated significant value and we are optimistic that the field will continue to progress rapidly.

Chapter 3

Characterizing Discrete Binding Conformations of T4 L99A via Markov State Modeling

As a model system, the binding pocket of the L99A mutant of the T4 bacteriophage has been the subject of numerous computational free energy studies. However, previous studies have failed to fully sample and account for the observed changes in the binding pocket of T4 L99A upon binding of a congeneric ligand series, limiting the accuracy of their results. In this work, we establish definitions for the conformational states of the T4 L99A binding pocket based on the dynamics of the system. We estimate the timescales for the transitions between states and discuss the need to develop enhanced sampling methods to properly account for large changes in protein conformation upon small ligand perturbations.

3.1 Introduction

Reliable candidate discovery is a major bottleneck in the drug development pipeline. Identification and validation of compounds in pre-clinical trials span many years with thousands of potential candidates whittled down to a single approved drug [104]. Modern developments to *in silico* methods seek to remedy the time and cost drain of early stage discovery, providing information on ligand properties without the need for time- and resource-intensive experiments. In particular, relative binding free energy (RBFE) calculations have been widely adopted due to their ability to provide thermodynamic information on potential drug candidates while only requiring simulation data generated from molecular dynamics (MD) [23, 91].

The lysozyme of the T4 bacteriophage serves as an ideal system for binding free energy calculations. A unique leucine to alanine mutation (L99A) creates a solvent-inaccessible and virtually apolar binding pocket with high binding rates for small, organic molecules. The properties of the binding pocket allow simulations of T4 L99A–ligand systems to exclude the movement of bulk water in and out of the binding pocket, decreasing computational cost and increasing the ease of performing RBFE calculations [92, 90]. However, when bound to a congeneric series of alkyl benzenes, the T4 L99A binding pocket adopts three discrete conformations (Figure 3.1), each with increasingly more area open to bulk solvent [87]. The emergence of several discrete binding pocket states upon binding of congeneric ligands has been observed in other systems as well [87], suggesting that many protein binding pockets may undergo significant changes for even small ligand perturbations. This has complicated RBFE calculations, as current methods fail to account for protein conformational changes.

Consequently, attempts at calculating binding free energies for a congeneric series of alkyl benzene ligands bound to T4 L99A have been inaccurate. In a study by Lim et al. [73], the authors found that the accuracy of RBFE calculations could be significantly influenced

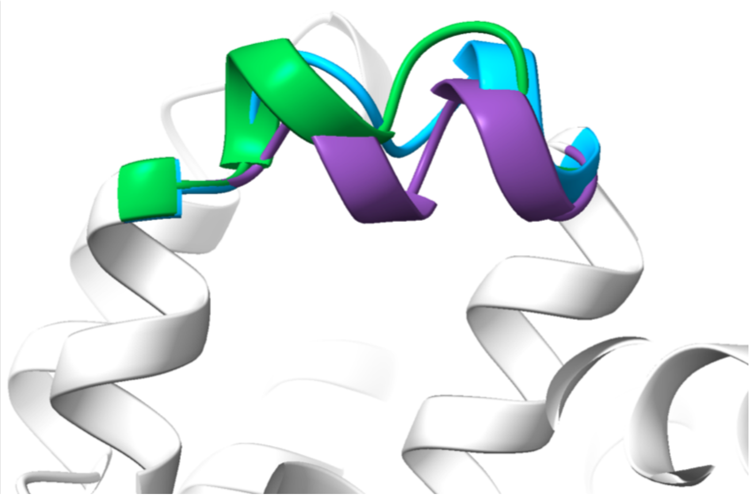


Figure 3.1: Discrete conformations of the T4 L99A F-helix. The F-helix region (residues 107–115) of T4 L99A is reported [87] to adopt three distinct conformations as observed crystallographically: closed, intermediate and open. Here we show three overlays of the F-helix region from the crystal structures we use to represent each state in this work. Shown is the benzene-bound crystal structure (4w52) to represent the experimentally defined closed state (purple), the butylbenzene-bound crystal structure (4w57) to represent the experimentally defined intermediate state (cyan), and the hexylbenzene-bound crystal structure (4w59) to represent the experimentally defined open state (green).

by choice of protein starting conformation. The study demonstrated that improving RBFE calculations may require knowledge of the conformational changes of a system and their relative timescales, which is often difficult to define and sufficiently sample in the course of MD timescales typically used for these calculations [91]. Interconversion between binding pocket states is slow, requiring significant simulation time to capture even a single event. However, insufficient sampling of different protein conformational states leads to significant errors in RBFE calculations [73].

In this study, we capture transitions among the experimentally observed conformational states of the T4 L99A F-helix and make rough estimates for the timescales of these transitions. We begin by defining the discrete conformational states of the T4 L99A F-helix based on the slow dynamics of the system. Using Markov state models (MSMs), we estimate the timescales for transitions between discrete states in MD. Furthermore, we show how defining metastable states of a system from the slowest motions of the system is preferable to defin-

Bound ligand	Starting protein structure		
	closed (4w52)	int (4w57)	open (4w59)
benzene	**4w52–benzene**	4w57–benzene	4w59–benzene
butyl	4w52–butyl	**4w57–butyl**	4w59–butyl
hexyl	4w52–hexyl	4w57–hexyl	**4w59–hexyl**

Table 3.1: Design matrix of different simulated systems. Entries in bold denote systems in which the bound ligand is the native ligand of the corresponding starting crystal structure. Throughout this paper, we refer to the bolded elements as native systems and the others as mixed systems For the sake of brevity, we reference various systems throughout this work in the following manner: [PDB ID]–[ligand name]. This should be interpreted as "[ligand] bound to T4 L99A started from the [PDB ID] structure". As an example, "4w52–benzene" should be read as "benzene bound to T4 L99A started from the 4w52 structure".

ing states based on the RMSD to various crystal structures. Our approach demonstrates how changes in protein conformation for the T4 L99A system are not observed in the MD timescales typical for binding studies, supporting the need to apply Markov state modeling to short, parallel MD simulations of systems to identify potential changes in protein conformation. We conclude by discussing the need to further develop and employ enhanced sampling methods to account for potentially large changes in protein conformation in response to small ligand perturbations.

3.2 Methods

3.2.1 Different simulated systems

We selected different crystal structures in the Protein Data Bank to define the closed, intermediate and open structures for this study. We based our selections on the dominant discrete conformation reported in experiment for each ligand [87]. We arrived at the structure of benzene bound to T4 L99A (PDB ID: 4w52) as our starting structure for the closed state, butylbenzene bound to T4 L99A (PDB ID: 4w57) for the intermediate state and hexylbenzene bound to T4 L99A bound (PDB ID: 4w59) for the open state. We initialized systems for all combinations of these three protein structures (4w52, 4w57 and 4w59) and their three bound ligands (benzene, butylbenzene and hexylbenzene) to generate a total of nine different systems (Table 3.1). We refer to the three systems where the bound ligand is the ligand from the crystal structure as **native systems**. The remaining six systems where the bound ligand is different from the ligand in the crystal structure are referred to as **mixed systems**. We set up simulations for all nine systems based on the procedure outlined in subsection 3.2.2. For the sake of brevity, we reference various systems throughout this work in the following manner: [PDB ID]–[ligand name] (Table 3.1). This should be interpreted as "[ligand] bound to T4 L99A started from the [PDB ID] structure". As an example, "4w52–benzene" should be read as "benzene bound to T4 L99A started from the 4w52 structure".

3.2.2 Preparation and Parametrization of Proteins and Ligands

The T4 lysozyme protein and ligand structures were prepared as input structures for MD simulations. The topology and coordinate input files for GROMACS [9, 1] simulations can be found in the SI. For each protein of interest (4w52, 4w57 and 4w59), we prepared the structure using OpenEye Spruce[97] to add hydrogens and missing loops. Each protein was solvated with TIP3P waters and ions were added to achieve a concentration of 150 mM. The solvated systems were then parameterized with the Amber ff14SB[78] force field.

For each ligand of interest – benzene, butylbenzene, and hexylbenzene – partial charges were assigned using OpenEye's AM1-BCC [97] charge engine and the ligands were parameterized using Open Force Field version 2.0.0[12]. The prepared solvated protein structures and ligands were combined into native and mixed system complexes as described in subsection 3.2.1 resulting in a total of 9 starting structures (Table 3.1). For each protein-ligand complex structure, the crystallographic ligand pose from the native structure was used.

3.2.3 Running molecular dynamics in GROMACS

The protein-ligand complex systems were simulated using GROMACS (v.2021.2). Prior to production, the systems were energy minimized for 1500 steps using steepest descent. The systems were then equilibrated in two steps using a 20 ns NVT ensemble followed by a 5 ns NPT ensemble. Production simulations were run for 100 ns per replicate, with a total of 10 replicates per system. The MDP files for GROMACS simulations can be found in the SI.

3.2.4 Markov state model construction

Markov state models (MSMs) allow for the construction of a transition probability matrix of a system at equilibrium, which is used to obtain the relative populations of conformational states and the transition timescales between them [58]. This can be accomplished by running short MD simulations of a system in replicate, increasing the likelihood that slow transitions are observed compared to a single long trajectory. To build an MSM, relevant features of the system are selected and transformed using time-lagged independent component analysis (TICA) to resolve coordinates corresponding to the slowest motions of the system. From there, a set of metastable states can be identified by clustering along the TICA space. A transition probability matrix can be calculated by estimating the number of transitions between metastable states at some fixed lag time, τ.

For a transition matrix constructed under a specific set of conditions, the eigenvector corresponding to the largest eigenvalue of the transition matrix is equal to one and describes the stationary distribution of the system. Sorting the remaining eigenvalues of the transition matrix in descending order, the corresponding eigenvectors describe the subsequent slowest motions of the system. The timescale of each slowest motion, also known as the **implied timescales**, can be calculated as:

$$t_n = \frac{-\tau}{\ln \lambda_n} \tag{3.1}$$

where is t_n is the n^{th} implied timescale of the system, τ is the MSM lag time and λ_n is the n^{th} largest eigenvalue of the transition matrix. We evaluate how the timescales for different processes in a system change as a function of lag time. Processes with timescales shorter than the lag time cannot be resolved by the MSM. We select a lag time for our system corresponding to the shortest lag time at which the slowest processes of the system stabilize.

We built all MSMs in this study using the PyEMMA (v2.5.12) [113] Python library. We started by choosing features along the F-helix region of the protein (residues 107–115) to characterize the changes in the T4 L99A binding pocket. Using MDTraj (v.1.9.9) [83], we selected the pairwise distance between all $C_\alpha - C_\alpha$ and $C_\beta - C_\beta$ pairs for all residues in the F-helix for a total of 51 distances. We used an implementation of TICA from deeptime (v0.4.4) [56], selecting a TICA lag time of 0.02 ns (10 frames) to transform the set of features into coordinates that described the slowest motions of the system. We then applied k-means clustering on the projection of the trajectory frames onto the two largest TICA components in order to resolve metastable states for each system. We selected a lag time of $\tau = 10$ ns (500 frames) to construct each MSM; this was the shortest lag time beyond which the implied timescales for each system plateaued (Figure A.1).

3.2.5 Calculating RMSD to reference crystal structures

We calculate the root-mean-square deviation (RMSD) between each frame of our trajectories and the closed (PDB ID: 4w52), intermediate (PDB ID: 4w57) and open (PDB ID: 4w59) crystal structures using the `rmsd` function in MDTraj (v.1.9.9). The function centers the trajectory and then calculates the distance between all atoms in the F-helix for each frame and the reference crystal structures. RMSD is implemented in MDTraj using a fast matrix multiplication routine [52] to find the roots of a polynomial equation [122].

3.2.6 Estimating MSM mean first passage time

We estimate timescales between the observed discrete states using the `mfpt` function from the deeptime package [56]. The mean first passage time (MFPT) is defined as the expected time (reported in units of number of simulation frames) to reach one state when starting in another. To calculate the mean first passage time, we provide the function with a transition probability matrix for our metastable states as well as the lag time used to construct the MSM.

3.3 Results

3.3.1 Slowest motion of congeneric ligand systems corresponds to opening of the F-helix

We combine simulation data from all three native systems together in order to ensure the slowest motion of our system corresponds to the opening of the F-helix [42] and/or any important differences between these native systems. For clarity, these three **native systems** are

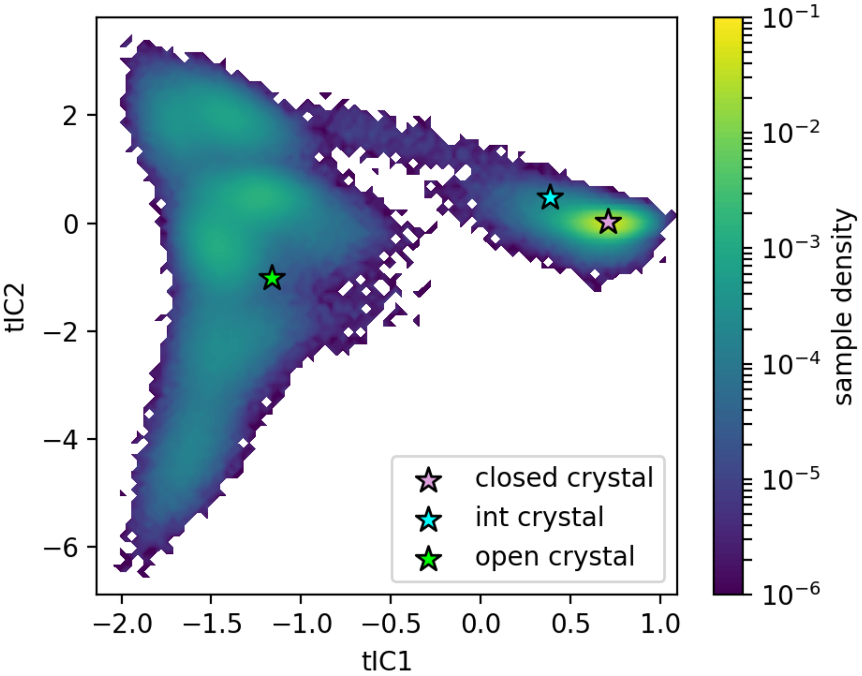


Figure 3.2: Concatenated trajectory data projected onto a 2D TICA landscape. We concatenate six 100 ns parallel trajectories each of benzene, butylbenzene and hexylbenzene bound to their native crystal structures (4w52, 4w57, 4w59, respectively) to form a concatenated trajectory (18 trajectories, total 1.8 μs). In this concatenated trajectory, we select the pairwise distances between all $C_\alpha - C_\alpha$ and $C_\beta - C_\beta$ atoms in the F-helix as features. Using TICA to resolve coordinates along the slowest motions of the system, we show the trajectory projected onto the top two tICs. We map where the crystal structures for 4w52 (purple star), 4w57 (cyan star) and 4w59 (green star) fall on this landscape. The arrangement of the crystal structures suggest that traversing along the first TICA component captures the discrete changes in the F-helix.

benzene bound to T4 L99A started from the 4w52 structure (4w52–benzene), butylbenzene bound to T4 L99A started from the 4w57 structure (4w57–butylbenzene) and hexylbenzene bound to T4 L99A started from the 4w59 structure (4w59–hexylbenzene) (see Section 3.2.1 for more explanation on terminology). Specifically, we expect these native systems to occupy three distinct states with no or few transitions between them. Thus, a concatenated trajectory of these native systems would show distinct transitions between states at the end of each trajectory, turning these state transitions into easily recognizable slow motions. This allows us to resolve a set of time-lagged independent components (tICs) that can be used to categorize the frames of each trajectory into discrete states. To this end, we concatenate six, 100 ns parallel trajectories each of benzene, butylbenzene and hexylbenzene bound to their native crystal structures to generate a concatenated trajectory totaling 1.8 μs. In this concatenated trajectory, we select all the $C_\alpha - C_\alpha$ and $C_\beta - C_\beta$ pairwise distances within the protein F-helix (residues 107–115) and apply TICA to resolve a set of tICs. We map all frames of the concatenated trajectory onto the top two tICs, thus projecting the data onto a two-dimensional (2D) TICA surface. We find that our 2D TICA surface consists of two primary regions of density – a larger half-moon region on the left and a smaller oval-shaped well on the right – with a bridge region connecting them (Figure 3.2).

We find that moving along the first tIC generated by the concatenated trajectory corresponds to transitioning between the closed, intermediate and open crystal structures (Figure 3.2). To better understand how regions of the TICA space correspond to different F-helix conformations, we project the three native crystal structures (4w52, 4w57 and 4w59) onto our 2D TICA surface. For each native crystal structure, we calculate the relevant $C_\alpha - C_\alpha$ and $C_\beta - C_\beta$ pairwise distances and use those features to calculate coordinates in TICA space. We find that the open crystal structure (4w59) is roughly centered in the larger half-moon region on the left of the TICA space, the intermediate crystal structure (4w57) is right of the bridge region between wells and the closed crystal structure (4w52) is centered in the oval-shaped well on the right (Figure 3.2). Consistent with what was reported in experiment [87], we find from our TICA projections that the intermediate crystal structure falls between the closed and open crystal structures. Based on the position of each crystal structure in the 2D TICA surface, we determine that the differences among discrete states is predominately captured by variation in the first independent component (Figure 3.2).

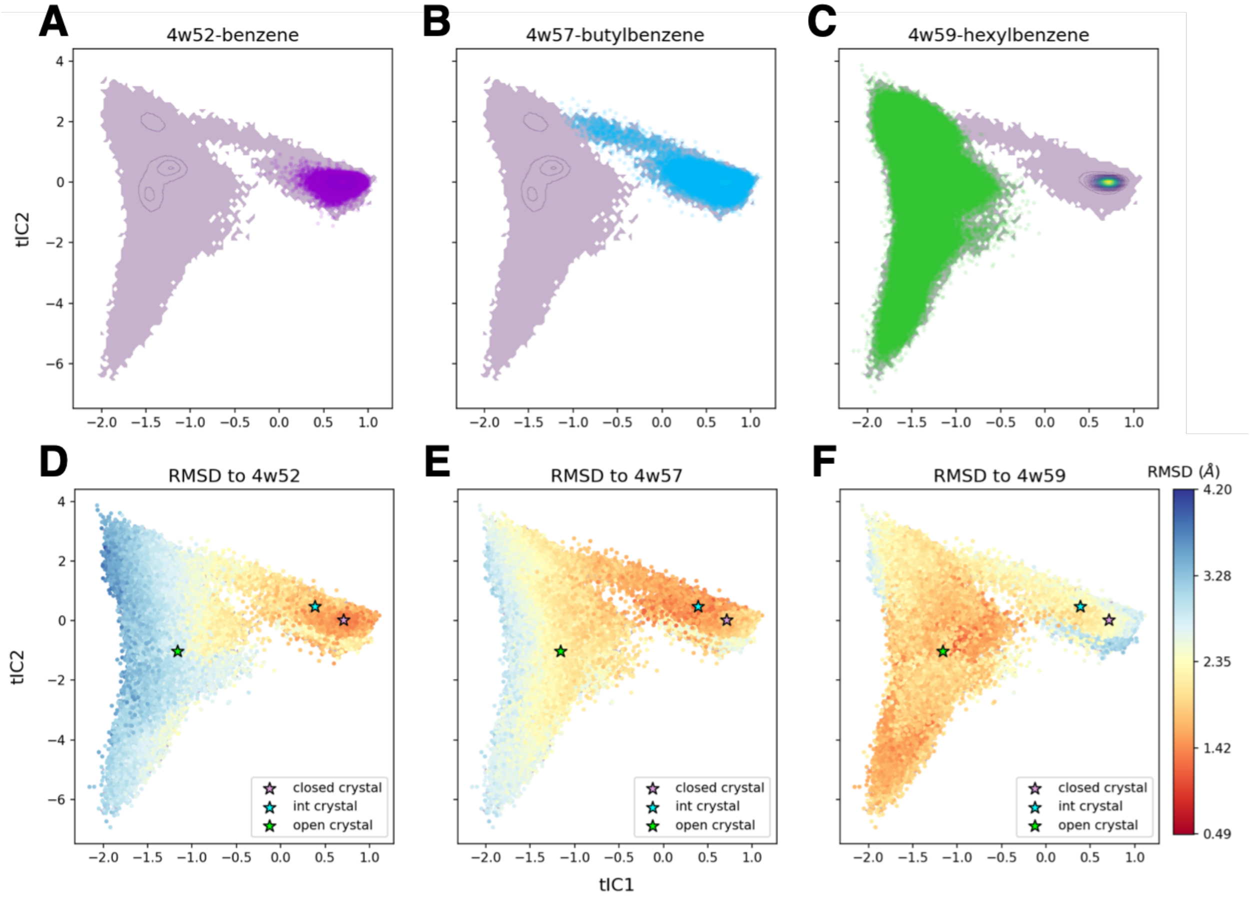


Figure 3.3: RMSD of trajectory frames in 2D TICA space relative to experimental crystal structures. (A–C) On top of the density of the concatenated trajectory (light purple), we project the frames from simulations of the (A) 4w52–benzene (purple dots), (B) 4w57–butylbenzene (cyan dots) and (C) 4w59–hexylbenzene (green dots) systems in 2D TICA space. (D–F) For each frame of the concatenated trajectory, we calculate the RMSD of the F-helix to different crystal structures. Shown is RMSD to (D) the benzene-bound crystal structure (PDB ID: 4w52), (E) the butylbenzene-bound crystal structure (PDB ID: 4w57) and (F) the hexylbenzene-bound crystal structure (PDB ID: 4w59). The heatmap signifies the RMSD of each frame of the concatenated trajectory to the specified reference crystal structure. We additionally map where the crystal structures for 4w52 (purple star), 4w57 (cyan star) and 4w59 (green star) fall on this landscape for reference. Frames closer to each respective crystal structure in TICA space also tend to have lower RMSD to that structure.

3.3.2 Time-lagged independent component analysis provides an alternative to RMSD for defining discrete states

We find that our native system trajectories generally occupy distinct regions of the TICA space. Using the tICs we previously define, we map each native system trajectory onto the 2D TICA surface. We observe that all frames of simulations of the 4w52–benzene system map to the right side of the oval-shaped well and center around the 4w52 crystal structure (Figure 3.3A). Simulations of the 4w57–butylbenzene system sample all regions of the oval-shaped well in addition to the bridge region linking the two wells together (Figure 3.3B). Lastly, simulations of the 4w59–hexylbenzene system exclusively sample the half-moon region on the left side of the TICA plot (Figure 3.3C). By contrasting the different regions of TICA space these systems occupy, we posit that the experimentally observed closed conformation is represented by the right oval-shaped well, the experimentally observed open conformation is the left half-moon well and the experimentally observed intermediate conformation is the bridge region between them in the 2D TICA space (Figure 3.3A–C).

We find that using the RMSD to reference crystal structures to define states shows some agreement with our TICA-based approach but ultimately results in different state categorizations. For each native system trajectory, we calculate the RMSD of each trajectory frame to one of three reference crystal structures and plot onto our 2D TICA surface, color-coding by RMSD (see Methods). We observe that generally, frames closer to each crystal structure in TICA space also tend to have lower RMSD to that crystal. However, we also observe that state definitions based on RMSD are quite fuzzy. We find that the majority of trajectory frames with low RMSD to the 4w52 crystal structure fall in the right hand region of the oval-shaped well. However, we also observe a number of frames right under the 4w52 in TICA space with relatively high RMSD to the structure. (Figure 3.3D). Frames with low RMSD to the 4w57 crystal structure generally fall in the bridge region, but we also observe a number of frames in the closed and open regions within a given RMSD of the 4w57 structure

(Figure 3.3E). Lastly, we find that frames with low RMSD to the 4w59 crystal mostly fall in the half-moon shaped well in the left region of the plot. However, we find frames close to the 4w57 structure with relatively low RMSD to the 4w59 crystal as well as frames in the upper left corner of the half-moon well with just as high of RMSD to the 4w59 structure as frames close to the 4w52 crystal (Figure 3.3F).

3.3.3 Choice of protein starting structure affects sampling of protein conformational states

We proceed to construct Markov state models (MSMs) for each native system to resolve metastable states based on slow dynamics. As described in the Methods, we project all native system simulation frames onto the top two tICs resolved using the concatenated trajectory. We then sample along the TICA space via *k*-means clustering to identify metastable states. By analyzing the eigenvectors of the estimated transition probability matrix – corresponding to the timescales of the slowest motions of the system – we identify macrostates for each system.

We observe that the 4w52–benzene system undergoes no kinetically meaningful transitions in the elapsed simulation time frame. We find that the slowest motions of the system decay faster than the MSM lag time and correspond to random fluctuations (Figure A.1). Based on these findings, we determine there is only a single state sampled in the simulations of the 4w52–benzene system (Figure 3.4A), which is consistent with the region of TICA space we identify as the closed region (Figure 3.3A).

In simulations of the 4w57–butylbenzene system, we find the slowest motion to be a transition resulting in two states within the oval-shaped well of the 2D TICA surface (Figure 3.4B). While we are not clear on what this transition is, it occurs on a much slower timescale than other processes and merits investigation (Figure A.1). Interestingly, the *third* slowest motion

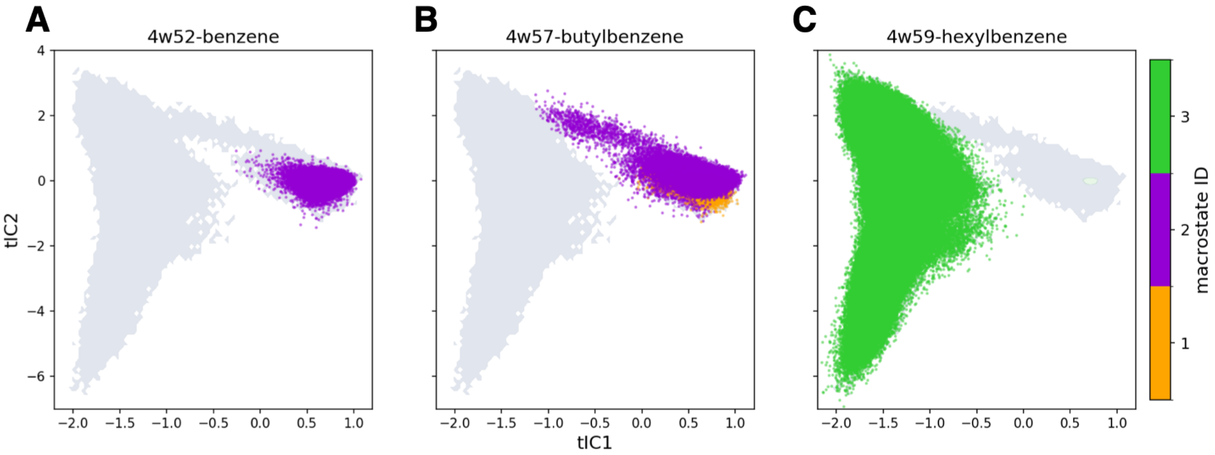


Figure 3.4: MSM-resolved states for each native system. We construct MSMs to resolve discrete states based on the slowest processes in each of the three native systems. On top of the density of the concatenated trajectory (gray), we project the frames of simulations of the (A) 4w52–benzene, (B) 4w57–butylbenzene and (C) 4w59–hexylbenzene systems. We color frames based on their MSM-assigned macrostates, keeping macrostate definitions consistent across the different native systems.In their elapsed simulation time (1 μs), native systems are mostly confined to their starting protein conformational states.

of the 4w57–butylbenzene system corresponds to changes in density between the bridge and oval-shaped regions of the TICA space (Figure A.3B) – which we believe correspond to the closed and intermediate states, respectively (Figure 3.3B). However, this process decays faster than the MSM lag time and thus we do not consider the intermediate state to be a discrete state based on this analysis (Figure A.1).

Lastly, we determine there is only a single state sampled in simulations of the 4w59–hexylbenzene system (Figure 3.4C). While there are motions in the system occurring on slow timescales (Figure A.1), they correspond to fluctuations within the half-moon shaped well, and are primarily described by differences in the second tIC (Figure A.4C). Given we primarily attribute the F-helix opening motion to the first tIC (Figure 3.2), we do not consider these to be states relevant to our analysis.

3.3.4 Simulation of mixed protein-ligand systems allows for estimation of discrete state transition timescales

In attempts to observe transitions within our simulation timescales, we also simulate and build MSMs for a series of *mixed* systems. These are simulations begun from a crystallographic state belonging with a different ligand than the one they are simulated with, so we expect these simulations to transition to a different state while they are run, even if only once (see Section 3.2.1 for more details). We find that simulations of both the 4w57–benzene (Figure 3.5A) and 4w59–benzene (Figure 3.5B) systems primarily sample the region of TICA space we believe corresponds to the closed state. For the 4w57–benzene system, we observe a single slow motion (Figure A.2B) corresponding to an exchange between two regions of the oval-shaped well (Figure 3.5A), which appears to be a similar motion as the one we observe in the 4w57–butylbenzene system (Figure 3.4B).

In mixed systems for the butylbenzene ligand, we find that there are no meaningful slow motions in the 4w52–butylbenzene system (Figure A.3A) and only identify a single state for that system (Figure 3.5C). We also find that none of the frames in the 4w52–butylbenzene trajectories sample the bridge region of the TICA space, which we previously hypothesized was the intermediate state (Figure 3.3B). However, we observe sampling of the bridge region in the 4w59–butylbenzene system. In this system, the slowest motion corresponds to an exchange between the bridge and oval-shaped regions of the TICA space (Figure A.3C). We subsequently identify two states for the 4w59–butylbenzene system (Figure 3.5D).

Lastly, we find multiple states for both mixed systems containing the hexylbenzene ligand. Simulations of hexylbenzene interconvert between the closed and open states when begun from the 4w52 structure (Figure 3.5E). Interestingly, the slowest motion of this 4w52–hexylbenzene system is an exchange between those two discrete states, but we do not observe any sampling of the intermediate state (Figure A.4A). When begun from the 4w57 structure,

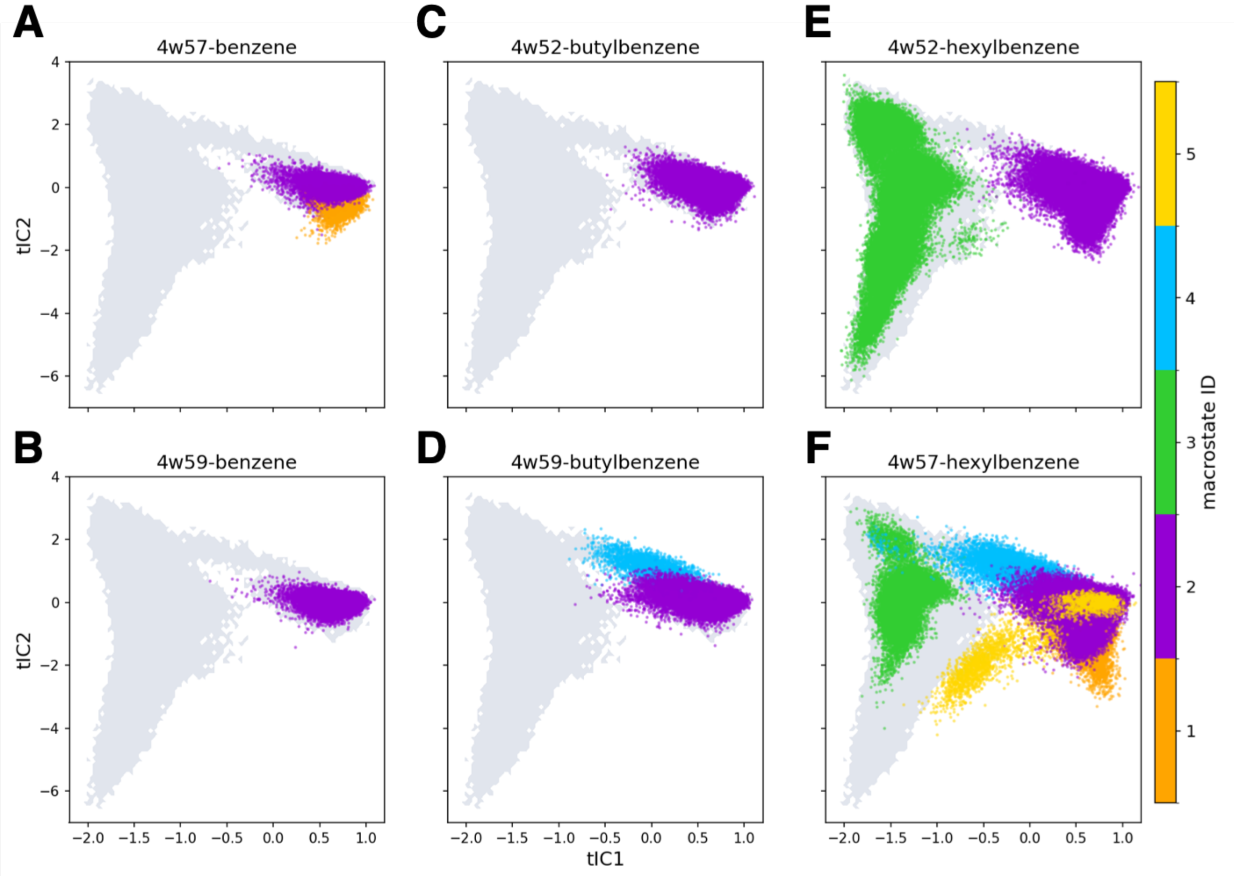
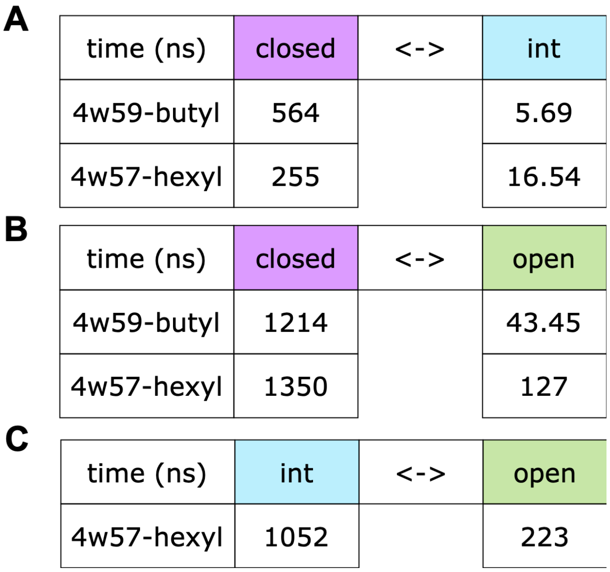


Figure 3.5: MSM-resolved states for each mixed system. We initialize benzene, butylbenzene and hexylbenzene ligands each in their respective non-native crystal structures to observe potential changes in the F-helix. On top of the density of the concatenated trajectory (gray), we project the frames corresponding to simulations of the (A) 4w57–benzene, (B) 4w59–benzene, (C) 4w52–butylbenzene, (D) 4w59–butylbenzene, (E) 4w52–hexylbenzene and (F) 4w57–hexylbenzene systems. We color frames based on their MSM-assigned macrostates, keeping macrostate definitions consistent across the different native systems. On these timescales, simulations of benzene in T4 L99A primarily stay in the closed conformation when simulated beginning from either the (A) 4w57 or (B) 4w59 structures. Simulations of butylbenzene only sample the closed state when begun from the (C) 4w52 structure but undergo interconversion with the intermediate state when begun from the (D) 4w59 structure. Simulations of hexylbenzene interconvert between the closed and open states without sampling the intermediate state when begun from the (E) 4w52 structure. When begun from the (F) 4w57 structure, simulations of hexylbenzene sample the closed, intermediate and open states. We also observe an additional transition pathway (yellow) between closed and open states not found in any other systems.



A

time (ns)	closed	<->	int
4w59-butyl	564		5.69
4w57-hexyl	255		16.54

B

time (ns)	closed	<->	open
4w59-butyl	1214		43.45
4w57-hexyl	1350		127

C

time (ns)	int	<->	open
4w57-hexyl	1052		223

Figure 3.6: Estimates of timescales for discrete state transitions. Using our constructed MSMs, we estimate the transition timescales between states. Each row corresponds to the transition estimates for that specific mixed system. Columns denote the starting state of the transition. The entry in each cell is the estimated transition time from the state in that given column to the state in the other column. We report estimates in units of ns for transitions betwen (A) closed and intermediate, (B) closed and open and (C) intermediate and open.

simulations of hexylbenzene sample five distinct states (Figure 3.5E). Four of the five states are observed in other systems, corresponding to the closed, intermediate and open states as well as the state observed in both the 4w57–butylbenzene (Figure 3.4B) and 4w57–benzene (Figure 3.5A) systems. The final discrete state falls between the closed and open states, similar to the intermediate state. However, it falls lower along the second tIC and is distinct from the previously established intermediate state.

For each system with multiple MSM-identified states, we estimate the timescales of the transitions between them by discretizing our trajectories to calculate a mean-first passage time (see Methods). This is highly approximate, especially given that in many cases we do not have multiple transitions between states and/or transitions may be unidirectional. However, these are useful in providing at least order of magnitude estimates of the relevant timescales. Our estimates of timescales further support the idea that the intermediate state is a relatively short-lived state. In both the 4w59–butylbenzene system (Figure 3.5D) and

4w57–hexylbenzene system (Figure 3.5F) where we observe transitions between the closed and intermediate states, we estimate that the process of going from closed to intermediate is over an order of magnitude slower than going from intermediate to closed (Figure 3.6A). Similarly, for the 4w57–hexylbenzene system (Figure 3.5E) where we observe transitions between the intermediate and open states, we estimate that the process of going from open to intermediate is also around an order of magnitude slower than going from intermediate to open (Figure 3.6C).

We also find further evidence supporting the claim that the open state is a more structurally dissimilar state compared to the closed and intermediate states. From the 4w52–hexylbenzene system (Figure 3.5E) and 4w57–hexylbenzene system (Figure 3.5F), we estimate that the transitions between the closed and open states (Figure 3.6B) are several times slower than the transitions between either the closed and intermediate (Figure 3.6A) or the intermediate and open states (Figure 3.6C).

3.4 Discussion

In this study, we resolve the discrete states of the T4 L99A binding pocket observed in experiment [87] in MD. By selecting a combination of features which capture structural elements of the F-helix, we design a trajectory where the slowest motion of the system corresponds to the opening of the T4 L99A binding pocket. In doing so, we are able map a set of continuous trajectories onto a two-dimensional TICA surface. We find based on the relative positions of the crystal structures in TICA space that the closed and intermediate structures are much closer in distance to each other than to the open state, suggesting greater structural similarity between those two states to each other than to the open conformation.

3.4.1 Evidence of the intermediate state in MD timescales

We find evidence that the intermediate state is a kinetically distinct but short-lived state in our MD simulations. When evaluating the native 4w57–butylbenzene system, we find that the trajectory samples the bridge region of the TICA surface which we believe corresponds to the intermediate state (Figure 3.4B). However, this bridge region is not identified as a discrete state by the MSM built from simulations of the native 4w57–butylbenzene system (Figure 3.4B) but is by the MSM built from simulations of the 4w59–butylbenzene system (Figure 3.5D). This suggests that while the intermediate state may be observed in our total MD simulation time of 1 μs, transitions are not significant. However, transitions do become significant when we attempt to force the transition in simulations of the 4w59–butylbenzene mixed system. Our estimates of the timescales to and from the intermediate state additionally corroborate that the intermediate state is short-lived. We find that moving to the intermediate from either the closed or open states is about an order of magnitude slower than leaving the intermediate to either of those states (Figure 3.6A, C).

3.4.2 TICA-based state definitions may be preferable to RMSD-based definitions

We believe using TICA and slow motions to define states may be more robust than RMSD. In particular, we find that while there is agreement between our TICA-based state definitions and the RMSD to reference crystal structures (Figure 3.3D–F), definitions based on the dynamics of the system result in clearer boundaries between states. For example, we observe that trajectory frames close to the 4w59 crystal structure in TICA space generally have low RMSD to that structure (Figure 3.3F). However, some of these same frames also have low RMSD to the 4w57 crystal (Figure 3.3E). Given the 4w57 and 4w59 structures are considered distinct in experiment, considering a structure to be similar to both states is confusing.

Additionally, as a distance metric, RMSD does not account for the potential size of states. As an example, we observe that frames in the top left corner of the half-moon shape well and frames in the right side of the oval-shaped well are similarly distant to the 4w59 structure based on RMSD (Figure 3.3F). However, we instead find with our kinetics-based definition that the entirety of the half-moon shaped well describes the open state (Figure 3.4C). While the top and bottom portions of the half-moon shaped well can be further separated into kinetically distinct states (Figure SI), this separation is entirely along the second tIC, which does not primarily correspond to opening of the F-helix.

One implication of this finding is that previous work looking at the transitions between states for this system may have used incorrect state definitions and potentially miscounted transitions [73]. It may be worth revisiting this work and applying these TICA-based state definitions to evaluate how much enhanced sampling methods are able to accelerate transitions.

Lastly, we believe our approach of using slow system dynamics to define individual states could be extended to other systems where there are known discrete conformational changes in response to a congeneric series of ligands. Examples of known systems with such behavior include the heat shock protein 90 (HSP90) and the human estrogen receptor alpha (ERα) [87]. This approach could especially be useful in systems where there may not necessarily be reference crystal structures for each discrete state.

3.4.3 Implications for future binding studies

One caveat of this study is that the procedure presented may not be suitable for many biological systems. For the T4 L99A system, we have access to crystal structures for all discrete states, allowing us to effectively reverse engineer a trajectory in which the slowest motion of the system corresponds to the opening of the F-helix observed in experiment. In many cases, it may not be known what prompts conformational changes or whether the system undergoes any conformational change at all. Furthermore, we find in this study that even when running MD for timescales much longer than those run for typical binding studies, our native system trajectories do not sample any protein conformational changes. Thus, simply running longer simulations may not resolve issues with sampling.

Binding studies, including in L99A, are typically done on much shorter timescales than those studied here. Thus, it is vital for researchers running such studies to have an idea of the relevant timescales for protein conformational sampling, and ensure that (a) they are using the appropriate protein structure(s) for their ligand, or (b) they are somehow enhancing sampling of protein conformational transitions, or (c) they diagnose the quality of protein sampling. Given that it is relatively common for structure-based design studies to uncover multiple distinct protein conformations upon binding of different ligands, it may be possible to anticipate some of these conformation changes [87, 89]. However, timescales for transitions between relevant protein conformations might be slow and uncharacterized, in which case researchers need to be cautious. Thus, we believe it may be necessary to further develop and employ enhanced sampling methods for dealing with protein conformational changes.

Lastly, we make all files necessary to reproduce our trajectories freely available. We also include the code used for analysis so that researchers can access our state analyses. Additionally, one unexplored avenue of this work is investigation into the structures classified into the various discrete states. We include snapshots of structures along various points in the TICA surface should researchers want to analyze this data.

3.5 Conclusion

Overall, we have demonstrated how Markov state modeling can be combined with short, parallel MD trajectories in order to define discrete states based on the slow dynamics of a system. We verify that the discrete states of the T4 L99A binding pocket previously established in experiment [87] can also be observed in MD, although even longer MD simulations of native systems do not sample any conformational changes. Thus, we simulate a series of mixed systems in order to estimate the transition timescales between states. Our timescale estimates are consistent with previous findings, where it was shown that for the T4 L99A system, the choice of starting protein conformational state heavily influences which conformations are sampled through the course of the simulation [73]. For known systems undergoing discrete conformational changes upon binding of a congeneric ligand series, we suggest combining short, replicate MD simulations of different bound ligands with MSMs to resolve biologically-relevant slow motions of the system. We believe the coordinates resolved from kinetic information offers an alternative to RMSD-based definitions and can be used to establish more robust state definitions for other biological systems.

Chapter 4

Building Block-Based Binding Predictions for DNA-Encoded Libraries

DNA-encoded libraries (DELs) provide the means to make and screen millions of diverse compounds against a target of interest in a single experiment. However, despite producing large volumes of binding data at a relatively low cost, the DEL selection process is susceptible to noise, necessitating computational follow-up to increase signal-to-noise ratios. In this work, we present a set of informatics tools to employ data from prior DEL screen(s) to gain information about which building blocks are most likely to be productive when designing new DELs for the same target. We demonstrate that similar building blocks have similar probabilities of forming compounds that bind. We then build a model from the inference that the combined behavior of individual building blocks is predictive of whether an overall compound binds. We illustrate our approach on a set of three-cycle OpenDEL libraries screened against soluble epoxide hydrolase (sEH) and report a performance of more than an order of magnitude greater than random guessing on a holdout set, demonstrating that our model can serve as a baseline for comparison against other machine learning models on DEL data. Lastly, we provide a discussion for how we believe this informatics workflow could be applied to benefit researchers in their specific DEL campaigns.

4.1 Introduction

Drug discovery campaigns have increasingly adopted DNA-encoded libraries (DELs) in recent years because they allow for relatively cheap and rapid exploration of diverse areas of chemical space [45, 39, 5, 7, 32, 137]. In DELs, a concept first introduced by Brenner and Lerner [13], scientists sequentially couple small molecules known as building blocks via split-and-pool combinatorial synthesis. The process tags each building block with a unique DNA oligomer such that each final library member will be covalently attached to a record of its synthesis in the form of a sequenceable DNA barcode. Researchers then incubate the entire library with a target of interest and wash away any compounds that do not bind. Finally, experimentalists amplify and sequence the DNA barcodes of the observed binders and further investigate any compounds with detected DNA read counts as potential binders [20, 79]. Typically, DELs incorporate two to four cycles of encoding and chemistry, which can achieve a diversity of up to billions of unique compounds [112, 40, 31].

Given the large combinatorial scale of DELs, selection data can be quite noisy due to issues such as variable reaction yields and formation of truncates [69, 111, 10], as well as errors within experimental procedures and noise during DNA sequencing [66, 44]. These sources of noise have made it common to analyze selection data with computational models to prevent wasting time and resources re-synthesizing and evaluating unproductive candidates. Recent work suggests how machine learning approaches can denoise DEL data [69, 72, 10] and identify promising candidates in out-of-sample data [82]. Computational models likely will yield even further insights as they are applied to DEL selection data [116].

In this paper, we introduce a method for analyzing DEL selection data at the building block-level, with the goal of gaining insights we can use to design better DELs for subsequent screening rounds. First, we introduce an interpretable analysis of individual building blocks. Second, we quantify how building blocks interact with each other to affect whether a compound binds to the target of interest. Third, leveraging the idea that similar compounds have similar properties [59], we demonstrate how we can use similarity scoring methods to predict the productivity of new building blocks and how similarity metrics differ in their ability to do so. Finally, we build a model which combines the behavior of building blocks at each position into a statistical prediction on the probability of an untested molecule binding to the target of interest.

We note that all the results in this paper come from a pooled set of three-cycle OpenDEL libraries from HitGen screened against a single target, soluble epoxide hydrolase (sEH). We release all the data we analyzed in this study, so that interested researchers are able to reproduce our findings. We emphasize that while the findings presented here are specific to this set of DELs on sEH, we believe our informatics workflow can be extended to analyze the results of various DEL campaigns.

4.2 Results and Discussion

We begin by defining the idea of **productivity** for individual building blocks, which we use to assess whether an overall compound binds to a target. We demonstrate how quantifying the productivity of individual building blocks can provide general insights into structures that could contribute to binding for a target of interest. We then develop a method to guide subsequent DEL screens on a target by (1) identifying productive candidates from a list of proposed building blocks and (2) predicting whether compounds containing those identified building blocks bind to the target of interest. We demonstrate this concept by splitting our data into training and holdout sets (where the holdout sets contain building blocks not seen in training) and provide a workflow for how to incorporate this method in a practical setting.

4.2.1 A building block metric, P(bind), identifies the most productive building blocks at each position

This section introduces a metric we call P(bind) to quantify the productivity of building blocks from a set of DEL selection data.

Notation for building block positions

To aid in interpretation, we establish a bit of notation. For building block positions, we refer to the position closest to the DNA tag as p_1, the middle position as p_2 and the position furthest from the DNA tag as p_3 (Figure 4.1). Each of these building block positions is called a **monosynthon**. We denote the set of all building blocks for a given position as BB_i, where $1 \leq i \leq 3$ in this study. Individual building blocks are denoted as bb_x, where x is the identifier, ID, assigned to each unique building block. We refer to two building block positions considered jointly, also known as a **disynthon**, using the notation BB_iBB_j. In our definition, the two positions considered for a disynthon do not need to be adjacent. Finally, we denote **trisynthons** as $BB_1BB_2BB_3$. To specify a subset, we use a vertical bar from set-building notation [119] where subset conditions are to the right of the bar. For example, $\{BB_1BB_2 \mid BB_1 = bb_1\}$ represents the set of disynthons where position one contains the building block with ID 1.

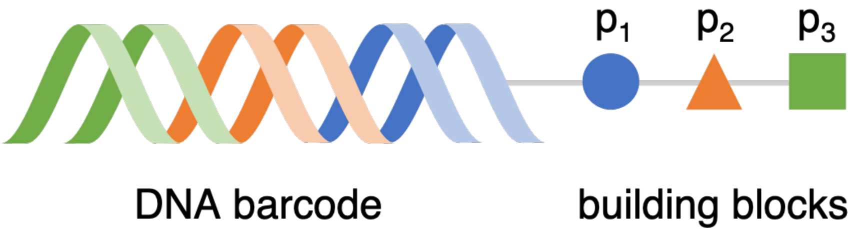


Figure 4.1: A schematic of DEL library members. All DEL library members in this study are composed of three small molecule building blocks and referred to as trisynthons. The first added building block is closest to the DNA (position 1) and the last added is furthest (position 3). Each building block has a corresponding DNA tag encoding its identity, shown in this figure via color coordination. The combined DNA tags form a unique barcode, which is amplified and sequenced in experiment to verify the presence of the trisynthon. Pictured is position 1 (blue), position 2 (orange) and position 3 (green), which we refer to as p_1, p_2 and p_3, respectively.

Each position in the library contains a small number of highly productive building blocks

Firstly, to compare building blocks quantitatively, we require a metric to characterize a desirable versus undesirable building block. We define the productivity of a building block, **P(bind)**, as the fraction of compounds that bind to the target when a given building block occurs in a particular position. In this study, we defined binders as compounds with a read count statistically different from 0 at a 95% confidence threshold, making the assumption that read counts follow a Poisson distribution [66] (see Methods for more details).

To illustrate how we calculate P(bind), we provide the following example. Let S be the subset of trisynthons such that position p_1 contains the arbitrary building block bb_x. This would be expressed as:

$$S = \{BB_1BB_2BB_3 \mid BB_1 = bb_x\} \tag{4.1}$$

If the number of trisynthons in the subset S is N, the P(bind) of the building block bb_x is

$$P(bind) = \frac{\sum_{k=1}^{N} I_k}{N} \tag{4.2}$$

where I_k is 1 if the k^{th} compound in S binds to the target and 0 otherwise, as defined in Section 4.4.2. We repeat this calculation by changing the subset represented in eq 4.1 for each building block in each position of the library. Comparing building blocks by their P(bind) values then allows us to identify the most productive building blocks for each position.

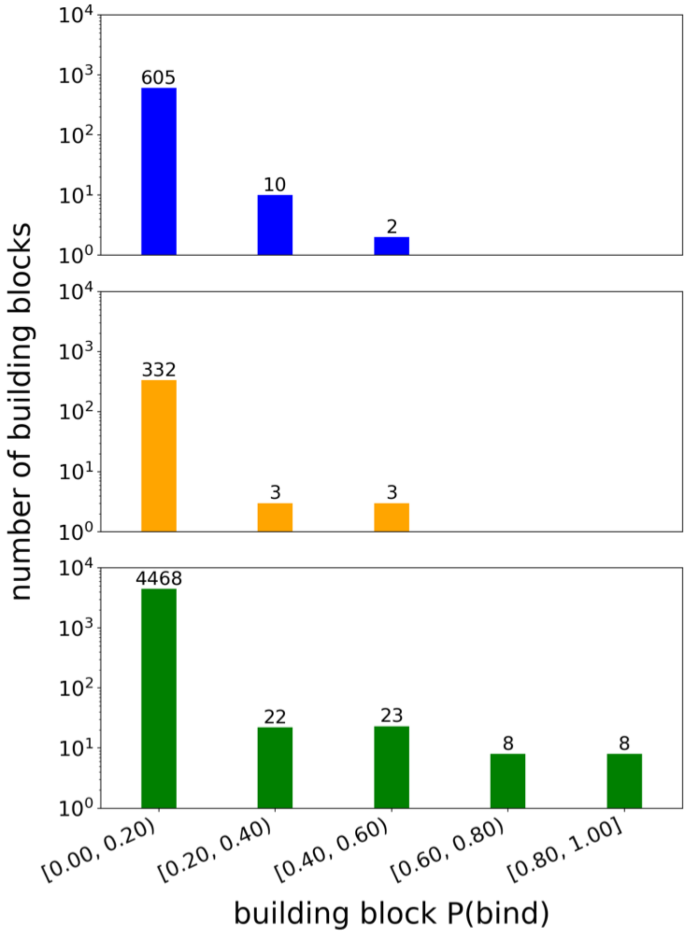


Figure 4.2: Distributions of P(bind) values for each building block position. Building blocks at each position are separated into P(bind) bins with the value above each bar indicating the number of building blocks contained in each interval. Shown are the distributions of P(bind) values for building blocks in p_1 (blue, top), p_2 (orange, middle) and p_3 (green, bottom).

We identify a small fraction of building blocks in each position with P(bind) values significantly higher than average. Splitting building blocks into intervals based on their P(bind) values, we find that the distribution of P(bind) at every position is heavily right-skewed, with more than 95% of building blocks at every position having P(bind) values less than 0.20 (Figure 4.2). For positions 1 and 2, the top 1% of P(bind) values are contained in the P(bind) interval, $[0.40, 0.60)$, whereas for position 3, the top 1% of P(bind) values extends across the P(bind) interval, $[0.80, 1.00]$. Given the mean P(bind) value of all building blocks at each position is on the order of 10^{-2}, this tells us that the top building blocks occur in binders at a rate about 50 times higher than average.

In this analysis, the difference in the maximum P(bind) value between one position and another reveals that trisynthons are more sensitive to the building block present in certain positions. We posit that in this DEL where trisynthons are synthesized linearly (Figure 4.1), position 3, being the furthest from the DNA barcode and therefore the most exposed, has the greatest effect on whether the compound binds. Position 1 has the smallest effect on overall compound binding as the position closest to the DNA tag. Since DELs are typically screened with DNA tags still attached, we believe the presence of the DNA tag may partially obstruct interactions with the target. We note that our observation of the importance of position 3 could also be confounded by a larger and more diverse selection of building blocks in that position. However, with the exception of 10 building blocks in position 3, every building block in each of the three positions is used in a statistically significant number of compounds ($N > 30$) [68] (Figure S1). This suggests our calculation of the P(bind) metric for each building block should not be highly impacted by small sample sizes. Thus, we believe that when we observe building blocks with high P(bind) values, these values indicate the building blocks are truly productive rather than having values which appear high as an artifact of sampling bias.

It is certainly possible that our finding that building block productivity varies based on library position also points to an issue with false negatives in DELs. Due to the high throughput nature of DEL screens, it has been demonstrated that larger library sizes lead to high false negative rates [112]. However, we believe that because the P(bind) metric is aggregated across all the compounds the BB occurs in, the metric should be more robust to false negatives. Moreover, we observe some alignment between building blocks we find to be most productive in position 3 and structural motifs of sEH inhibitors in the literature. Notably, the top two most productive building blocks in position 3 resemble benzhydryl pharmacophores that have been reported in the literature to form favorable pi-stacking interactions with residues in the binding pocket of sEH (Figure S2) [34].

When comparing various physicochemical properties of more and less productive building blocks at each position, we find some commonalities. For example, the most productive building blocks in all positions have higher calculated logP. The most productive building blocks in positions 1 and 2 are also characterized by fewer hydrogen bond donors, whereas the most productive building blocks in position 3 have fewer hydrogen bond acceptors and more hydrogen bond donors than their less productive counterparts (Figure S3). We note that our method may be able to broadly detect target-specific architectures that are favored for binding (as in this case with sEH) based on the differences in productivity for the building blocks in different positions.

Building block productivity increases the variety of binding disynthon pairs

Having identified productive building blocks at each position, we proceed to investigate what characterizes a building block with high P(bind) value chemically. To do so, we analyze how building blocks combine at a disynthon (pairwise) level. We hypothesize that building blocks with high P(bind) values are **compatible** with a greater number of other building blocks. Here, we define two building blocks as compatible if they co-occur in a compound that binds to sEH.

To test our hypothesis, we evaluate how the number of compatible partners for a building block varies with the P(bind) value of the building block. To calculate the number of compatible partners, we first identify all compounds that bind when a building block is in a certain position. We then count how many unique building blocks are in the other two positions from this list of binders. The number of compatible building blocks in position p_j for an arbitrary building block x in position p_i can then be expressed as

$$N_{ij} = |\{BB_iBB_j \mid BB_i = bb_x\}| \tag{4.3}$$

where the vertical bars on each side of the subset is the cardinality or number of elements in the subset [119].

We find that high P(bind) building blocks form binders with a broader range of partners in both positions. We observe a monotonically increasing relationship between P(bind) and number of compatible partners for all building block pairs (Figure 4.3). P(bind) tells us how successful a building block is when it is placed in a certain position, but tells us nothing about the behavior at other positions that may lead to the success (or lack thereof) at one position. Hypothetically, a building block could have a high P(bind) value but only form binders with a very limited selection of partners in one of the other positions. To illustrate this possibility, imagine a scenario where all binders that contain bb_x in p_3 only occur if one or a few specific building blocks are present in p_2. This would attribute all the variation between these compounds to the identity of the building block in p_1. Since we find the least sensitivity to molecule binding in p_1 (Figure 4.2), this is a realistic hypothesis to rule out.

On the contrary, we see that building blocks which are successful in one position are compatible with a broader diversity of building blocks in all other positions. We note that this could partially be attributed to variations in coupling reactions present in our library, which we address later in our discussion. Previous work has shown that in DELs, reactions are

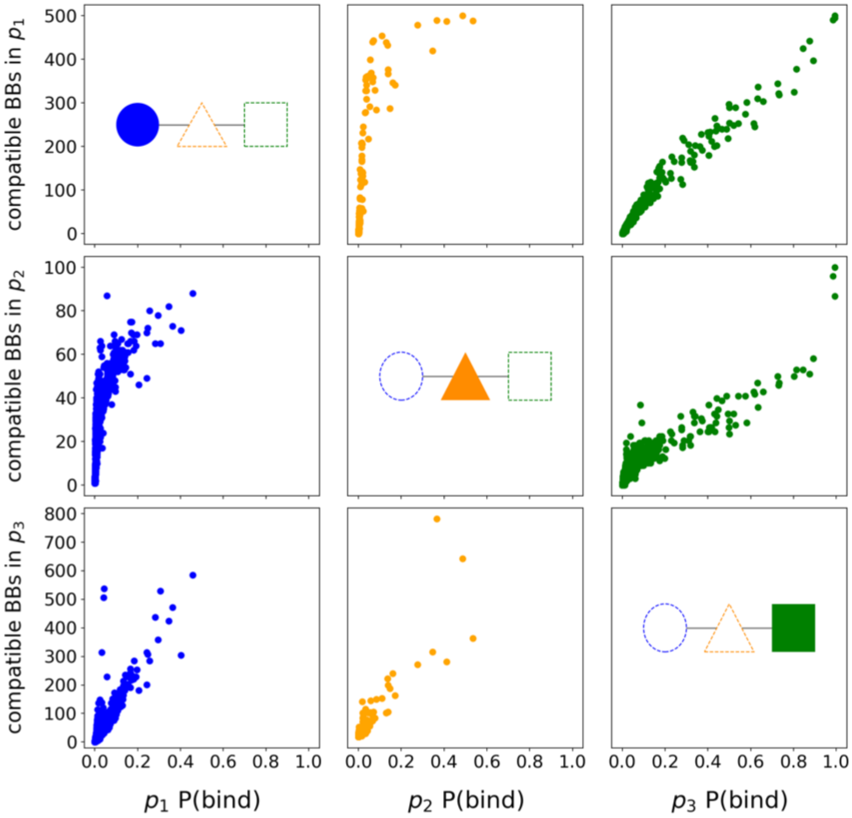


Figure 4.3: Number of compatible partners as a function of P(bind) for each building block. Building blocks are called compatible if they are present together in a compound that binds. Each column shows how as the P(bind) of the building block in one position changes (filled shape), so does the number of compatible building blocks in the other two positions (dotted shapes). Shown are the results when building blocks in p_1 (left column), p_2 (middle column) and p_3 (right column) are taken as reference.

more or less prevalent based on their compatibility with the available building blocks rather than their perceived robustness in traditional settings [81]. However, what this analysis determines is that we can be generally be more confident that a compound containing an untested building block is more likely to bind to sEH if it contains a high P(bind) building block in any position (Figure 4.4A–C, Table S1–S3).

4.2.2 Evaluating the P(bind) of building blocks jointly predicts the binding of trisynthons

In the following section, we transition to analyzing DEL selection data at the trisynthon level. We quantify the probability of forming binders by combining building blocks of varying P(bind) values. We note here that while we only demonstrate this analysis on 3-cycle DEL data in this work, we believe our methodology can be applied to DELs of various cycle numbers in order to identify productive BBs at each position. For 2-cycle DELs, we would only need to consider the interaction between a single pair of positions, which we believe would simplify the analysis.

Higher P(bind) in individual positions leads to higher probability of molecule binding

To understand how varying the P(bind) values of building blocks at each position affects the probability of forming a trisynthon that binds, we calculate joint probabilities for pairs of building block positions, using the P(bind) bins shown in Figure 4.2. We refer to bin positions numerically, where 1 is the lowest bin of P(bind) values, $[0.00, 0.20)$, and 5 is the highest bin of P(bind) values, $[0.80, 1.00]$. The subset of compounds where the building block in p_i is **a member of** bin_x (denoted by the set membership symbol $\in$) [119] and the

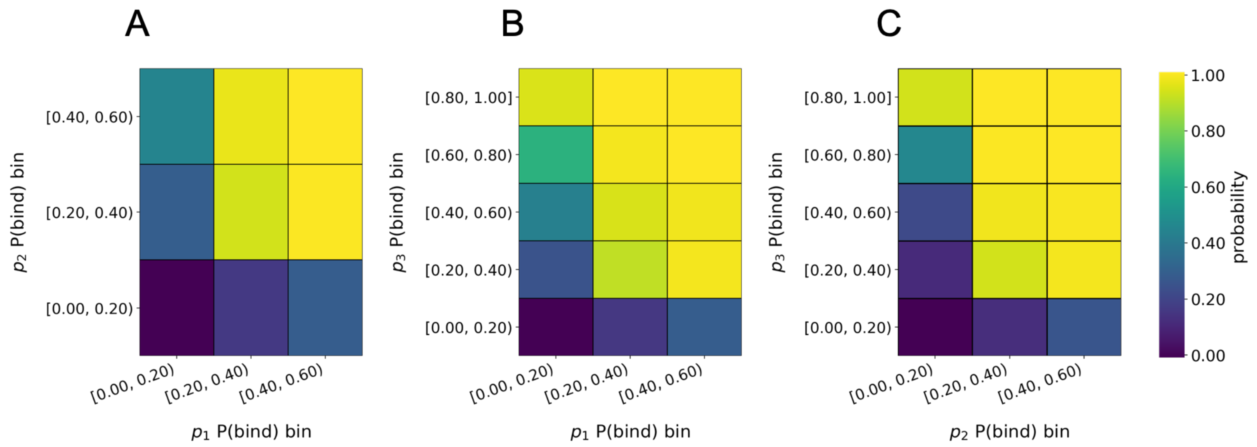


Figure 4.4: Joint probability of forming a binder using P(bind) bins. The P(bind) bins for each position are the same, but p_3 has more bins because its building blocks span a wider range of P(bind) values. Pictured are the joint probabilities of forming binders from building blocks in bins of (A) p_1 and p_2, (B) p_1 and p_3, and (C) p_2 and p_3.

building block in p_j is a member of bin$_y$ is

$$\begin{aligned} S = \{&BB_1BB_2BB_3 \mid \\ &BB_i \in \text{bin}_x, BB_j \in \text{bin}_y\} \end{aligned} \tag{4.4}$$

where $1 \leq x, y \leq 5$. We calculate the number of elements in eq 4.4, N, and then use eq 5.2 to find the joint probability of forming binders for pairs of building block positions (Figure 4.4).

The joint probabilities reveal that typically for **disynthon combinations**, or a pair of building block positions, increasing the P(bind) of either building block increases the probability of forming a binder (Figure 4.4A–C, Table S1–S3). Furthermore, we find that high P(bind) building blocks can be used to **rescue** binding when combined with building blocks with lower P(bind). The higher the P(bind) of a building block in one position is, the lower the P(bind) in another needs to be to achieve the same probability of forming a binder. There is a noticeable increase in the probability of forming a binder when the building blocks in both positions have P(bind) values greater than 0.20 (bin 1) (Figure 4.4A–C, Table S1–S3).

We find additional evidence that the building block in position 3 has the greatest effect on trisynthon binding. When the building block in position 3 has P(bind) value in the range $[0.80, 1.00]$ (bin 5), the probability of forming an binder is never less than 93% (Figure 4.4B, C, Table S2, S3). Building blocks in positions 1 and 2 exhibit far less influence and subsequently do not rescue binding to the same extent that building blocks in position 3 can (Figure 4.4A, Table S1).

Despite variations in the extent to which each building block position contributes, the general trend is clear: introducing a high P(bind) building block in any position increases the probability of forming a compound that binds to sEH. We find that on average combining monosynthons constructively increases P(bind) (Figure 4.4). This means that building blocks that are good independently are still good together on average. While this finding is true on the aggregate, we note that we cannot necessarily propose *a specific* combination of building blocks which includes a high P(bind) building block and expect them to form a binder without considering the chemistry used to form the DEL. For example, the DEL might have used different reactions for linking different categories of building blocks, so that one part of the DEL might contain productive building blocks which simply cannot be linked to other building blocks that would require linking via a different reaction. Or, certain building blocks might be hindered from linking due to steric constraints or other reasons – in other words, the linkage is not synthetically accessible. Thus, we raise an important caveat: the results presented are conditional on the fact that a product is and can be formed, i.e. that the product is a result of what we call compatible building blocks.

Training on building block P(bind) values yields precise predictions for the binding of trisynthons

To determine how much signal P(bind) value alone has in predicting the whether a trisynthon binds, we design a simple test. We randomly split our total dataset into a training set

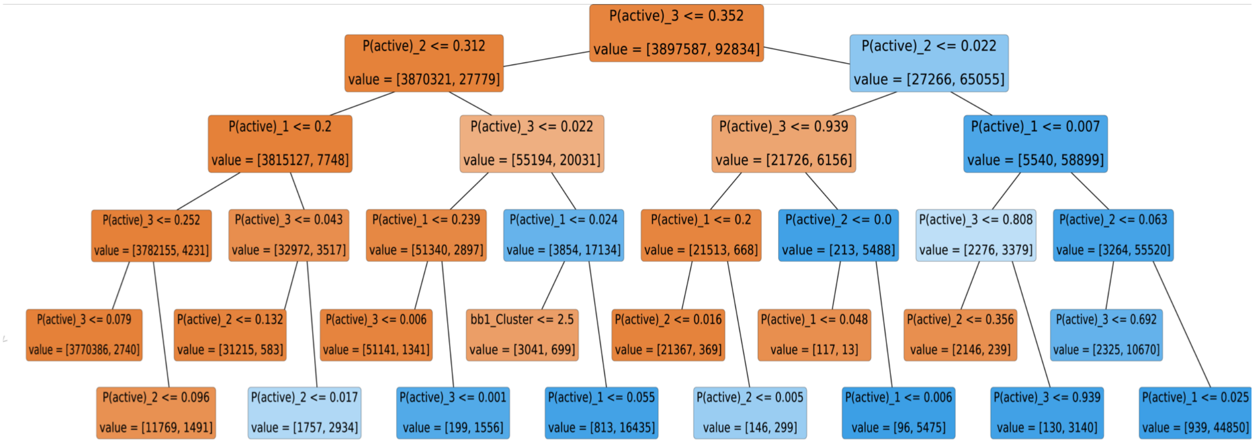


Figure 4.5: Decision tree based on the P(bind) values at each building block position. Each node, shown as boxes, of the decision tree indicates a split of the data on the condition specified in the first line of text in each box. If the condition is true, the data is split into the lower left node, otherwise the data is split to the lower right node. Darker orange nodes indicate a higher proportion of non-binders and darker blue nodes indicate a higher proportion of binders. On the bottom of each node is the value of the number of [non-binding, binding] compounds.

containing 90% of the data and a test set with the remaining 10%, while ensuring that all building blocks in the training set are sampled in the test set. This means all the trisynthons in the test set are strictly new combinations of already tested building blocks, allowing us to evaluate whether P(bind) values can be used to predict if a trisynthon binds when the P(bind) values for each building block can be calculated. In later sections, we tackle the issue of predicting whether trisynthons composed of untested building blocks are binders.

We find that we can identify trisynthons that bind reliably solely using the P(bind) values of their constituent building blocks. We construct a simple decision tree which splits the data based on the P(bind) value at one of the building block positions (Figure 4.5) and evaluate the performance of the model using the metrics precision and recall (see Methods). Of the 10,302 binders in the test set of 443,380 trisynthons, the decision tree model identifies 9432 true positives and incorrectly predicts 364 false positives, resulting in a test precision of 0.963 ($\frac{9432}{9432+364}$) and a test recall of 0.916 ($\frac{9432}{10302}$). The area under the curve (AUC) of the precision-recall curve is 0.961, which is significantly higher than the AUC for a random guessing model, which is equal to the hit rate of the test set ($\frac{10302}{443380} \approx 0.0232$). Given the AUC of a perfect classifier is equal to 1.0, this demonstrates that using building block P(bind) to predict whether a trisynthon binds is highly reliable for this DEL data. A similar analysis can be performed for other DELs and targets to verify the fidelity of this analysis for alternative systems.

We note that our analysis ignores singletons, cases in which species are only enriched in a single selection, and could improve upon this by analyzing the results from replicate selections [77, 140]. As been shown in the literature, some apparent nulls from a single selection have turned out to be high-affinity hits when multiple selections were performed [66] or when the binding affinity of singletons have been further assessed [20, 61]. Thus, we acknowledge that failing to account for singletons in our analysis could potentially result in an increased false negative rate [66], a shortcoming accounted for in other existing methods in the literature [4]. However, DEL practitioners have also leaned towards investigating compounds whose neighboring structures show similar behavior [120], with the goal of identifying families of related ligands and gaining general insights into structure-activity relationships (SARs) for a target of interest [20]. Thus we believe that in spite of its inability to address singletons, our method provides a systematic and reproducible way of elucidating general SARs, and offers value as a better alternative than manually evaluating DEL selection data [120].

4.2.3 Clustering based off chemical similarity estimates the P(bind) of untested building blocks

In this section, we discuss how to use similarity scoring to predict the P(bind) value of building blocks that have not been tested, allowing us to extend the applicability of our method to new data.

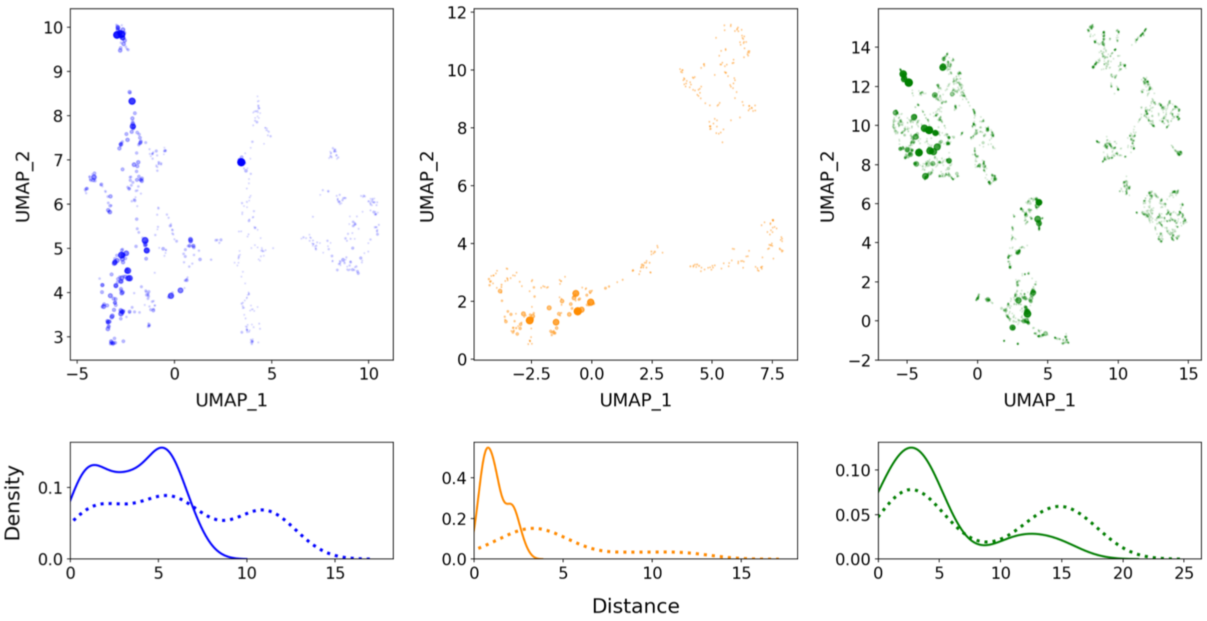


Figure 4.6: (A–C) UMAP projection of chemical space for each library position. The relative chemical distance between building blocks at each position is represented by the distance between points in the UMAP projections. The size and transparency of each point is scaled by the P(bind) of the building block, with larger, solid color dots indicating building blocks with higher P(bind) values. Pictured are the building blocks in (A) p_1, (B) p_2 and (C) p_3. (D–F) Distributions of distances in UMAP space between the top 10 building blocks by P(bind) and randomly selected building blocks. Pictured are the distances between top 10 to top 10 (solid line) and top 10 to random (dotted line) building blocks for (D) p_1, (E) p_2 and (F) p_3.

Building blocks with similar P(bind) are close to each other in projections of chemical space

We hypothesize that by the **similar property principle** [59], building blocks that are similar to each other will have similar P(bind) values [94, 80]. In this study, we elect to use a combination of 3D shape and color Tanimoto, otherwise known as Tanimoto combo as our similarity metric [54, 95]. For each position, we calculate the Tanimoto combo scores between all building blocks and transform these scores into 2D coordinates via Uniform Manifold Approximation and Projection (UMAP), a dimensionality reduction technique [85, 86]. Using the UMAP coordinates, we create an approximation of chemical space, where each building block is represented by a point and the Euclidean distance between points is inversely proportional to the chemical similarity of the respective building blocks (Figure 4.6A–C). We emphasize that no information regarding the P(bind) value of building blocks is introduced in this process.

We find that high P(bind) building blocks generally are much closer (and therefore more similar) to one another than they are to random building blocks (Figure 4.6D–F, Table S4). Here, we define high P(bind) building blocks as the top 10 by descending P(bind) value. On average, the Euclidean distance from a high P(bind) to a randomly selected building block is twice as large as the distance from one high P(bind) building block to another (Figure 4.6D–F, Table S4). This supports our hypothesis that similar building blocks have similar P(bind) values and motivates our next step: to predict the P(bind) value of an untested building block based on the P(bind) values of the building block(s) most similar to it.

We also test if 2D or 3D Tanimoto similarity results in clearer separation of clusters. We observe more random separation between building blocks of similar P(bind) value when using 2D Tanimoto similarity instead of 3D Tanimoto combo (Figure S4). The UMAP projections from 2D Tanimoto show building blocks with similar P(bind) value scattered throughout

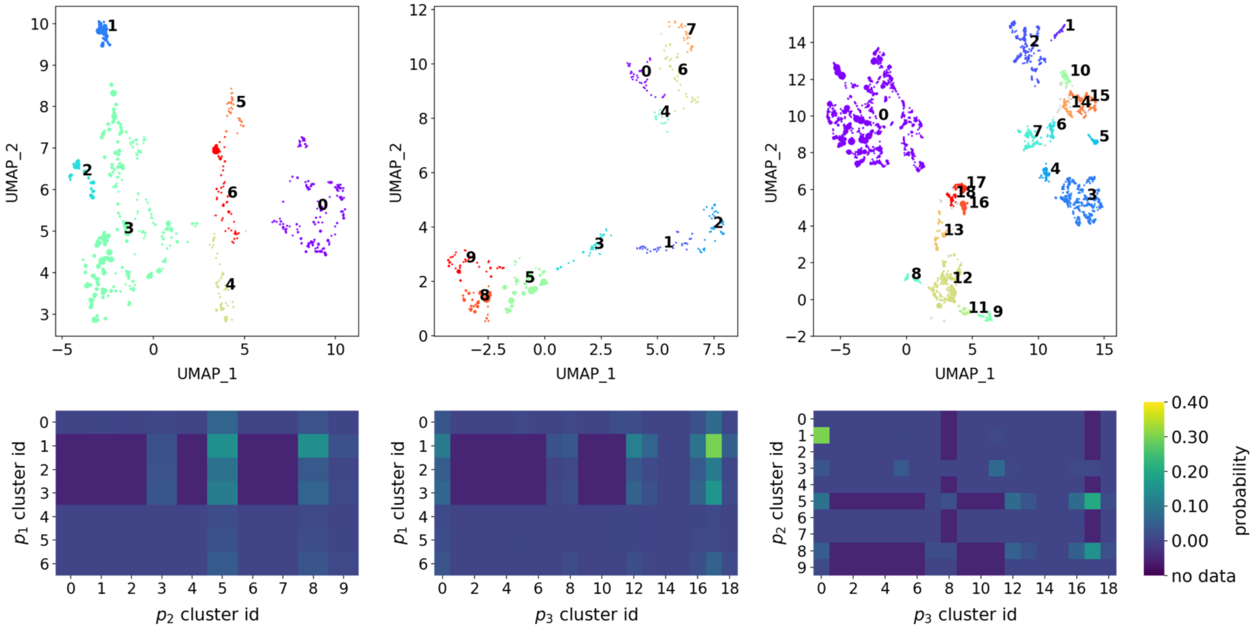


Figure 4.7: HDBSCAN clusters on UMAP projection of each library position. (A–C) We apply HDBSCAN to the UMAP projections of each library position in order to group similar building blocks into clusters. Each cluster is identified visually by a different color and assigned a numeric cluster ID. Pictured are the cluster assignments for (A) p_1, (B) p_2 and (C) p_3. (D–F) Joint probability of forming binders using HDBSCAN clusters. Aggregating trisynthon data by cluster ID allows us to identify which combinations of building blocks have high and low probability of forming binders. We also indicate combinations of building blocks that are never observed in the data. Shown are joint probabilities when combining clusters from (D) p_1 and p_2, (E) p_1 and p_3 and (F) p_2 and p_3.

chemical space with less structure (Figure S4, Table S5). A potential explanation for this is because 3D Tanimoto takes into account multi-conformer overlays of 3D structures, it is able to better relate the binding ability of molecules compared to 2D Tanimoto.

Groupings by chemical similarity are predictive of building block binding

We find that we can form clusters to estimate the P(bind) value of untested building blocks. To do so, we first apply HDBSCAN [17, 84] to the UMAP coordinates of building blocks (Figure 4.6A–C), resulting in a set of clusters for each position in projected chemical space (Figure 4.7A–C). After assigning clusters, we compare the full width at half maximum (FWHM)

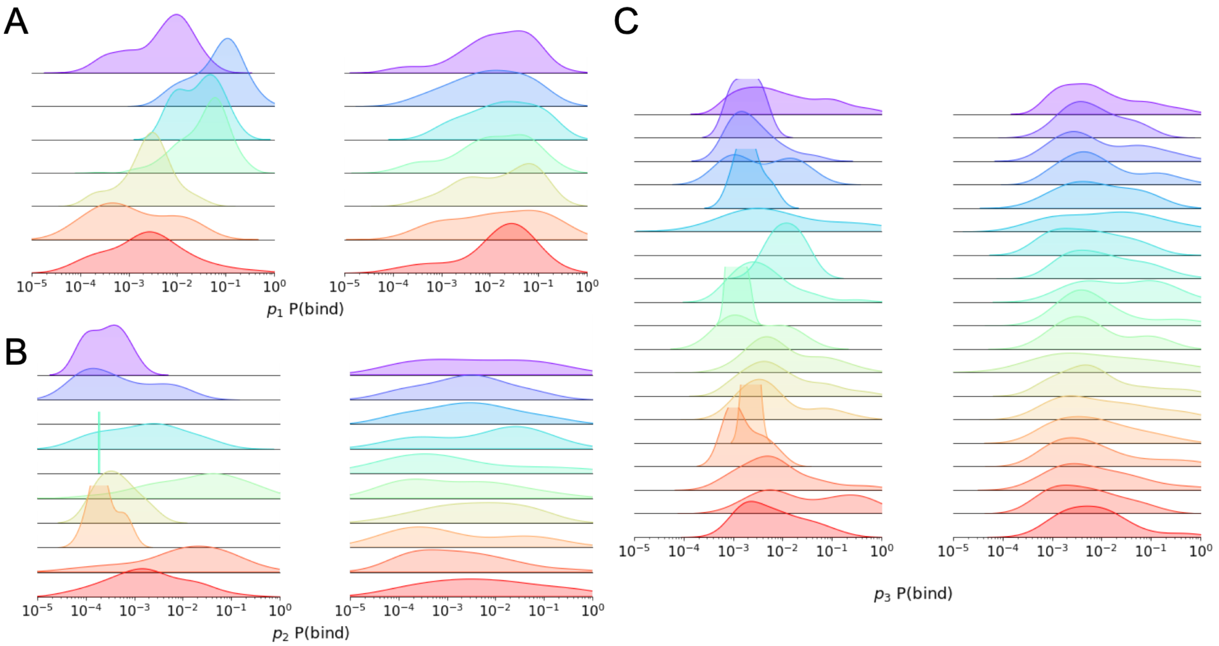


Figure 4.8: Distribution of P(bind) values for clusters. We visualize the distribution of P(bind) values for clusters formed via HDBSCAN (left) and clusters formed from randomly selecting compounds (right). The color of each cluster matches with the color assignments in Figure 4.7. To better visualize each distribution, we remove all building blocks where P(bind) = 0 and plot P(bind) values on a log scale. Empty grids indicate clusters where all members have P(bind) = 0. Shown are results for (A) p_1, (B) p_2 and (C) p_3.

[130] of the distribution of P(bind) values for HDBSCAN-generated clusters to randomly-generated clusters. We find that the average FWHM of the P(bind) distributions from HDBSCAN-generated clusters is less than that for random clustering (Figure S5), showing that compounds tend to be grouped into clusters of somewhat similar P(bind) values. Thus, we conclude that using a building block's cluster assignment to predict its P(bind) value (Figure 4.8) improves accuracy compared to random guessing.

Beyond predicting the P(bind) of untested building blocks, clusters can also be used to identify groups of building blocks that are compatible. After assigning each building block to a cluster, we join the cluster results on the trisynthon data in order to get a list of three cluster IDs (corresponding to the cluster assignment for the building block at each position) for each trisynthon. Grouping by the cluster ID at each position then allows us to calculate the probability of forming compounds that bind to the target for every distinct cluster combination (Figure 4.7D–F, Table S6–S8). We can describe the subset of compounds where the building blocks in positions p_i and p_j are members of the x^{th} cluster of p_i and the y^{th} cluster of p_j as

$$\begin{aligned} S = \Big\{ & BB_1 BB_2 BB_3 \mid \\ & BB_i \in \text{cluster}^i_x, BB_j \in \text{cluster}^j_y \Big\} \end{aligned} \tag{4.5}$$

As before, we calculate the joint probability for disynthons by calculating the number of entries in S, N, and then we apply eq 5.2.

Moreover, we also identify certain combinations of clusters that are not observed in the experimental data (Figure 4.7D–F, Table S6–S8). While the analysis does not indicate why these combinations of building blocks are not observed, we believe characterizing these gaps could be useful. For example, gaps could be new combinations of building blocks that might be desirable to test. On the other hand, these gaps could also indicate that the combination cannot not be made (e.g. due to a DEL being formed using several different reactions, so that certain building blocks cannot be cross-linked given the reactions employed) or that something went wrong experimentally so that even though the combination was thought to be tested, no data was collected. Thus, in some cases, gaps in the data may represent building blocks or combinations of building blocks to avoid and in others, areas to test in further rounds of experimentation.

4.2.4 Application to Holdout Data

In this final section, we demonstrate how we would apply this method in a practical setting, where we would work to guide design of a new DEL using information available from a prior screen. Here, we model this design process by testing performance of our model using a holdout set (using building blocks not seen previously) to mimic a new set of building blocks to test.

Building block level analysis predicts whether trisynthons containing untested building blocks bind to sEH

To simulate an experimental setting in which we would like to choose promising new building blocks to study after performing an initial set of DEL screen(s), we randomly select 5% of the building blocks at each position and remove all trisynthons containing any of those building blocks and place them into a holdout set. The training set, now composed of the remaining compounds, represents the information we might have after an initial experimental screen that had used only a limited set of building blocks. The holdout set represents a set of proposed follow-up candidates that all contain at least one untested building block.

Our workflow proceeds as follows: (1) Calculate the P(bind) values for all the building blocks in our training set (Figure 4.9A). This P(bind) information is used to train a decision tree classifier to predict whether compounds bind to the target of interest. We can also visualize the most productive building blocks at each position to get a sense of what sorts of chemistries may be favored for binding to the target of interest. (2) Compute the 3D Tanimoto combo among all the building blocks at each position (Figure 4.9B). (3) Apply UMAP to each similarity matrix to create a mapping of chemical space for the building blocks at each position and cluster with HDBSCAN to resolve groups of similar building blocks with similar P(bind) values (Figure 4.9C). Peeking into the building blocks in each cluster can further elucidate structures that are potentially favorable for binding to the target and aggregating by cluster ID can identify combinations of building blocks at each position that are more and less likely to result in a binder. (4) Identify a new set of building blocks to mix in combination with already tested ones (Figure 4.9D). (5) Calculate the 3D Tanimoto combo between all the training set building blocks and a new set of building blocks (Figure

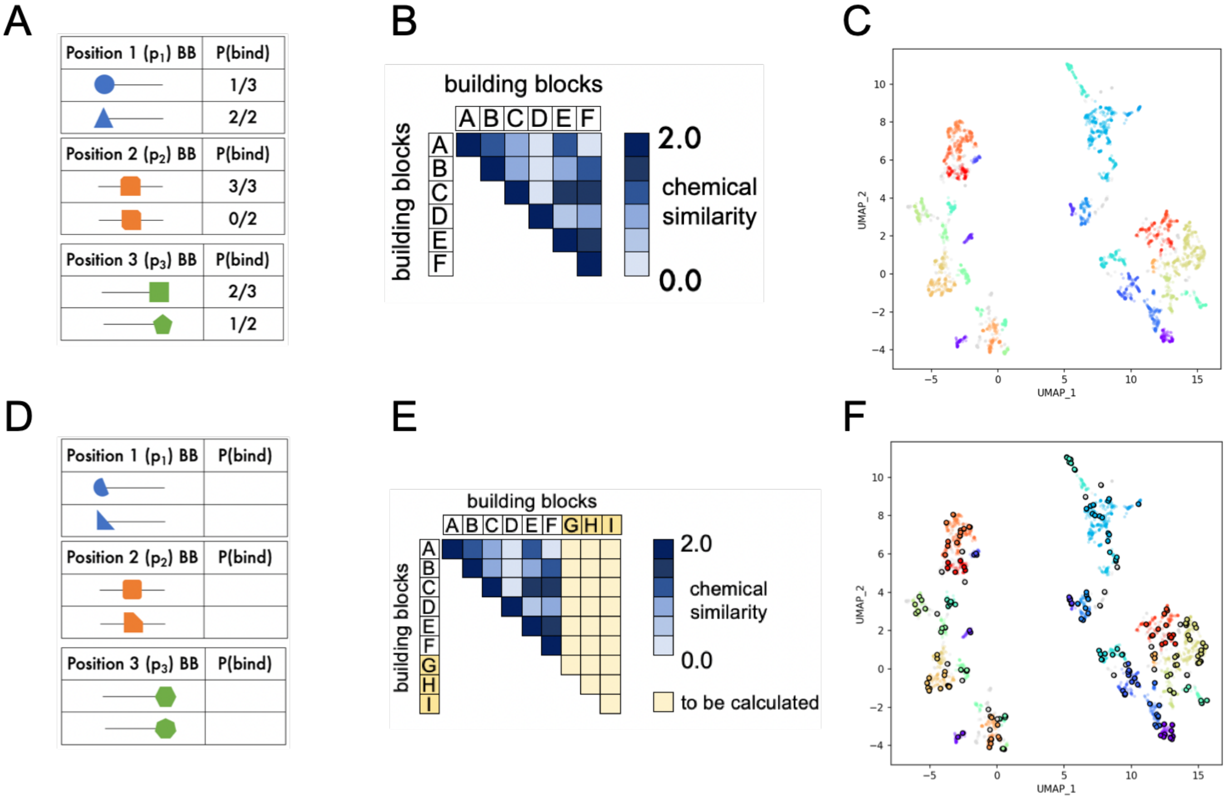


Figure 4.9: Overview of protocol to predict the productivity of out-of-sample building blocks. (A-C) Protocol for processing existing DEL selection data. (A) We calculate the P(bind) metric for all the building blocks in the library. (B) We compute the 3D Tanimoto combo between all the building blocks at each position in the library. (C) We transform similarity scores among building blocks into a mapping of chemical space via UMAP and resolve clusters with HDBSCAN. (D-F) Protocol to apply our methodology to new proposed building blocks. (D) We propose a set of building blocks that have not been tested experimentally. (E) For each position separately, we calculate the 3D Tanimoto combo between the new set of building blocks and the existing ones. (F) We map the new building blocks onto the existing UMAP projections and assign each one to a cluster.

4.9E). (6) Map new building blocks onto the existing UMAP embedding and classify them into the existing clusters (Figure 4.9F). (7) Predict the P(bind) of each building block in the holdout set using the building blocks in its cluster. We explore four different methods of approximating the P(bind) value of each building block in the holdout set:

- the median P(bind) of the cluster
- the mean P(bind) of the cluster
- P(bind) of randomly selected BBs from the cluster
- P(bind) of most similar BBs in the cluster

We train a decision tree classifier using the P(bind) and cluster information of every building block in the training set as input (training precision: 0.960, training recall: 0.926) and then apply the classifier to our holdout set. We find that every method outperforms random guessing by at least an order of magnitude (Figure 4.10). In addition, using the cluster nearest neighbor to approximate untested building block P(bind) gives the best result for predicting the binding of trisynthons to sEH (AUC: 0.799; averaged over 50 random trials) (Figure S6). This finding further supports the argument that on average, chemically similar compounds have greater probability of similar productivity [94, 80].

4.2.5 Building block analysis identifies productive regions of chemical space to probe for subsequent screening rounds

Compared to existing methods in the literature such as the tagFinder [4] and deldenoiser [69], one distinct advantage of our method is the ability to take pooled DEL data screened against a particular target of interest and identify new combinations of building blocks that are more or less likely to form binders for this target. We imagine this can greatly inform subsequent

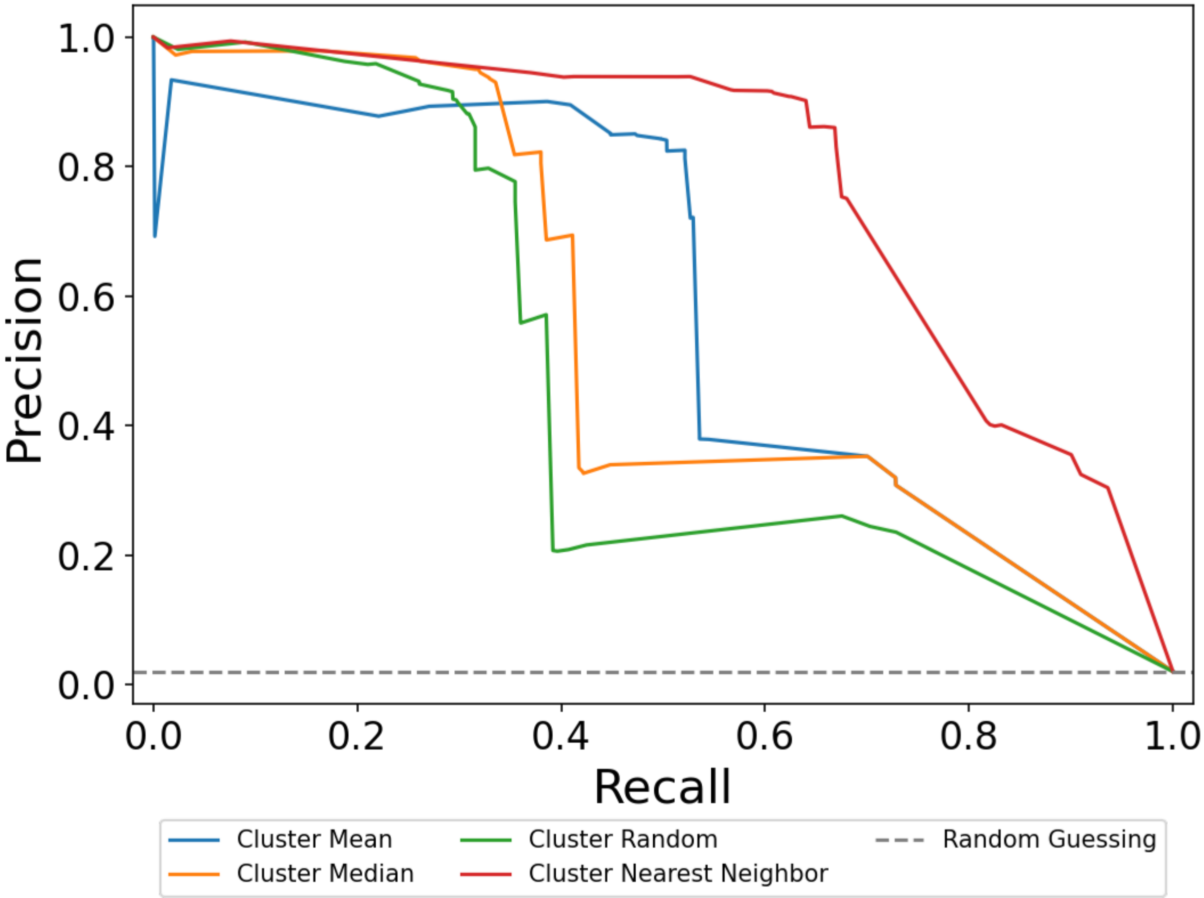


Figure 4.10: Comparison of the AUC of the precision-recall curves for different prediction methods. We evaluate four different ways to estimate the P(bind) of untested building blocks from HDBSCAN clusters. Predictions are made on a test set where each compound contains at least one building block not seen in the training set. The random guessing benchmark is equal to the hit rate of the holdout set, which is approximately 2%.

screening rounds, empowering researchers to either exploit combinations of building blocks that are conducive to forming binders or explore different regions of chemical space to build up a diverse assortment of compounds that bind a target of interest. While we only predict the behavior of building blocks that are similar to existing ones, we imagine that because we combine these building blocks in new combinations, we can build up a diverse set of final products that are likely to bind to the target. As an illustrative example, we showcase a diverse set of compounds that our method successfully determined bind to sEH (Figure S7).

Using similarity scoring and a decision tree model, we predict binders from a set of compounds containing building blocks not seen in the training set at a rate more than an order of magnitude greater than random. The performance of this approach demonstrates that even relatively simple models can estimate whether new trisynthons containing building blocks similar to previously tested ones will bind to sEH. As the intersection between machine learning and DELs grows, we challenge researchers to pay attention to straightforward models like the one employed here and evaluate whether more complex machine learning methods perform significantly better.

4.3 Conclusion

In this work, we applied computational modeling to understand the productivity of building blocks in a set of DELs and predicted how individual building blocks can be combined to form compounds that are likely to bind to a single target, soluble epoxide hydrolase (sEH). We developed a simple and interpretable method to predict the behavior of new building blocks, their interactions with known building blocks, and whether compounds consisting of holdout set building blocks would bind to sEH.

Our model can be an effective baseline for future studies due to its high accuracy and relative simplicity. In the future, it may be interesting to explore the relative merits of more complex deep learning architectures versus similarity-based methods. For example, approaches employed here may have similar performance to more complex neural networks if out-of-sample data resembles existing data, such as during medicinal chemistry efforts or when only small numbers of building blocks and compounds have been explored. On the other hand, when the amount of available data becomes large, it seems likely that deep learning models perform better.

Given the promise of DEL screens for high throughput testing of ideas for drug discovery [82, 72, 69, 10], the refinement of subsequent DEL screens to minimize cost and enhance follow-up on promising structures is likely to improve outcomes in drug discovery. While the ability to gather a vast variety of data in a DEL screen is an experimental advantage, the volume of data poses challenges for interpretation. Improved computational methods are pertinent to aid the experimental workflow. Our method and open-source software [24] can be applied to experimental DEL screens in the future to guide building block selection, identify essential features needed to bind the target of interest, and reduce the search space when following up on potential binders.

4.4 Methods

4.4.1 Data Collection

The dataset was generated from in-house screening of several commercially available DEL libraries (OpenDEL from HitGen) against soluble epoxide hydrolase (sEH). The DEL screen was performed as previously described in Clark et al. [20]. Briefly, N-term his-tagged human Soluble Epoxide Hydrolase (sEH) protein (1 uM, N-term His-tagged) was incubated with pooled DEL libraries in a 100 uL reaction (50 mM HEPES (pH 7.4); 150 mM NaCl;

0.01% Tween-20; 10 mM Imidazole; 1 mM TCEP; 0.1 mg/mL ssDNA). Post-incubation, the protein was captured by magnetic beads (Invitrogen Dynabeads His-Tag, and Pierce Ni-NTA magnetic beads), and the samples washed with buffer. Each round of selection was completed by a heat elution (95°C, 10 min) to separate protein from bound molecules. A new round was initiated by introduction of fresh protein and the process repeated for a total of 3 rounds. In parallel, a matrix-binding only sample was included to account for non-specific binding. Post-selection, samples were PCR amplified and sequenced on a next-generation sequencing platform.

4.4.2 Data Curation

We compiled input files after experiment as comma-separated values (CSVs) containing the SMILES of each composite structure, its experimentally determined read count and the SMILES of its constituent building blocks. In some cases, the SMILES strings for building blocks included the protecting groups used during synthesis which would be removed in the process of constructing full compounds.

As a first step in curation, we classified compounds into binary categories of either "non-binder" or "binder" – with respective labels of 0 and 1 – based on whether their NGS sequencing counts (which we call read counts) were statistically different from 0 at a 95% confidence threshold. We assumed read counts were drawn from a Poisson distribution, a treatment used across several studies in the literature [66, 72, 69]. Using this definition, we selected the top 10K binders by read count value and a random selection of 10M non-binders from the total collection of DEL data screened against sEH. The end result of this is that the lowest read count for any compounds classified as a "binder" was 81, which means we only analyze very clear binders from this dataset (Figure S8). We note that sEH is a particularly rich target for DELs [7] and there may not always be such a clear delineation between binders and non-binders for other targets. In view of this, we include data reporting how the distribution of compounds classified as either a “non-binder” or “binder” to the target changes as we vary the minimum read count threshold (Table S9). We emphasize that our analysis is not on a complete set of DEL selection data, but rather a subset of a larger dataset.

We further curated input files using pandas (v.1.2.1) [102, 131] to remove duplicate compounds and lines containing fields with null entries. In cases where we had building blocks reported in duplicate with both unspecified and specified stereochemistry, we elected to remove all compounds containing the building block with unspecified stereochemistry. This was the case for fewer than 5% of the building blocks in any of the positions of the library. We also removed compounds with building blocks containing boron because we could not generate conformers for them due to force field limitations; this library initially had many boron-containing compounds. Furthermore, some building blocks were reported with protecting groups still present. We used ChemDraw (v.17.0) to generate SMIRKS reactions and used the OEChem toolkit (v.2021.1.1) [97] to deprotect Fmoc, nBoc, methyl ester and ethyl ester groups on those relevant building blocks. We did this to ensure that the presence of protecting groups would not bias our similarity calculations and because the protecting groups were not present in the final products. After applying the deprotecting functions, we saved all the unique building blocks at each library position to separate files. Associated code for these steps can be found at https://github.com/MobleyLab/DEL_analysis.

4.4.3 2D Tanimoto

We calculated 2D Tanimoto scores using RDKit (v.2020.09.1.0) [70] by first converting compounds into Morgan fingerprints [93] with the radius parameter set to 3 bonds.

4.4.4 3D Tanimoto Combo

We calculated 3D Tanimoto combo using the FastROCS toolkit (v.2021.1.1) [97] from OpenEye. The 3D Tanimoto combo score takes into account both volume (shape) and pharmacophore (color) overlap between two molecules to produce an aggregated similarity score. Both the shape and the color scores range from 0 to 1, so the 3D Tanimoto combo has a maximum value of 2.

We first generated up to 200 conformers for each of the building blocks in the library using the Omega toolkit (v.2021.1.1) [97]. We maintained the same settings as the defaults in the Classic OMEGA floe on Orion (Spring 2020), but restricted the stereochemistry of input molecules. For building blocks with unspecified stereochemistry, we used the OEFlipper function in Omega to enumerate all possible stereoisomers and generated up to 200 conformers for each of them.

Next, we used FastROCS to generate an all-by-all matrix of 3D Tanimoto combo scores for all the building blocks (including enumerated stereoisomers) in each library position. We iterated over each conformer of each building block to identify the highest possible shape and color overlap between every pair of compounds. Thus, each entry (i, j) of the 3D Tanimoto combo matrix represented the largest possible overlap in both shape and color between any conformer of compound i and any conformer of compound j. For compounds with multiple stereoisomers, we identified a single stereoisomer that gave the highest similarity to other compounds. To do so, we first enumerated all stereoisomers for the compound and evaluated the similarity of a given stereoisomer to all other compounds. We then selected the stereoisomer that gave the highest average similarity to all other compounds and discarded the rest. Associated code for these steps can be found at https://github.com/MobleyLab/DEL_analysis

Computational Considerations

It was infeasible to calculate all-by-all similarity matrices for libraries on the order of 10^6 molecules, as the task would require performing 10^{12} similarity scoring operations. We instead calculated all-by-all similarity matrices for building blocks at each position individually and then evaluated combinatorial effects at a later step in the analysis. This was a much more computationally tractable approach. Additionally, it mimics considerations involved in library design, one of our key interests, where one might want to use knowledge about current building blocks to help design libraries for screening.

4.4.5 Generating Clusters

We formed clusters based on the 3D Tanimoto combo score of building blocks at each position. First, we transformed 3D Tanimoto combo scores into distances by subtracting each similarity score from 2, the maximum value for the Tanimoto combo. Due to slight variations in the conformer overlay process, distance matrices were not perfectly symmetric and some diagonal elements (a compound to itself) had distances slightly greater than zero. To symmetrize the distance matrix, we averaged it with its transpose and set the diagonal elements to zero.

We then used Uniform Manifold Approximation and Projection (UMAP) (v.0.5.3) [85, 86] to perform a dimensionality reduction of our dataset from 3D to 2D space, both to help with visualization and because we wanted to pick a coordinate space to use for subsequent prediction of properties for new building blocks. This resulted in converting the set of 3D distance matrices into 2D coordinates for each building block. We inputted these coordinates into HDBSCAN (v.0.8.28) [17, 84] to generate clusters for each library position.

We designed and minimized an objective function to determine the optimal number of clusters. Specifically, we arrived at an objective function, L

$$L = n_{noise} + 10 * ICD \tag{4.6}$$

where n_{noise} was the number of points classified as noise and ICD was the average intracluster distance between clustered points for each HDBSCAN run. We performed a grid search over HDBSCAN hyperparameters and calculated the value of the objective function for each set of clusters. We selected the hyperparameters corresponding to the global minimum of the objective function to use for clustering (Figure S10). More information on the design of the objective function can be found in the Supporting Information.

Following cluster assignment, we predicted the cluster assignment for new building blocks by projecting points onto existing UMAP embeddings and applying `approximate_predict` function in HDBSCAN. We elected to use UMAP because it was reported to have better performance and reproducibility than other commonly used methods [6] and was demonstrated to improve the results from clustering algorithms [3]. Associated code for these steps can be found at https://github.com/MobleyLab/DEL_analysis

4.4.6 Model Construction and Evaluation

We used Scikit-Learn (v.0.23.2) [100] to build a decision tree model and assess the quality of model predictions. To reduce the chance of overfitting to the training data, we performed 5-fold cross validation to determine the maximum depth of the decision tree (Figure S11). For our evaluation criteria, we elected to use precision and recall because of the imbalance of class labels in our dataset. Recall evaluates the fraction of all true binders that are correctly identified by a classifier and precision evaluates the fraction of true binders from all compounds classified as binders [26]. For a given imperfect classifier, tuning to yield an increase in precision (better prediction of binders) results in a decrease in recall (fewer binders identified) and vice versa. The quality of a classifier can be described by the extent of this trade-off, which is quantified by the area under the curve (AUC) of the precision-recall curve (PRC)[26, 98]. Associated code for these steps can be found at https://github.com/MobleyLab/DEL_analysis.

Chapter 5

A Building Block Centric Approach to DNA-Encoded Library Design.

Combining small molecule building blocks (BBs) to generate diverse chemical structures, DNA-encoded libraries (DELs) are a means to quickly explore chemical space. With so many potential options for building DELs, it becomes useful to establish general guidelines on how different design decisions affect the character of a final output library. In this chapter, we investigate different ways of selecting building blocks for DEL and the impacts of various considerations such as BB cost and physicochemical properties. We provide a means of picking BBs based on various selection strategies and computationally enumerating sample libraries to evaluate different DEL designs.

5.1 Background

DELs have played an increasingly important role in modern day drug discovery, allowing for a vast number of novel compounds to be synthesized and tested against targets of interest in a single experiment [13, 20, 45, 39]. The high throughput nature of DELs lends itself to an incredibly high degree of flexibility and modularity, empowering experimentalists to not only discover probable hits, but potentially elucidate structure-activity relationships along the way [7, 82, 116]. However, with so many options for building DELs, it becomes useful to establish potential design guidelines. A unique facet of DEL chemistry is that all products are derived from the starting set of selected building blocks (BBs) – thus to an extent, controlling for the properties of BBs impacts the character of the final output library.

In this chapter, we begin by exploring several avenues for computationally culling BBs for DEL synthesis using building block catalogs from the Enamine. Among chemical databases, the Enamine collection is one of the most widely used and currently encompasses over 6 billion compounds (at time of writing) [35]. We use building block catalogs from Enamine because the Enamine collection boasts a wide diversity of compounds that are all readily synthesizable. Given the relative ease to acquire compounds from Enamine, we believe using building block lists from Enamine would be most practical and reproducible for this study. Upon providing various options for selecting BBs, we demonstrate different means by which researchers can explore different concepts for the "quality" of the final output DEL. The focus of this chapter is to provide examples of considerations when designing DELs and to provide a computational framework and open source code for implementing these design decisions. We emphasize that there is no single correct way to build a DEL and the examples we provide are illustrative of how computational modeling can (and should) be integrated in designing DELs. The goals and definition for "success" may vary vastly for each research group.

5.2 Methods

5.2.1 Building Block List Acquisition

We acquired separate building block catalog files (in .sdf format) for purchasable Fmoc-protected amino acid BBs, primary amine BBs and carboxylic acid BBs directly through correspondence with a representative at Enamine. Updated BB lists can also be downloaded by creating an account on the Enamine website or through the database search feature on DataWarrior (v5.5.0) [109]. Each catalog file contains information regarding basic physicochemical properties, cost, IUPAC name, catalog information and RDKit molecule rendering for BBs. We provide the exact raw structure files we used on our GitHub, denoted with the file name following the format: `"[functional group]_stock.sdf"`.

5.2.2 Building Block Truncation

For each Enamine BB stock file, we used the `OELibraryGen` function from OpenEye's OEChem module (v.2021.1.1) [97] to generate truncated versions of each BB via reaction SMIRKS. For Fmoc-protected amino acid BBs, we replaced the -Fmoc, -NH_2 and -COOH groups with -H. For amine BBs, we replaced -NH_2 groups with -H. Lastly for carboxylic acid BBs, we replaced -COOH groups with -H (Figure 5.1).

5.2.3 Building Block Filtering

We applied a SMIRKS filter to remove BBs containing secondary amines, secondary amino acids, and aniline nitrogens. To filter BBs by cost, we used pandas (v.1.2.1) [102] to write queries based on our established cost thresholds. For physicochemical property (PCP) filtering, we calculated the relevant properties for each *BB* before writing queries to filter based on RO3 (MW $<$ 300, cLogP $\leq$ 3, HBD $\leq$ 3, HBA $\leq$3) or RO5 (MW $<$ 500, cLogP $<$ 5, HBD $\leq$ 5, HBA $\leq$ 10) specifications. We exported files containing the canonical SMILES, Enamine ID, cost and PCP of each BB as well as the canonical SMILES of its corresponding truncate as .csv files for further analysis. We provide these cleaned structure files in our GitHub and denote each file as `"[functional group]_df.csv"`

5.2.4 Similarity Calculation

We used RDKit (v.2020.09.01) [70] to calculate the 2D Tanimoto similarity among all truncates. Our rationale was that by stripping the functional groups off each BB, we could directly compare the similarity of the resultant truncates to each other, even if they originated from different BB classes. We used the canonical SMILES for each truncate to generate an RDKit Molecule object and transformed the Molecule into a Morgan fingerprint with radius of 3 bonds and 2048 bits. For each truncate, we calculated its similarity to all other truncates, yielding an all-by-all similarity matrix. We provide a script for this procedure on our GitHub.

5.2.5 Chemical Space Projections

To visualize the coverage of chemical space of different truncate selections, we used a dimensionality reduction technique known as Uniform Manifold Approximation and Projection (UMAP) (v.0.5.3) [85]. UMAP assigns coordinates to each truncate based on the truncate's chemical distance to other truncates. We converted the calculated all-by-all similarity matrix into an all-by-all distance matrix and input this into the UMAP algorithm. Here, we define chemical distance by subtracting the 2D Tanimoto similarity from 1, leading to distance values ranging from 0 (very similar) to 1 (very dissimilar). The resultant output of UMAP was a set of 2D coordinates for each truncate, where chemically similar BBs were closer together in the UMAP projection and more dissimilar BBs were further away from each other, providing a depiction of chemical space. We combined these projections with a Gaussian kernel density estimate from scipy (v.1.5.3) to illustrate coverage of chemical space in this work.

One nuance to note is that we observe instances where multiple distinct BBs result in the same truncate. To deal handle this situation, we de-duplicate redundant truncates before calculating all-by-all similarity matrices. This prevents the similarity matrix from becoming skewed, as all duplicate truncates would have a similarity of 1 to each other. We then run UMAP on the all-by-all truncate similarity matrix, giving us a set of (x, y) coordinates for each *truncate* in UMAP space. We then join these coordinates back on the original list of BBs. Thus, distinct BBs that reduce to the same truncates are given the same coordinates in UMAP space.

5.2.6 Building Block Selection Strategies

We investigated three different BB selection strategies in this work: random, diversity and uniform. For random selection, we used pandas (v.1.2.1) [102] to randomly sample a specified number of BBs. In diversity selection, we used the `MinMaxPicker` function from RDKit to select a maximally diverse set of compounds given an initial seed and a specified number of compounds to select. The algorithm performs similarity calculations based on the underlying Morgan fingerprints. Lastly, in uniform selection, we sample from the density of the UMAP projection. In this final selection strategy, we establish a minimum distance threshold in UMAP space and given an initial truncate, identify truncates that are at least the specified distance away in UMAP space.

5.2.7 Library Enumeration and Analysis

Once we had selected a set of BBs for each library cycle, we performed a library enumeration using the `OELibraryGen` function from OpenEye's OEChem module (v.2021.1.1) [97]. In this work, we limited our analysis to two-cycle libraries where BBs in cycle 1 contain an amine group and BBs in cycle 2 contain a carboxylic acid group. Thus, our only BB linking reaction was a reductive amination, which we represented as a SMIRKS reaction string. Once we enumerated all library products, we calculated various physicochemical properties (PCP) using OpenEye's OEMolProp module (v.2021.1.1) [97]. We provide code to reproduce our specific library enumerations as well as a script to generate custom libraries in our GitHub.

5.3 Results

5.3.1 Proper treatment of building blocks is important for computational studies

To accurately assess the overlap of chemical structures across different BB sets, we generate truncated versions of all building blocks prior to calculating Tanimoto similarity. Otherwise, the presence of the functional groups involved in linking reactions would affect our similarity calculations. For example, the same chemistry found across two different BB classes would be considered as different compounds. In particular, the relative bulk of the Fmoc protecting group compared to the size of our amino acid BBs affect our similarity calculations. For example, we find the 2D Tanimoto similarity of the selected primary amine and primary amino acid BBs to be 0.11 (Figure 5.1 left column). However, after removing their respective functional groups involved in the linking reaction (-NH_2 and -NHFmoc), the 2D Tanimoto similarity between the resultant truncates is 0.09 (Figure 5.1 middle column). While not a significant change, comparing the similarity of truncates allows us to compare the similarity of strictly the atoms which will contribute to the new compound being formed.

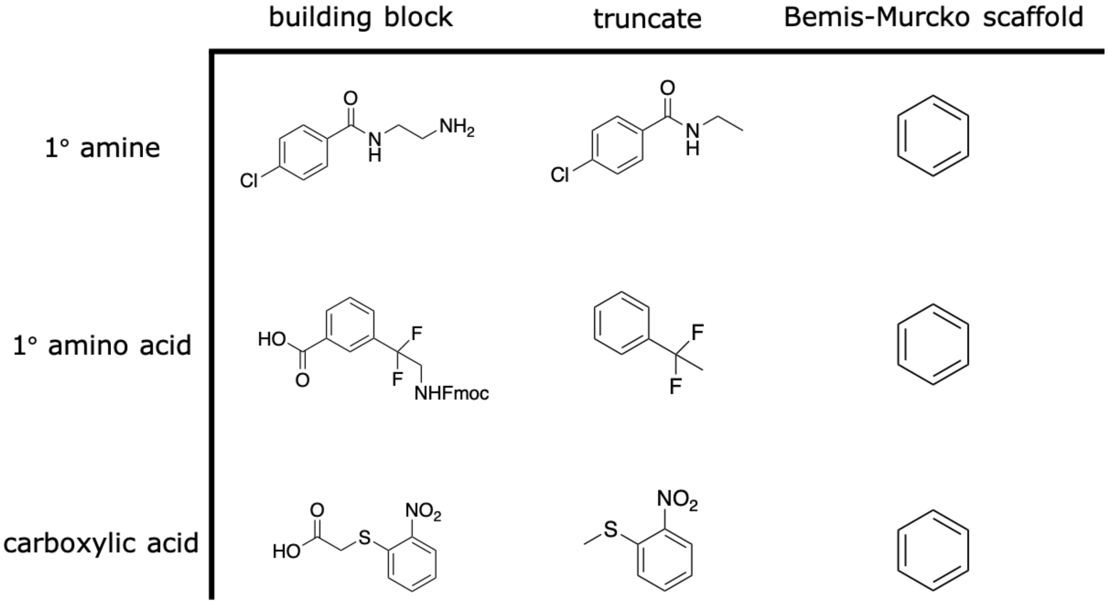


Figure 5.1: Schematic of BB truncates. To compare the chemistries conferred by different BB sets, we analyze the truncates for each BB. From the original building block sourced from Enamine (left column), we replace the functional group involved in the linking reaction in each BB (-NH_2, -NHFmoc, -COOH) with -H to get a truncate (middle column). These truncates are distinct from scaffolds such as Bemis-Murcko scaffolding (right column), which we found reduced structures too much. We elect to use truncates to focus similarity scoring only on the atoms that would comprise the resultant compound following linking reactions.

Moreover, we find that Bemis-Murcko scaffolding is not suitable for reducing the BBs in our datasets (Figure 5.1 right column). Bemis-Murcko scaffolding, which strips a compound to only its rings and any linking atoms connecting ring structures [8], is commonly applied to generalize the chemistries found in large datasets [134, 135]. However, given most compounds in our BB sets are small molecules with few rings and linking atoms, we find that Bemis-Murcko scaffolding reduces BB structures down too much. In our set of example BBs, three chemically distinct building blocks all reduce down to the same scaffold (Figure 5.1). The over-simplification of structures via Bemis-Murcko scaffolding causes structures to appear more similar to each other. Thus, we elect to use truncates and not building blocks nor Bemis-Murcko scaffolds for our similarity calculations in this study. After calculating the

Tanimoto similarity of all truncates across all BB sets to each other, we use the similarity matrix to generate UMAP coordinates that can be used for all three BB sets.

5.3.2 Building block availability and linking reactions are important considerations for library construction

We restrict our analysis to the collection of publicly available and synthesizable on-demand building blocks from the Enamine database. While there are countless places to source BBs for DEL construction, the diversity and accessibility of BBs from the Enamine database serves as a great starting point. For reference, using the BBs available from Enamine, our analysis considers 1396 amino acid BBs, 21109 primary amine BBs and 31091 carboxylic acid BBs. We refer to each of these groups of building blocks (amino acid, primary amine and carboxylic acid) as a **BB set**.

Furthermore, for the scope of this work, we only consider 2-cycle libraries. Previous works in the literature have demonstrated that products from 2-cycle libraries are more likely to possess drug-like physicochemical properties than libraries with additional cycles due to the large size of resulting compounds in the latter [41, 138]. The Paegel group at UCI has also built libraries using Fmoc-amino acid BBs in position 1 of 2-cycle libraries [21, 51, 38]. In this work, we expand upon this previous library design and additionally evaluate the value of replacing cycle 1 primary amino acid BBs with primary amines. We propose that constructing two-cycle DELs with primary amines instead of primary amino acids has multiple advantages.

Firstly, using amines in place of amino acids yields an easier scheme for DEL synthesis. Previously reported two-cycle DELs incorporating Fmoc-amino acid BBs [21, 51, 38] require many synthetic manipulations. On the other hand, the proposed library design using primary amine BBs takes fewer synthetic steps and links BBs together with a single amide, reducing the number of atoms involved in bond synthesis [88].

Secondly, amides are reported to be one of the most common functional groups in bioactive molecules [36], suggesting that resultant libraries are likely to be productive. Amide bond formation is also one of the most prominent reactions in medicinal chemistry hit optimization campaigns, due in part to the increasing availability of unique amines [124]. Furthermore, there is precedent in the literature for DEL selections incorporating compounds of similar design [96, 108, 128].

Finally, we find that the chemical diversity covered by the selection of primary amine BBs in the Enamine database is significantly greater than the diversity covered by primary amino acid BBs (Figure 5.2). We find that all areas covered by the truncates of primary amino acid BBs (Figure 5.2A) are also covered by the truncates of primary amine BBs (Figure 5.2B), also due in part to the fact that there are more than an order of magnitude more primary amine BBs than primary amino acid BBs available in the Enamine database. Moreover, the truncates of primary amine BBs sample many regions of UMAP space not covered by primary amino acid truncates. We believe the availability of primary amine BBs compared to primary amino acid BBs also makes it easier to explore more chemical diversity with our proposed DEL design.

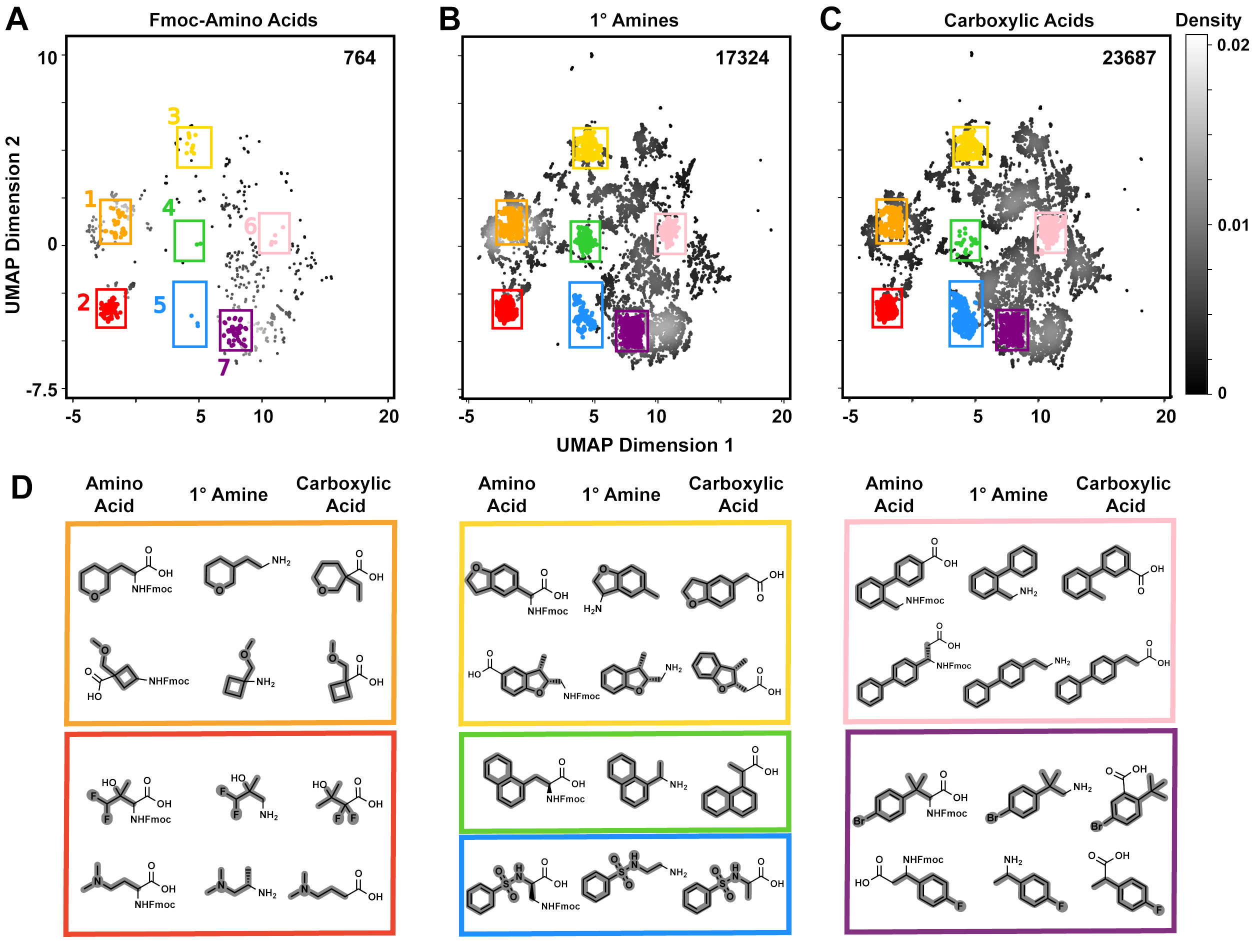


Figure 5.2: Chemical space occupied by primary amino acid, primary amine, and carboxylic acid truncates from Enamine. Density plots arranged by chemical similarity are compared for (A) primary amino acid, (B) primary amine, and (C) carboxylic acid truncates. Grayscale intensity denotes the probability density of points. BB truncate counts in each pool are indicated in the top right of each plot. (D) Example BBs from primary amino acid, primary amine, and carboxylic acid sets are depicted for specific regions of UMAP space (colored points/rectangular outlines in A–C), and their truncates are indicated (bold). Truncates of primary amino acids cover a limited portion of UMAP space relative that of primary amines or carboxylic acids.

5.3.3 Combinatorial libraries should be cheaply available and sample diverse regions of chemical space

When considering the cost of building a DEL, we want to consider the trade-off in how much chemical space coverage we lose when restricting BBs by cost. Costs for building a library can rapidly scale due to the number of BBs that need to be purchased to produce a reasonably sized chemical library. However, restricting BB choices too much may lead to a less productive library. For each BB, we set its cost as the price of the largest purchasable unit quantity (250 mg) listed in its respective Enamine catalog. We represent thresholds for cheap, intermediate and expensive BBs at three different values: ≤\$100, ≤\$250 and ≤\$500 per 250 mg. We find that the availability of BBs across all different BB classes (primary amino acid, primary amine, carboxylic acids) seems to scale roughly linearly with cost (Figure 5.3). However, more costly BBs of the same class seem to mostly be in regions of chemical space that are already covered by cheaper BBs. This suggests that opting to choose a cheaper set of BBs does not severely impact the chemical diversity sampled by the final output library. We also observe additional evidence that primary amine BBs should be preferred over primary amino acid BBs. If filtering only for BBs that cost ≤\$100 per 250 mg, only 104 primary amino acid BBs are available from Enamine, compared to 1614 primary amine building blocks.

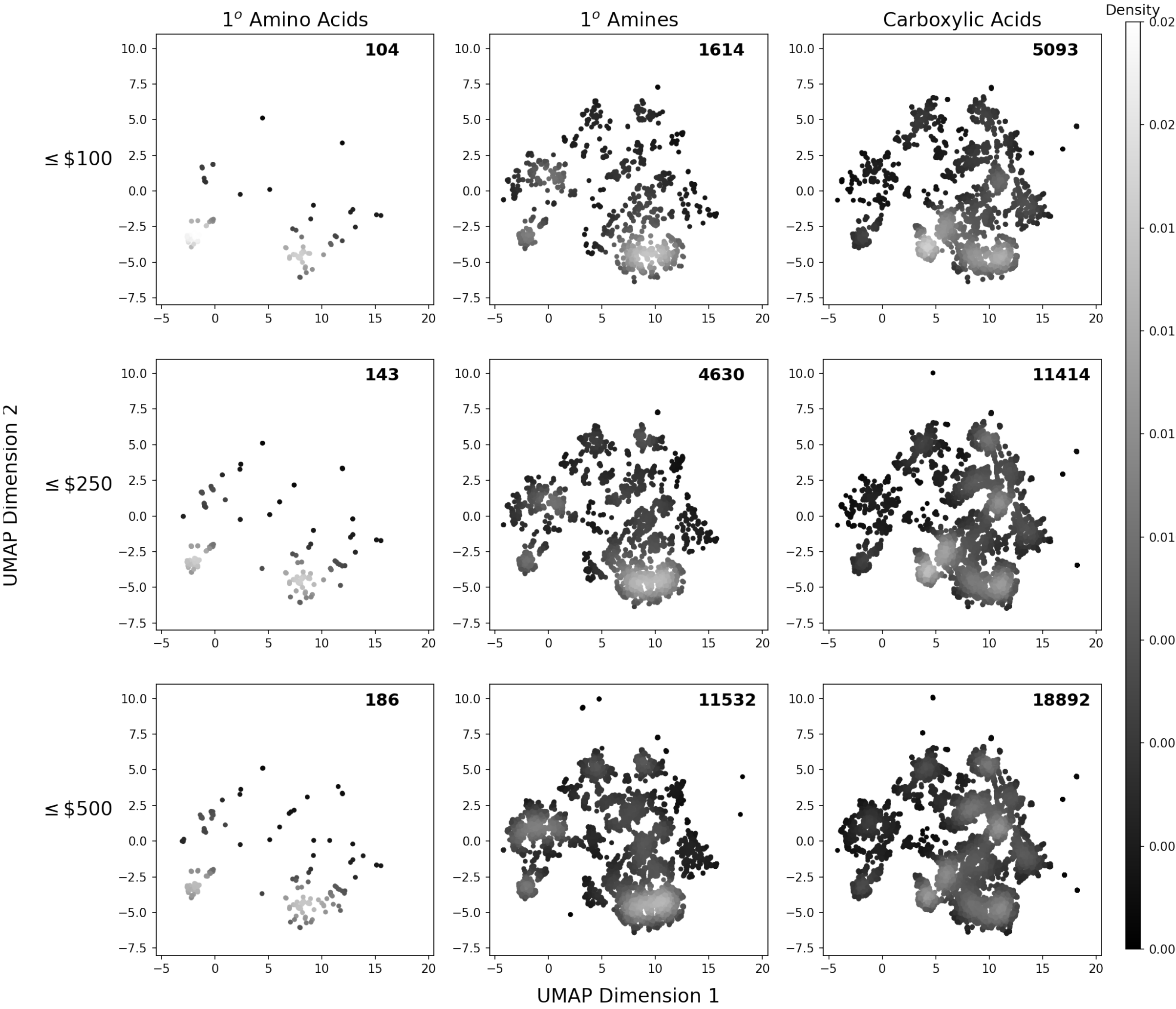


Figure 5.3: Availability of BB truncates following cost filtering. UMAP analyses are shown for three different cost thresholds (≤\$100, ≤\$250, and ≤\$500 per 250 mg) for (A) primary amino acids, (B) primary amines, and (C) carboxylic acids. Grayscale intensity denotes the probability density of points. BB truncate counts in each pool are indicated on the top right of each plot. Primary amines and carboxylic acids cover more diverse chemical space compared to primary amino acids at all cost points, especially at the lowest cost cutoff.

5.3.4 Physicochemical property profiles of individual building blocks can be adjusted to yield a library with generally Ro5 compliant compounds

Lipinski's Rule of Five is widely regarded as a general guideline for designing promising drug candidates [74]. The rule states that a compound should have a molecular weight no greater than 500 Da, partition coefficient less than 5 (cLogP), and fewer than 5 hydrogen bond donors and 10 hydrogen bond acceptors. In enumerating different virtual libraries, we evaluate how selecting BBs under different constraints affects the fraction of Ro5 compliant compounds in the resultant library. While the Ro5 is not a hard and fast rule for what makes a drug, we believe this still provides a useful lens by which to evaluate a library. While forming some products outside of Ro5 space is not deleterious, libraries where the majority of compounds do not comply with these guidelines may be less productive.

We explore how filtering by physicochemical properties at the BB-level affects the characteristics of the products in the output library, using molecular weight (MW) as an example. We select three different molecular weight thresholds, MW $\leq$ 200 Da, $\leq$ 250 Da and $\leq$ 300 Da, representing cases in which all enumerated products of the output library would be well under the Ro5 MW threshold, right at the threshold, or not necessarily under the threshold, respectively. While this is not an essential criteria to adhere to, we refer to the 500 Da threshold as a guideline for what a potentially desirable MW for a drug-like compound should be. We find that across all BB classes, the jump in diversity between MW $\leq$ 200 Da to MW $\leq$ 250 Da BBs is the most significant (Figure 5.4). Not only does this jump numerically increase the number of available BBs in each class the most, but particularly for the carboxylic acid BBs, it brings in BBs that cover new regions of chemical space (Figure 5.4C). While the jump between MW $\leq$ 250 Da to MW $\leq$ 300 Da BBs also results in an appreciable increase in the number of available BBs from Enamine, there appears to be more redundancy in the chemical space covered, at the price of no longer guaranteeing that all final library members have MW $\leq$ 500 Da.

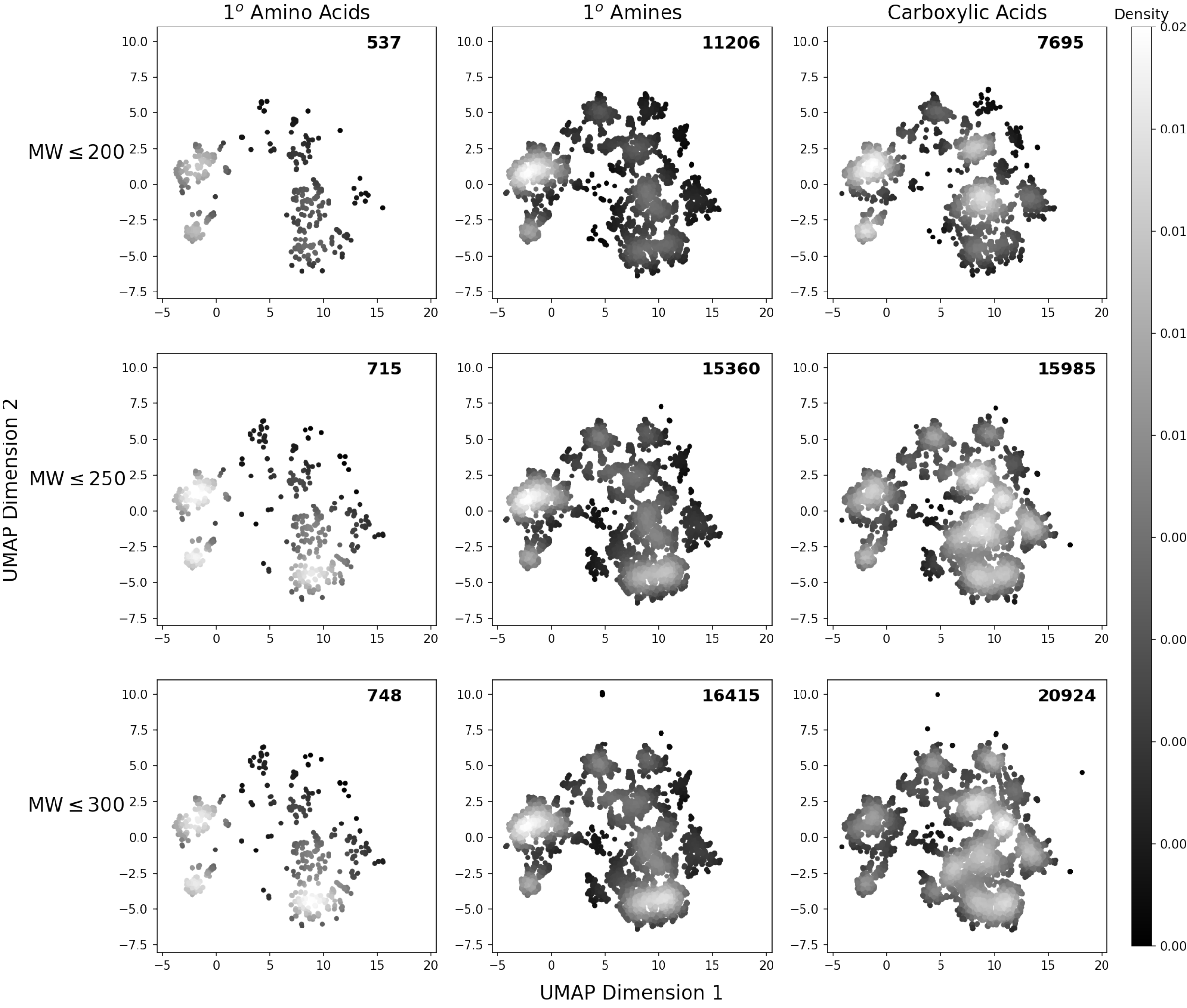


Figure 5.4: Availability of BB truncates following molecular weight filtering. UMAP analyses are shown for three different MW thresholds (≤200 Da, ≤250 Da, and ≤300 Da) for (A) primary amino acids, (B) primary amines, and (C) carboxylic acids. In cases where the purchasable BB included a bulky protecting group, such as Fmoc-protected amino acids, we calculated the molecular weight of the BB with the protecting group removed. Grayscale intensity denotes the probability density. BB truncate counts in each pool are indicated on the top right of each plot. Filtering at these MW thresholds does not substantially alter the accessibility of chemical diversity for any of the BB sets.

5.3.5 Different selection strategies for building blocks

We propose three different strategies to select from a filtered list of building blocks. Two of the methods are fairly standard – selecting either randomly or based on minimizing the Tanimoto similarity of all selected compounds – while the last leverages the UMAP embeddings that we use for visualization throughout this work. We test out these various strategies because we believe they each provide a different framework for building out a DEL.

Random selection is used as a baseline for comparison. A random selection simply samples across the distribution of available BBs, without accounting for any potential experimental biases. Commercially available BBs sets are all likely to have some degree of experimental bias, the source of which could range from certain BBs being more commonly used or easier to synthesize. In any case, it is likely that certain structures are represented more than others. We observe this being the case with the Enamine BB sets across all classes, as all the UMAP projections have areas of more and less density (Figure 5.2). Not surprisingly then, we find more BBs selected from regions of high density than low ones when implementing a random selection strategy (Figure 5.5A).

A common way to ensure one samples diverse regions of chemical space is to minimize the Tanimoto similarity between selected compounds [15]. This is what we hope to achieve with the diversity selection strategy. We observe that with the goal of minimizing the similarity among compounds, the diversity method picks more compounds that appear to be singletons on the periphery of the density map (Figure 5.5B).

Lastly, we introduce the uniform selection method which uses the UMAP projection as a basis for sampling. Both the random and diversity methods do not require UMAP – we only use UMAP for those methods to visualize the selected compounds. However, we design the uniform strategy specifically to select compounds that are spread evenly along UMAP space. We propose this method to evaluate how effective UMAP is as a dimensionality reduction method for chemical similarity and not just as a visualization tool. We find that the uniform strategy samples more from regions of low density and less from regions of high density compared to the random and diversity based methods (Figure 5.5C).

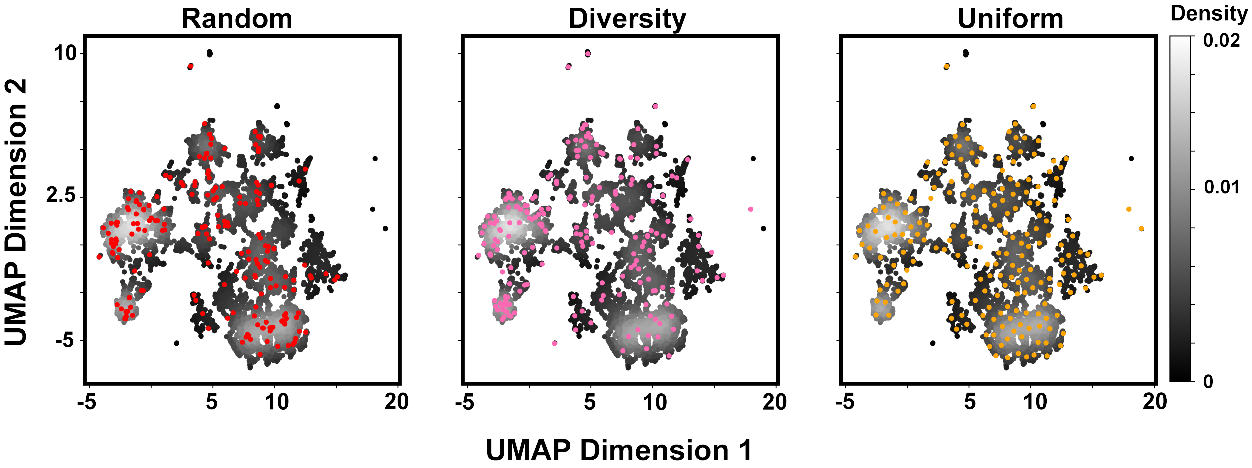


Figure 5.5: Impact of different BB selection strategies on chemical space sampling. Density plots of primary amine truncates arranged by chemical similarity are overlaid with BB selections (colored points) generated by random, diversity, and uniform methods. Grayscale intensity denotes the probability density of points. Random selection is biased toward dense UMAP regions. Diversity selection improves sampling of sparsely populated regions, but remains biased towards densely populated regions. Uniform selection maximizes coverage of UMAP space.

5.3.6 Properties of different enumerated libraries

Having established ways of filtering and selecting building blocks for DEL design, we now define potential metrics to evaluate the "quality" of a library and explore the impact of different design decisions. We propose a set of metrics which look at the physicochemical properties of products, chemical similarity of selected BBs to each other and total cost of building the library. We believe this offers a set of starting considerations that cover a number of potential use cases. Specifically, for physicochemical properties, we look at the distributions of molecular weight, predicted water/octanol partition coefficient (a measure of lipophilicity), total polar surface area and number of hydrogen bond donors – all considerations for Lipinski's Rule of Five – for the enumerated compounds of the proposed library. In our plots, we also show where the Ro5 threshold is for each specific property to provide a sense of the fraction of compliant compounds in the library. For chemical similarity, we choose to evaluate the nearest neighbor Tanimoto for each selected BB in both cycle 1 and cycle 2. In our library designs, this corresponds to primary amine BBs for cycle 1 and carboxylic acid BBs in cycle 2. While we acknowledge that an alternative analysis would be to evaluate the similarity of enumerated *products*, this quickly becomes computationally intractable due to the combinatorial nature of DEL. Furthermore, nearest neighbor similarity for each library cycle provides insight into the chemical diversity of the BBs sampled. Lastly, we show the total cost of purchasing all the selected BBs for each library cycle. While this is more of a pragmatic consideration than anything else, it could be useful to consider trade-offs in cost with any of the aforementioned qualities of a library.

In the following examples, we conduct an analysis of how different BB filters impact the compounds in a final library. To reiterate, we have three dimensions we can adjust along in this study: BB cost, BB molecular weight and BB selection strategy. We examine how changing one of these three variables while keeping the other two constant affects the output library. For the sake of simplicity (since 3^3 library designs are possible here), we select defaults to be: cost $\leq$ \$250 per 250 mg, MW $\leq$ 250 Da and random selection when holding properties constant.

Our first example looks at changing the BB cost filter while holding the MW filter constant (at MW $\leq$ 250 Da) and implementing a random selection strategy (Figure 5.7). We find virtually no difference in the PCP among these libraries with the exception of a slight left skew in the MW and TPSA for the library built with the cheapest BBs (Figure 5.7A, D). Differences become more pronounced when evaluating the nearest neighbor Tanimoto for cycle 1 and 2 BBs. Again, the difference occurs in the library built from the cheapest BBs, with the distribution of BB similarity more skewed towards one (Figure 5.7E, F). This suggests that a cheaper BB set limits the available chemical diversity, resulting in more BBs

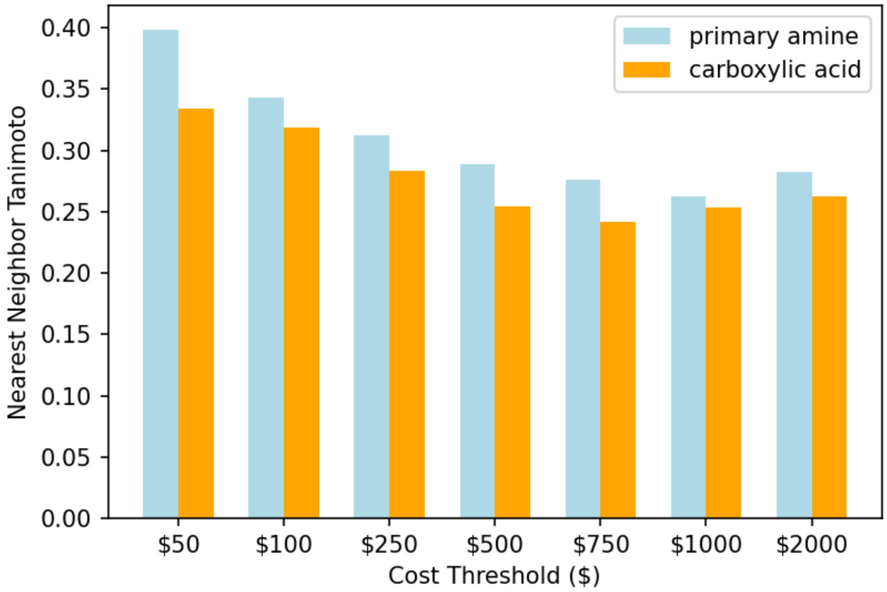


Figure 5.6: Average nearest neighbor (NN) Tanimoto for randomly selected building blocks at different cost thresholds. After filtering BBs at a given cost threshold, we randomly select 192 BBs and calculate the average NN Tanimoto. Generally, more restrictive cost thresholds yield BB selections that are more similar. However, the difference in NN Tanimoto between budget and expensive selections is not significant.

that are similar to each other (Figure 5.6). Lastly, there is an obvious difference in total library cost because we have different BB cost filters. With BB costs restricted at $\leq$ \$100, $\leq$ \$250 and $\leq$ \$500 per 250 mg, we find the total cost to be roughly \$10,000, \$25,000 and \$50,000, respectively for both cycle 1 and cycle 2 BBs (Figure 5.7G, H). A potential takeaway from this analysis is that filtering BBs by cost yields improvements, but only to a certain extent. We observe that the cost $\leq$ \$100 per 250 mg threshold results in a less diverse library with some small polar surface area and low molecular weight compounds. However, there is virtually no difference in the PCP or BB similarities of the compounds produced by the cost $\leq$ \$250 per 250 mg versus cost $\leq$ \$500 per 250 mg filters, apart from the latter costing twice as much to synthesize.

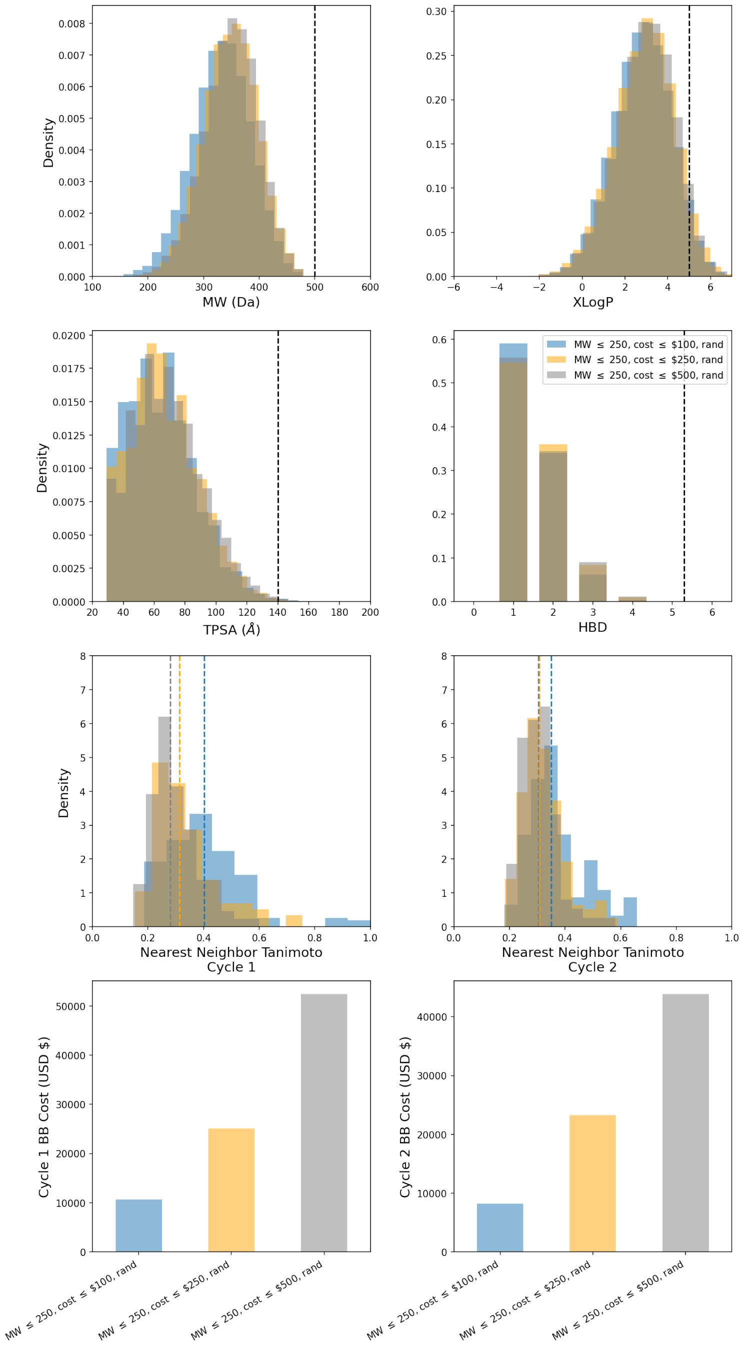


Figure 5.7: Selected properties for three sample libraries differentiated by BB cost threshold. Libraries were enumerated from the condensation product of 192 primary amines and 192 carboxylic acid BBs. Shown are distributions of various properties for the enumerated products of each library (from left to right, top to bottom): molecular weight, predicted octanol/water partition coefficient, total polar surface area, number of hydrogen bond donors, nearest neighbor Tanimoto similarity of selected primary amine BBs, nearest neighbor Tanimoto similarity of selected carboxylic acid BBs, cost of selected primary amine BBs and cost of selected carboxylic acid BBs.

In our next example, we evaluate the effect of changing the BB MW filter while holding cost constant (at cost ≤ \$250 per 250 mg) and implementing a random selection strategy (Figure 5.8). Not surprisingly, culling BBs by molecular weight affects the MW of the final products. We observe that the vast majority of compounds fall well under the maximum possible molecular weight allowable based on how the BBs are filtered. The lowest MW filter (MW ≤ 200 Da) results in a steep drop in compounds with weights approaching 400 Da, suggesting the filter primarily yields small, low molecular weight BBs. Most notably, we find that despite allowing for BBs at each cycle to be up to 300 Da – thus allowing for compounds to weigh up to 600 Da – the MW ≤ 300 Da filter results in very few compounds that violate the MW Ro5 constraint (MW ≤ 500 Da) (Figure 5.8A). However, this observation could be due in part to features of the Enamine BB set. For example, in BBs sets where MW distributions may be skewed, this may not necessarily be the case. It may be worth applying this analysis on different datasets to see if this conclusion generally holds true. Filtering BBs by molecular weight also results in differences in the other PCP for the enumerated products. The lowest MW filter generally results in compounds with lower XLogP, TPSA and fewer hydrogen bond donors; compounds resulting from higher MW filters trend higher for each of those respective properties (Figure 5.8B–D). Higher MW filters lead to slightly less similar BBs, which can be explained as lower MW filters reducing the number of BB options more than higher MW filters (Figure 5.8E, F). Lastly, there is no noticeable difference in cost between the intermediate and high MW filters, although the lowest MW filter seems to result in not only smaller but also cheaper BBs (Figure 5.8G, H). This analysis suggests that filtering BBs by molecular weight has some effect on the PCPs of the downstream library. Researchers should be careful in determining size cutoffs for BBs during library design based on the desired properties for compounds in their projects.

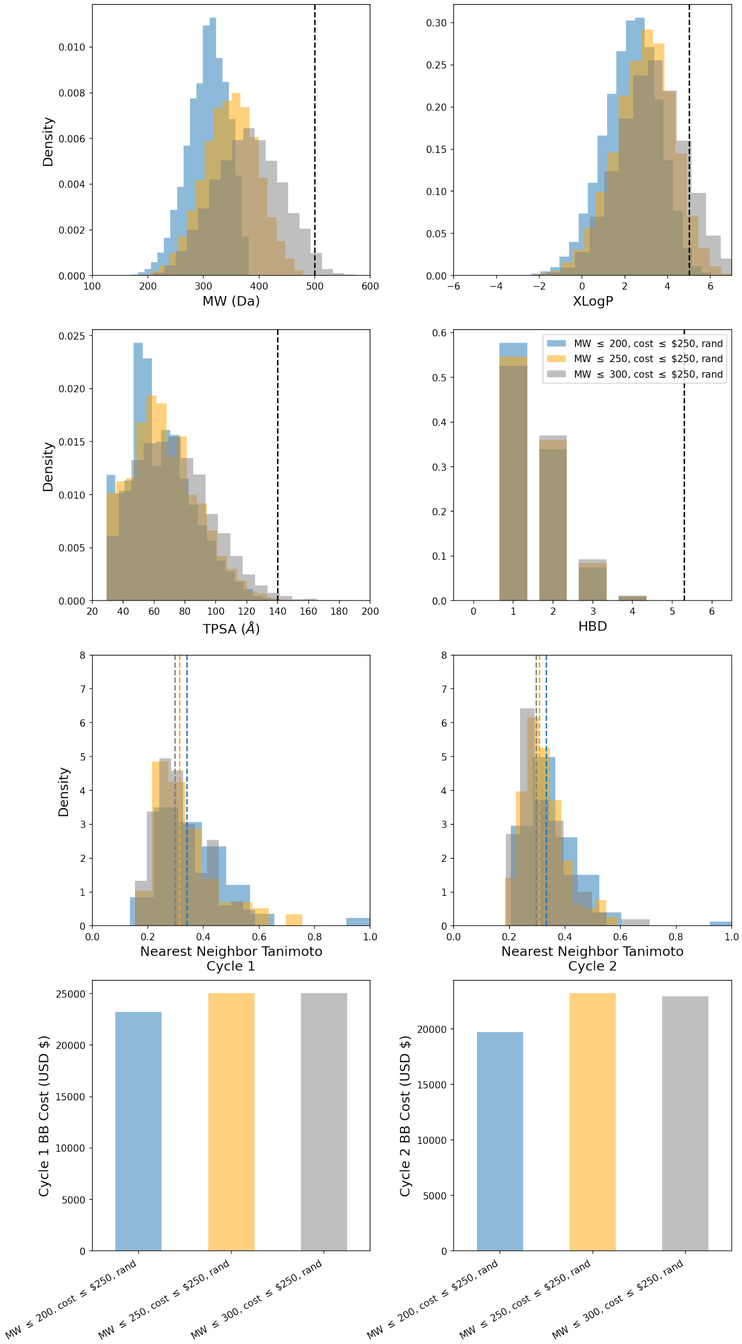


Figure 5.8: Selected properties for three sample libraries differentiated by BB MW threshold. Libraries were enumerated from the condensation product of 192 primary amines and 192 carboxylic acid BBs. Shown are distributions of various properties for the enumerated products of each library (from left to right, top to bottom): molecular weight, predicted octanol/water partition coefficient, total polar surface area, number of hydrogen bond donors, nearest neighbor Tanimoto similarity of selected primary amine BBs, nearest neighbor Tanimoto similarity of selected carboxylic acid BBs, cost of selected primary amine BBs and cost of selected carboxylic acid BBs.

Lastly, we evaluate the effect of holding the BB MW filter and the cost filter constant (MW $\leq$ 250 Da and cost $\leq$ \$250 per 250 mg, respectively) while changing the BB selection strategy. We find that across all library properties, random and diversity selection are on opposite ends with uniform selection in between them. Diversity selection results in heavier, more hydrophilic compounds with larger polar surface area and greater number of hydrogen bond donors than uniform selection (Figure 5.9A–D, gray). Random selection results in lighter, more lipophilic compounds with smaller polar surface area and fewer number of hydrogen bond donors than uniform selection (Figure 5.9A–D, orange). Not surprisingly, diversity selection yields a set of dissimilar BBs and random selection identifies areas of high density containing structures more similar to each other. We find that diversity selection results in a narrow distribution centered around low nearest neighbor Tanimoto similarity and random selection results in a more right-skewed distribution, with uniform between them (Figure 5.9E, F). Lastly, diversity selection results in slightly more expensive BBs than uniform selection whereas random selection results in slightly cheaper BBs (Figure 5.9G, H). One takeaway from this analysis is that diversity selection significantly outperforms both random and uniform selection when it comes to sampling different regions of chemical space. However, it bears warning that researchers should carefully examine the structures that are being selected prior to any actual experimentation. Furthermore, this analysis suggests that uniform selection could be a more balanced approach to early stage library design. We find that uniform selection sits in the middle of random and diversity selection across numerous properties, making a potentially viable alternative for research campaigns that wish to add more BB diversity.

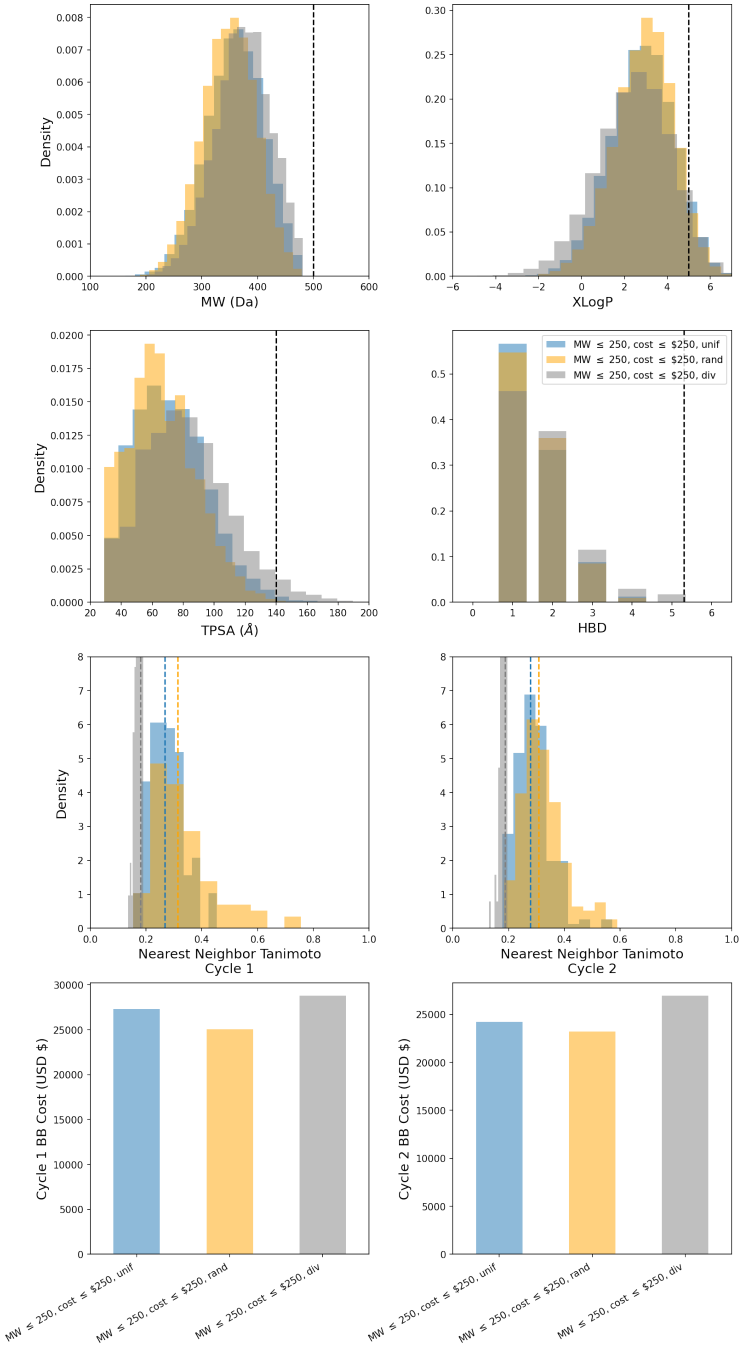


Figure 5.9: Selected properties for three sample libraries differentiated by BB selection strategy. Libraries were enumerated from the condensation product of 192 primary amines and 192 carboxylic acid BBs. Shown are distributions of various properties for the enumerated products of each library (from left to right, top to bottom): molecular weight, predicted octanol/water partition coefficient, total polar surface area, number of hydrogen bond donors, nearest neighbor Tanimoto similarity of selected primary amine BBs, nearest neighbor Tanimoto similarity of selected carboxylic acid BBs, cost of selected primary amine BBs and cost of selected carboxylic acid BBs.

5.3.7 Evaluation of BB selection rules for sEH binders

To evaluate the relative merits of our proposed selection rules, we extend our analysis to experimental binding data. An alternate way to define the "quality" of BB selection methods is to determine how well they lead to the recovery of experimental binders. We are unable to evaluate this for the Enamine datasets without first synthesizing the proposed libraries and running binding assays – which is well outside the scope of this work. Instead, we analyze analyze a subset of the Hitgen OpenDEL library published as part of our work in Chapter 4 [139]. Briefly, this data comes from pooled three-cycle DEL data screened against soluble epoxide hydrolase (sEH). In this library, cycles 1 and 3 mostly consist of primary amine BBs, and cycle 2 mainly consists of carboxylic acid BBs. For each compound in the library, we assign a binary label for whether the compound binds to sEH based on experimentally determined DNA-read count values. More detailed information regarding the sEH dataset can be found in the Methods section of Chapter 4.

We modify our sEH data to correspond to the two-cycle library design scheme we present for the Enamine BB sets earlier in this chapter. We only evaluate the first two cycles of the sEH library and select BBs so that cycle 1 only consists of primary amine BBs and cycle 2 only consists of carboxylic acid BBs. Although the sEH dataset is a three-cycle library, we are able to quantify the activity of two-cycle disynthons by aggregating BB data in the first two positions. Known as disynthon aggregation [82, 72], this method allows us to quantify the productivity of a pair of BBs. Using the notation we establish in Chapter 4, we define S to be the set of trisynthons where an arbitrary building block bb_x is in the first position in the library and an arbitrary building block bb_y is in the second position of the library. We express this as:

$$S = \{BB_1 BB_2 BB_3 \mid BB_1 = bb_x, BB_2 = bb_y\} \tag{5.1}$$

The productivity, or **P(bind)**, for a given disynthon pair (bb_x, bb_y) can then be calculated by the following equation:

$$P(\text{bind}) = \frac{\sum_{k=1}^{N} I_k}{N} \tag{5.2}$$

where N is the number of compounds in S and I_k is 1 if the k^{th} compound in S binds to the target and 0 otherwise, as defined in Section 4.4.2. Thus, the P(bind) value we calculate tells us the fraction of compounds that bind to the target for all compounds with a given pair of building blocks in the first two positions.

We identify a set of primary amine BBs for cycle 1 and carboxylic acid BBs for cycle 2 from the sEH dataset to use for our selection rule evaluation. One aspect of the sEH dataset that makes it challenging to analyze is that it consists of pooled DEL data and is a subset of a larger dataset. Thus, it is not guaranteed that we have binding data for all possible enumerated products in the library (see Section 4.2.2 for more details). However, using our approach of similarity scoring combined with dimensionality reduction and clustering, we determine combinations of building blocks that are all synthetically compatible with each other (Figure 4.7). Specifically, we identify 267 primary amine BBs for cycle 1 and 110 carboxylic acid BBs for cycle 2. We verify that we have binding data for all 29370 possible products (267 BB_1 × 110 BB_2).

We design a set of experiments to assess the quality of our BB selections. In each experiment, we first sample a specified fraction of BBs from each position using each of our three selection rules: random, diversity and uniform. We then enumerate all possible library products and calculate three different metrics: the average P(bind), the maximum P(bind) and the recall of top binders. We define top binders as the top 1% of disynthon pairs by P(bind) value from our 29,370 possible products. To account for the variation across random seeds, especially for lower fractions of BBs sampled, we repeat each selection across 50 trials.

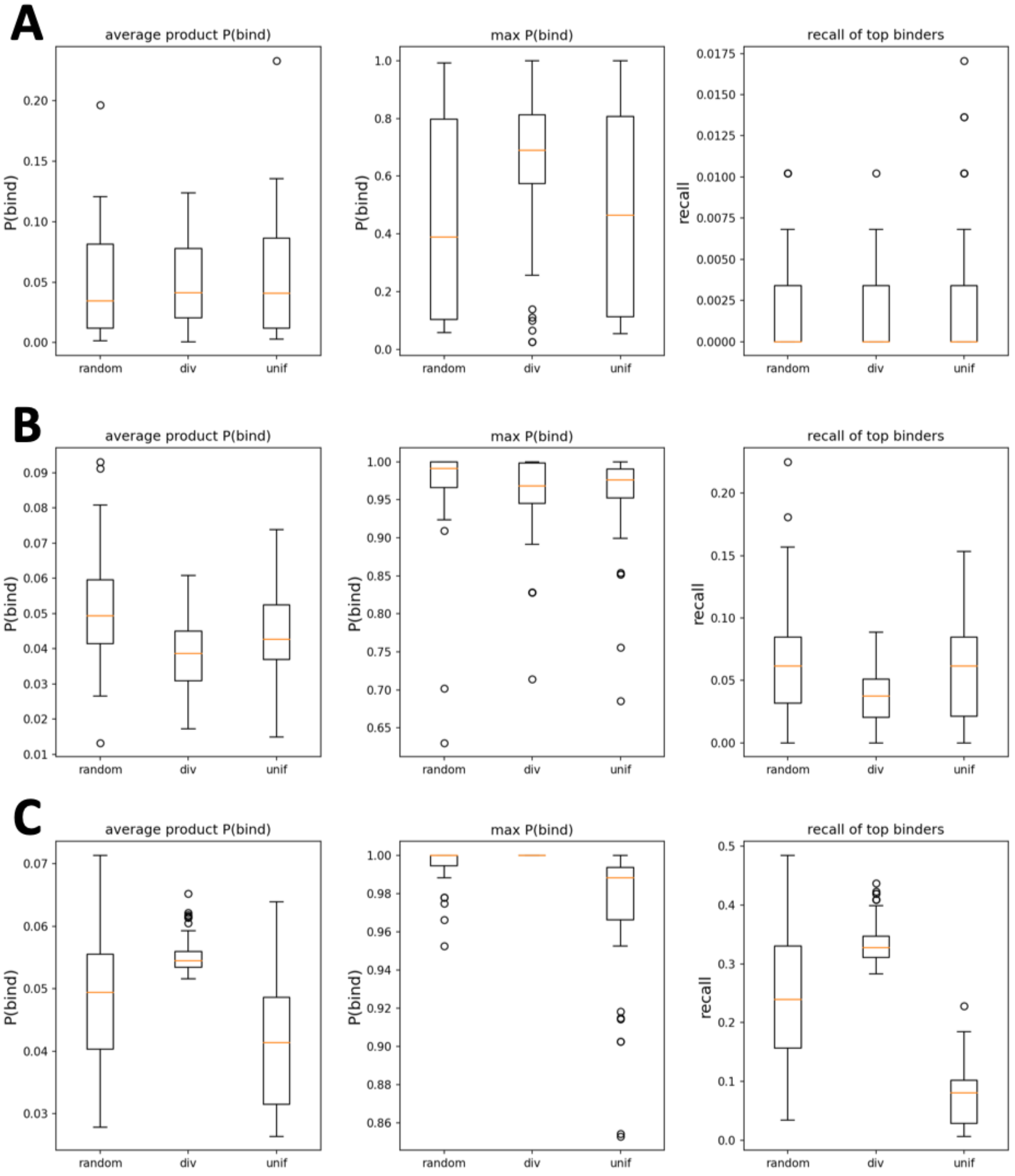


Figure 5.10: Productivity of enumerated libraries from different selection strategies. We apply our three different BB selection methods – random, diversity and uniform – to a subset of the sEH dataset. For each BB selection we enumerate all possible products of the library and measure quality of the BB selections. We evaluate selections across three metrics, the average P(bind), the maximum P(bind) and the fraction of top binders identified from the enumerated products. Shown are the results across 50 random trials for each selection method. We set the fraction of BBs sampled from each position to be (A) 5% (B) 25% and (C) 50% of the total BBs. While differences in selection strategy are not pronounced at lower fractions of BBs sampled, the diversity selection strategy yields more productive libraries when sampling a larger number of BBs.

We find that the differences between selection rules are minimal when sampling only 5% of the BBs at each position (Figure 5.10A). All three selection rules result in a set of compounds with a similar distribution of average P(bind) values. Furthermore, the distribution of maximum P(bind) for each set of compounds basically spans from 0 to 1, suggesting the sample size may be too small for high P(bind) compounds to be found consistently. However, we do observe that the distribution of maximum P(bind) for the diversity selection rule is shifted towards higher values compared to the other two methods. Lastly, we find that the recall of top binders is fairly consistent between all three approaches with the majority of trials resulting in no top binders being found. We observe a couple trials for all three selection strategies where a few top binders are recalled. At such a small sample size of BBs (13 BB_1 × 5 BB_2; 65 possible products), it appears that the differences in selection strategies are minimal and that the effects of randomness dominate.

The differences in the library products resulting from various selection rules become more evident when we increase the fraction of BBs sampled at each position to 25% (Figure 5.10B). At this threshold, we observe that the distribution of average product P(bind) for diversity selection has smaller variance but also a lower median value than both the random and uniform strategies. In terms of max P(bind), all methods perform similarly, but we notice that the distribution for diversity selection is a little lower. We also find that the distribution for the recall of top binders follows a similar trend among the three selection strategies. Both random and uniform selection outperform diversity selection, but the latter has smaller variance. At a more significant fraction of the available BBs sampled, we find that generally random or uniform strategies would be preferable to diversity selection.

We observe that the improved performance of the diversity selection rule becomes more pronounced when sampling 50% of the BBs at each position (Figure 5.10C). At this much larger sample size (133 BB_1 × 55 BB_2; 7315 possible products), the different selection rules lead to libraries with far more contrast. Here, the distribution of average product P(bind) for the diversity selection has smaller variance and a higher median value, suggesting that diversity selection consistently produces libraries containing compounds that are more productive on average. We observe in the distribution of max P(bind) that diversity selection identifies at least one binder with the maximum P(bind) value of 1 in every trial. The performance of the other two methods is not nearly as consistent, with uniform selection in particular suffering from several trials with lower performance. Lastly, differences in the distributions for the recall of top binders is also consistent with previous findings, where diversity selection yields the most consistent and best results. We find that the uniform method leads to libraries that fall behind even random selection in terms recall of top binders. At large fractions of the library sampled, diversity selection delivers most consistently across all performance metrics.

5.4 Conclusion

With myriad ways to approach DEL design, it becomes imperative to develop computational tools to guide experimental planning. In this chapter, we demonstrated a way to take a publicly available dataset and design a variety of libraries based on different desired experimental outcomes. We explored various ways in which experimentalists could try to slice data (e.g. cost, PCP, chemical similarity) and provided some metrics to define the "quality" of a library. Moreover, we showed how certain design considerations such as the choice of BB class has a significant effect on the final products of a library whereas considerations such as cost have a much more diminished effect. While this work is by no means a catch-all for DEL design, we believe it can serve as a useful reference which other scientists can adapt to fit their own research goals.

Chapter 6

Future Directions

6.1 Development of a benchmark for DEL x ML

In recent years, there has been growing adoption of DNA-encoded libraries (DELs) for early stage drug discovery with some candidates making it as far as clinical trials [20, 7, 45]. While experimental DEL technology continues to improve, there is a growing need to develop better computational models to extract signal from inherently noisy DEL selection datasets. The DEL field has seen the expansion of many different computational DEL frameworks, ranging from the generation of synthetic DEL data [69, 111, 110] to mathematical modeling of DNA sequencing counts for noise reduction [66, 72] to different interpretations of modeling DEL data [10, 18]. Most notably in 2020, McCloskey et al. introduced a computational framework for training machine learning (ML) models with experimental DEL data to identify binders for the targets sEH, ERα and c-KIT [82]. As one of the first demonstrations of how ML models can be trained on DEL data to identify out-of-sample binders, the work has paved the way for many other DEL x ML studies [72, 139, 2, 18, 10]. With the rise of so many different methods of integrating ML with DEL data, a system of benchmarking different procedures on a standardized set of publicly available DEL datasets would allow the field to compare new methods against each other moving forward. Following the Pande Group's prior work on MoleculeNet [133], we envision a similar framework being developed in the future to assess the quality of new ML models on DEL data. The benchmark would focus on standardizing several major pillars which we outline below: (1) public DEL dataset curation, (2) featurization of DEL data, (3) DEL de-noising approaches and (4) DEL modeling choices and (5) evaluation of model performance.

6.1.1 Dataset curation

Overview of current publicly available DEL datasets

There is currently no alignment on how DEL data should be reported and what experimental information is best suited for training an ML model. Our proposed standardization is that all datasets should be reported with the SMILES of all enumerated products and the SMILES of the *building blocks* used at each cycle. As we demonstrated in Chapters 4 and 5, there are various ways to treat BBs computationally such as removing protecting groups or creating truncates, but reporting the actual BBs purchased makes the data additionally reproducible at an experimental level. For each compound, both pre- and post-selection DNA read counts should also be provided, and in replicate if possible. This availability of this data would make it possible to test and compare a variety of DEL-denoising methodologies. Both pre- and post-selection DNA read counts can also be converted into binary labels for classification tasks or a numeric output label for regression tasks. An ideal benchmark would contain DEL datasets screened against a wide array of targets which are standardized and prepared for evaluation. Furthermore, there are clear distinctions between solid and solution phase DELs [39], and data for both should be included in the benchmark in the future. However, the current availability of public DEL datasets is limited. There are few options available in the literature, which we now discuss.

DOS-DEL Currently, the most widely adopted data for testing new computational DEL methods is the DOS-DEL dataset from Gerry et al. [43]. The set is a three-cycle library consisting of 8 scaffolds in cycle 1, 114 BBs in cycle 2 and 118 BBs in cycle 3 for a total of 107,616 products. While the library is designed to maximize chemical diversity and many of the products are shown to have "drug-like" properties, the size of the library is still orders of magnitude smaller than what is feasible for many DELs. Products of the library are screened against two well-studied protein targets: carbonic anhydrase IX (CA-IX) and horseradish peroxidase (HRP). The value of the library is that it reports replicates of the DNA abundance counts for both post-screening and no-target control experiments. This data provides researchers an opportunity to evaluate the quality of computational DEL models which consider both pre- and post-selection DNA read counts.

OpenDEL As part of the work presented in Chapter 4, we released portions of the HitGen OpenDEL dataset containing over 10M compounds screened against soluble epoxide hydrolase (sEH) [139]. The released data contains 617 BBs in cycle 1, 338 BBs in cycle 2 and 4529 BBs in cycle 3, but is only a subset of a larger dataset, so not all enumerated products are present. Furthermore, the data also comes from pooled DEL selections, meaning not all BBs are compatible given specific linking reactions. We believe this makes for a challenging dataset and discuss this more in Chapter 4. Lastly, the DNA read counts reported in the dataset are already adjusted using a Poisson treatment [66, 72, 69], so no raw and control values for read counts are provided. This could be a potential downside for researchers looking to evaluate DEL de-noising approaches.

Triazine library Clark et al. [20] presents two libraries built around a triazine scaffold: DEL-A is a three-cycle library with 192 BBs at each position (7M possible products) and

DEL-B is a four-cycle library with 192 BBs in cycle 1, 32 BBs in cycle 2, 340 BBs in cycle 3 and 384 BBs in cycle 4 (802M possible products). Both libraries have been experimentally synthesized and screened to identify inhibitors for the enzymes Aurora A kinase and p38 MAP kinase. However, screening results for only around 3K compounds are provided by the authors. Satz [111] has developed code to model the three-cycle triazine library using random probabilities to simulate BB reaction yields and generating equilibrium association constants for each product from a half-normal distribution.

Simulated DEL data Simulated DEL design has been explored as a way to generate data for ML training. Komar and Kalinic [69] developed a framework allowing for the simulation of two- to six-cycle libraries spanning from 1M to 1B compounds with a default size of 100M. Product reaction yields were sampled from a beta distribution and association constants were drawn from a half-normal distribution. Parameters for density (fraction of strong binders), DNA sequencing depth (the average number of reads per DNA tag) and tag imbalance are all adjustable as well.

6.1.2 Dataset splitting

We propose offering a few dataset splitting strategies to ensure prediction tasks are fairly assessed. Methods include random, stratified, scaffold and time splits [134, 133]. Additionally, unique to DEL data is the ability to split on building block cycles [72].

6.1.3 Data featurization

A researcher's choice on how to represent molecular information heavily impacts the resulting output of an ML model. We suggest a number of featurization choices to assess differences in the characteristics of a molecule an ML model learns. Feature choices include one-hot encoding [19], extended connectivity fingerprints (ECFP) [105] and graph-based representations [64, 22].

6.1.4 DEL de-noising approaches

A heavy emphasis of current computational treatments of DELs is determining how to adapt raw sequencing read counts into experimentally accurate enrichment values. Approaches currently reported in the literature include disynthon aggregation [82, 139], Poisson enrichment ratios [72], sparse learning [69], negative binomial regression models [76] and partial product modeling [10]. Implementing all these various de-noising procedures on the set of standardized datasets should be feasible using the open-source package, DeepChem [101].

6.1.5 Machine Learning Models

A variety of models have been used in the literature for DEL, many of which can already be implemented using sklearn [100] and DeepChem [101]. Approaches include decision tree/random forest [82, 139], graph convolutional neural networks (GCNN) [82, 10], feed-forward neural networks (FFNN) and directed message-passing networks (D-MPNN) [72].

6.1.6 Metrics for Model Performance

We propose using mean absolute error (MAE) and root mean squared error (RMSE) to evaluate model performance on regression-based tasks and also include the negative log-likelihood (NLL) loss from Lim et al [72]. Metrics for classification tasks can area under the curve (AUC) of the receiver operator characteristic (ROC) curve and precision recall curve

(PRC), similar to classification tasks in MoleculeNet [133].

We believe that while not a complex idea, this benchmark could be extremely useful for the nascent DEL x ML field. As new computational methods for modeling DEL data, de-noising DEL counts and predicting activity from DEL data continue to emerge, having some degree of standardization would allow for more clear assessments of the advantages of different methods. In particular, we extensively outline the current publicly-available DEL datasets to demonstrate how sparse the landscape is. We assert that not only does this make it more difficult for new players to enter into the domain if they do not have an ability to generate experimental DEL data, but it also reduces the degree of collaboration currently present in the field. We hope that in the future, researchers will publish DEL datasets (or portions of them) for public use when introducing new computational techniques for analysis.

6.2 Incorporating 3D information to analyze DEL data

The concept of integrating 3D structural information for DEL prediction tasks was first introduced by Shmilovich et al. in 2023 [116]. In their work, the authors demonstrate how combining molecular-level descriptors with spatial information from docking resulted in better performance than models built from either method alone. They show how their approach could be used to de-noise DEL data and subsequently predict top scoring docked poses for sulfonamides, a known feature of many carbonic anhydrase inhibitors [43, 14].

We propose that docking could also be used to serve as a first pass filter for DEL screening efforts and be utilized to incorporate a sense of "chemical intuition" into existing ML models. While docking is not a reliable predictor of binding affinity [75, 60], it can be used to filter out obviously bad compounds prior to subsequent analysis [99, 50]. This could help address the tendency for ML models to overfit to noise in the data.

Within the context of chemical property prediction, an ML model overfitting to the data could have the effect where certain features that are correlated with the predicted property are over-emphasized [19]. For example, in the DEL-Dock paper [116], the authors evaluated their model on a subset of the data in a specific molecular weight range. They found that molecular weight was somewhat correlated with binding and without this filtering, models could more effectively differentiate binders from non-binders. We imagine a scenario in which this manual filtering of molecular weights was not performed. Should we train a model with no sense of 3D structural information on this data, the model may learn that higher molecular weight correlates with binding. Thus, when the model is used to predict on out-of-sample compounds, it may suggest that bulky compounds – which may not even fit into the binding pocket of the target – make for favorable binders.

Thus, we suggest exploring an approach where, rather than combining both molecular descriptors and docked pose information together to generate a prediction, the two could be run in sequence. This was not an approach that was explored in the DEL-Dock paper. This would allow docking to potentially serve as a "sanity check", potentially reducing the number of false positives from the ML predictions.

Bibliography

[1] M. J. Abraham, T. Murtola, R. Schulz, S. Páll, J. C. Smith, B. Hess, and E. Lindahl. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX*, 1-2:19–25, 2015.

[2] S. Ahmad, J. Xu, J. A. Feng, A. Hutchinson, H. Zeng, P. Ghiabi, A. Dong, P. A. Centrella, M. A. Clark, M.-A. Guié, J. P. Guilinger, A. D. Keefe, Y. Zhang, T. Cerruti, J. W. Cuozzo, M. von Rechenberg, A. Bolotokova, Y. Li, P. Loppnau, A. Seitova, Y.-Y. Li, V. Santhakumar, P. J. Brown, S. Ackloo, and L. Halabelian. Discovery of a first-in-class small-molecule ligand for WDR91 using DNA-encoded chemical library selection followed by machine learning. *J. Med. Chem.*, 66(23):16051–16061, 2023.

[3] M. Allaoui, M. L. Kherfi, and A. Cheriet. Considerably improving clustering algorithms using UMAP dimensionality reduction technique: A comparative study. In A. El Moataz, D. Mammass, A. Mansouri, and F. Nouboud, editors, *Image and Signal Processing*, Lecture Notes in Computer Science, pages 317–325. Springer International Publishing, 2020.

[4] J. Amigo, R. Rama-Garda, X. Bello, B. Sobrino, J. de Blas, M. Martín-Ortega, T. C. Jessop, Carracedo, M. I. G. Loza, and E. Domínguez. tagFinder: A novel tag analysis methodology that enables detection of molecules from DNA-encoded chemical libraries. *SLAS Discov*, 23(5):397–404, 2018.

[5] C. Arico-Muendel. From haystack to needle: finding value with DNA encoded library technology at GSK. *MedChemComm*, 7(10):1898–1909, 2016.

[6] E. Becht, L. McInnes, J. Healy, C.-A. Dutertre, I. W. H. Kwok, L. G. Ng, F. Ginhoux, and E. Newell. Dimensionality reduction for visualizing single-cell data using UMAP. *Nature Biotechnology*, 37:38–44, 2019.

[7] S. L. Belyanskaya, Y. Ding, J. F. Callahan, A. L. Lazaar, and D. I. Israel. Discovering drugs with DNA-encoded library technology: From concept to clinic with an inhibitor of soluble epoxide hydrolase. *ChemBioChem*, 18(9):837–842, 2017.

[8] G. W. Bemis and M. A. Murcko. The properties of known drugs. 1. molecular frameworks. *Journal of Medicinal Chemistry*, 39(15):2887–2893, 1996.

[9] H. Berendsen, D. van der Spoel, and R. van Drunen. GROMACS: A message-passing parallel molecular dynamics implementation. *Comput Phys Commun*, 91(1):43–56, 1995.

[10] P. Binder, M. Lawler, L. Grady, N. Carlson, S. Leelananda, S. Belyanskaya, J. Franklin, N. Tilmans, and H. Palacci. Partial product aware machine learning on DNA-encoded libraries. *arXiv*, (arXiv:2205.08020), 2022.

[11] R. S. Bohacek, C. McMartin, and W. C. Guida. The art and practice of structure-based drug design: A molecular modeling perspective. *Medicinal Research Reviews*, 16(1):3–50, 1996.

[12] S. Boothroyd, P. K. Behara, O. C. Madin, D. F. Hahn, H. Jang, V. Gapsys, J. R. Wagner, J. T. Horton, D. L. Dotson, M. W. Thompson, J. Maat, T. Gokey, L.-P. Wang, D. J. Cole, M. K. Gilson, J. D. Chodera, C. I. Bayly, M. R. Shirts, and D. L. Mobley. Development and benchmarking of open force field 2.0.0: The sage small molecule force field. *J. Chem. Theory Comput.*, 19(11):3251–3275, 2023.

[13] S. Brenner and R. Lerner. Encoded combinatorial chemistry. *Proc. Natl. Acad. Sci. USA*, 89:5381–5383, 1992.

[14] F. Buller, M. Steiner, K. Frey, D. Mircsof, J. Scheuermann, M. Kalisch, P. Bühlmann, C. T. Supuran, and D. Neri. Selection of carbonic anhydrase IX inhibitors from one million DNA-encoded compounds. *ACS Chem. Biol.*, 6(4):336–344, 2011.

[15] D. Butina. Unsupervised data base clustering based on daylight's fingerprint and tanimoto similarity: A fast and automated way to cluster small and large data sets. *Journal of Chemical Information and Computer Sciences*, 39(4):747–750, 1999.

[16] T. Caliński and J. Harabasz. A dendrite method for cluster analysis. *Communications in Statistics*, 3(1):1–27, 1974.

[17] R. J. G. B. Campello, D. Moulavi, and J. Sander. Density-based clustering based on hierarchical density estimates. In J. Pei, V. S. Tseng, L. Cao, H. Motoda, and G. Xu, editors, *Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science, pages 160–172. Springer, 2013.

[18] B. Chen, M. M. Sultan, and T. Karaletsos. Compositional deep probabilistic models of DNA-encoded libraries. *J. Chem. Inf. Model.*, 64(4):1123–1133, 2024.

[19] K. V. Chuang and M. J. Keiser. Comment on "predicting reaction performance in c–n cross-coupling using machine learning". *Science*, 362(6416):eaat8603, 2018.

[20] M. A. Clark, R. A. Acharya, C. C. Arico-Muendel, S. L. Belyanskaya, D. R. Benjamin, N. R. Carlson, P. A. Centrella, C. H. Chiu, S. P. Creaser, J. W. Cuozzo, C. P. Davie, Y. Ding, G. J. Franklin, K. D. Franzen, M. L. Gefter, S. P. Hale, N. J. V. Hansen, D. I. Israel, J. Jiang, M. J. Kavarana, M. S. Kelley, C. S. Kollmann, F. Li, K. Lind, S. Mataruse, P. F. Medeiros, J. A. Messer, P. Myers, H. O'Keefe, M. C. Oliff, C. E.

Rise, A. L. Satz, S. R. Skinner, J. L. Svendsen, L. Tang, K. van Vloten, R. W. Wagner, G. Yao, B. Zhao, and B. A. Morgan. Design, synthesis and selection of DNA-encoded small-molecule libraries. *Nature Chemical Biology*, 5(9):647–654, 2009.

[21] W. G. Cochrane, M. L. Malone, V. Q. Dang, V. Cavett, A. L. Satz, and B. M. Paegel. Activity-based DNA-encoded library screening. *ACS Comb. Sci.*, 21(5):425–435, 2019.

[22] C. W. Coley, R. Barzilay, W. H. Green, T. S. Jaakkola, and K. F. Jensen. Convolutional embedding of attributed molecular graphs for physical property prediction. *Journal of Chemical Information and Modeling*, 57(8):1757–1772, 2017.

[23] Z. Cournia, B. Allen, and W. Sherman. Relative binding free energy calculations in drug discovery: Recent advances and practical considerations. *Journal of Chemical Information and Modeling*, 57(12):2911–2937, 2017.

[24] czhang475. MobleyLab/DEL_analysis: Added input data, 2023.

[25] D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227, 1979.

[26] J. Davis and M. Goadrich. The relationship between precision-recall and ROC curves. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 233–240. Association for Computing Machinery, 2006.

[27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv*, (arXiv:1810.04805), 2019.

[28] L. Di, C. Whitney-Pickett, J. P. Umland, H. Zhang, X. Zhang, D. F. Gebhard, Y. Lai, J. J. Federico, R. E. Davidson, R. Smith, E. L. Reyner, C. Lee, B. Feng, C. Rotter, M. V. Varma, S. Kempshall, K. Fenner, A. F. El-kattan, T. E. Liston, and M. D. Troutman. Development of a new permeability assay using low-efflux MDCKII cells. *Journal of Pharmaceutical Sciences*, 100(11):4974–4985, 2011.

[29] J. A. DiMasi, H. G. Grabowski, and R. W. Hansen. Innovation in the pharmaceutical industry: New estimates of r&d costs. *Journal of Health Economics*, 47:20–33, 2016.

[30] X. Ding, R. Cui, J. Yu, T. Liu, T. Zhu, D. Wang, J. Chang, Z. Fan, X. Liu, K. Chen, H. Jiang, X. Li, X. Luo, and M. Zheng. Active learning for drug design: A case study on the plasma exposure of orally administered drugs. *Journal of Medicinal Chemistry*, 64(22):16838–16853, 2021.

[31] Y. Ding, S. Belyanskaya, J. L. DeLorey, J. A. Messer, G. Joseph Franklin, P. A. Centrella, B. A. Morgan, M. A. Clark, S. R. Skinner, J. W. Dodson, P. Li, J. P. Marino, and D. I. Israel. Discovery of soluble epoxide hydrolase inhibitors through DNA-encoded library technology (ELT). *Bioorganic & Medicinal Chemistry*, 41:116216, 2021.

[32] A. Dixit, H. Barhoosh, and B. M. Paegel. Translating the genome into drugs. *Accounts of Chemical Research*, 56(4):489–499, 2023.

[33] J. D. Durrant and J. A. McCammon. Molecular dynamics simulations and drug discovery. *BMC Biology*, 9(1):71, 2011.

[34] A. B. Eldrup, F. Soleymanzadeh, S. J. Taylor, I. Muegge, N. A. Farrow, D. Joseph, K. McKellop, C. C. Man, A. Kukulka, and S. De Lombaert. Structure-based optimization of arylamides as inhibitors of soluble epoxide hydrolase. *Journal of Medicinal Chemistry*, 52(19):5880–5895, 2009.

[35] Enamine. Real database: https://enamine.net/compound-collections/real-compounds/real-database Accessed 01/18/2024, 2024.

[36] P. Ertl, E. Altmann, and J. M. McKenna. The most common functional groups in bioactive molecules and how their popularity has evolved over time. *J. Med. Chem.*, 63(15):8408–8418, 2020.

[37] M. Feurer and F. Hutter. Hyperparameter optimization. In F. Hutter, L. Kotthoff, and J. Vanschoren, editors, *Automated Machine Learning: Methods, Systems, Challenges*, The Springer Series on Challenges in Machine Learning, pages 3–33. Springer International Publishing, 2019.

[38] P. R. Fitzgerald, W. G. Cochrane, and B. M. Paegel. Dose–response activity-based DNA-encoded library screening. *ACS Med. Chem. Lett.*, 14(9):1295–1303, 2023.

[39] P. R. Fitzgerald and B. M. Paegel. DNA-encoded chemistry: Drug discovery from a few good reactions. *Chemical Reviews*, 121(12):7155–7177, 2021.

[40] R. M. Franzini, D. Neri, and J. Scheuermann. DNA-encoded chemical libraries: Advancing beyond conventional small-molecule libraries. *Accounts of Chemical Research*, 47(4):1247–1255, 2014.

[41] R. M. Franzini and C. Randolph. Chemical space of DNA-encoded libraries. *J. Med. Chem.*, 59(14):6629–6644, 2016.

[42] Y. Ge, B. L. Kier, N. H. Andersen, and V. A. Voelz. Computational and experimental evaluation of designed -cap hairpins using molecular simulations and kinetic network models. *Journal of Chemical Information and Modeling*, 57(7):1609–1620, 2017.

[43] C. J. Gerry, M. J. Wawer, P. A. Clemons, and S. L. Schreiber. DNA barcoding a complete matrix of stereoisomeric small molecules. *Journal of the American Chemical Society*, 141(26):10225–10235, 2019.

[44] A. Gironda-Martínez, E. J. Donckele, F. Samain, and D. Neri. DNA-encoded chemical libraries: A comprehensive review with succesful stories and future challenges. *ACS Pharmacology & Translational Science*, 4(4):1265–1279, 2021.

[45] R. A. Goodnow, C. E. Dumelin, and A. D. Keefe. DNA-encoded chemistry: enabling the deeper sampling of chemical space. *Nature Reviews Drug Discovery*, 16(2):131–147, 2017.

[46] C. Gorgulla, A. Boeszoermenyi, Z.-F. Wang, P. D. Fischer, P. Coote, K. M. P. Das, Y. S. Malets, D. S. Radchenko, Y. S. Moroz, D. A. Scott, K. Fackeldey, M. Hoffmann, I. Iavniuk, G. Wagner, and H. Arthanari. An open-source drug discovery platform enables ultra-large virtual screens. *Nature*, 580(7805):663–668, 2020.

[47] D. E. Graff, M. Aldeghi, J. A. Morrone, K. E. Jordan, E. O. Pyzer-Knapp, and C. W. Coley. Self-focusing virtual screening with active design space pruning. *Journal of Chemical Information and Modeling*, 62(16):3854–3862, 2022.

[48] D. E. Graff, E. I. Shakhnovich, and C. W. Coley. Accelerating high-throughput virtual screening through molecular pool-based active learning. *Chemical Science*, 12(22):7866–7881, 2021.

[49] F. Gusev, E. Gutkin, M. G. Kurnikova, and O. Isayev. Active learning guided drug design lead optimization based on relative binding free energy modeling. *Journal of Chemical Information and Modeling*, 63(2):583–594, 2023.

[50] E. J. Ha, C. T. Lwin, and J. D. Durrant. LigGrep: a tool for filtering docked poses to improve virtual-screening hit rates. *J Cheminform*, 12(1):69, 2020.

[51] A. L. Hackler, F. G. FitzGerald, V. Q. Dang, A. L. Satz, and B. M. Paegel. Off-DNA DNA-encoded library affinity screening. *ACS Comb. Sci.*, 22(1):25–34, 2020.

[52] I. S. Haque, K. A. Beauchamp, and V. S. Pande. A fast 3 × n matrix multiply routine for calculation of protein RMSD, 2014.

[53] I. V. Hartung, B. R. Huck, and A. Crespo. Rules were made to be broken. *Nat Rev Chem*, 7(1):3–4, 2023.

[54] P. C. D. Hawkins, A. G. Skillman, and A. Nicholls. Comparison of shape-matching and docking as virtual screening tools. *Journal of Medicinal Chemistry*, 50(1):74–82, 2007.

[55] E. Heid, K. P. Greenman, Y. Chung, S.-C. Li, D. E. Graff, F. H. Vermeire, H. Wu, W. H. Green, and C. J. McGill. Chemprop: A machine learning package for chemical property prediction. *J. Chem. Inf. Model.*, 64(1):9–17, 2024.

[56] M. Hoffmann, M. Scherer, T. Hempel, A. Mardt, B. d. Silva, B. E. Husic, S. Klus, H. Wu, N. Kutz, S. L. Brunton, and F. Noé. Deeptime: a python library for machine learning dynamical models from time series data. *Mach. Learn.: Sci. Technol.*, 3(1):015009, 2021.

[57] J. Hughes, S. Rees, S. Kalindjian, and K. Philpott. Principles of early drug discovery. *British Journal of Pharmacology*, 162(6):1239–1249, 2011.

[58] B. E. Husic and V. S. Pande. Markov state models: From an art to a science. *Journal of the American Chemical Society*, 140(7):2386–2396, 2018.

[59] M. A. Johnson and G. M. Maggiora. *Concepts and applications of molecular similarity.* Wiley, 1990.

[60] W. L. Jorgensen. The many roles of computation in drug discovery. *Science*, 303(5665):1813–1818, 2004.

[61] L. K. Petersen, P. Blakskjær, A. Chaikuad, A. B. Christensen, J. Dietvorst, J. Holmkvist, S. Knapp, M. Kořínek, L. K. Larsen, A. E. Pedersen, S. Röhm, F. A. Sløk, and N. J. V. Hansen. Novel p38 MAP kinase inhibitors identified from yoctoReactor DNA-encoded small molecule library. *MedChemComm*, 7(7):1332–1339, 2016.

[62] T. Kalliokoski. Machine learning boosted docking (HASTEN): An open-source tool to accelerate structure-based virtual screening campaigns. *Molecular Informatics*, 40(9):2100089, 2021.

[63] H. Kang, S. Goo, H. Lee, J.-w. Chae, H.-y. Yun, and S. Jung. Fine-tuning of BERT model to accurately predict drug–target interactions. *Pharmaceutics*, 14(8):1710, 2022.

[64] S. Kearnes, K. McCloskey, M. Berndl, V. Pande, and P. Riley. Molecular graph convolutions: moving beyond fingerprints. *Journal of Computer-Aided Molecular Design*, 30(8):595–608, 2016.

[65] Y. Khalak, G. Tresadern, D. F. Hahn, B. L. de Groot, and V. Gapsys. Chemical space exploration with active learning and alchemical free energies. *Journal of Chemical Theory and Computation*, 18(10):6259–6270, 2022.

[66] L. Kuai, T. O'Keeffe, and C. Arico-Muendel. Randomness in DNA encoded library selection data can be modeled for more reliable enrichment calculation. *SLAS DISCOVERY: Advancing the Science of Drug Discovery*, 23(5):405–416, 2018.

[67] K. Kumar, V. Chupakhin, A. Vos, D. Morrison, D. Rassokhin, M. J. Dellwo, K. McCormick, E. Paternoster, H. Ceulemans, and R. L. DesJarlais. Development and implementation of an enterprise-wide predictive model for early absorption, distribution, metabolism and excretion properties. *Future Medicinal Chemistry*, 13(19):1639–1654, 2021.

[68] S. G. Kwak and J. H. Kim. Central limit theorem: the cornerstone of modern statistics. *Korean Journal of Anesthesiology*, 70(2):144–156, 2017.

[69] P. Kómár and M. Kalinić. Denoising DNA encoded library screens with sparse learning. *ACS Combinatorial Science*, 22(8):410–421, 2020.

[70] G. Landrum. RDKit: Open-source cheminformatics, 2020.

[71] S. P. Leelananda and S. Lindert. Computational methods in drug discovery. *Beilstein J. Org. Chem.*, 12(1):2694–2718, 2016.

[72] K. S. Lim, A. G. Reidenbach, B. K. Hua, J. W. Mason, C. J. Gerry, P. A. Clemons, and C. W. Coley. Machine learning on DNA-encoded library count data using an uncertainty-aware probabilistic loss function. *Journal of Chemical Information and Modeling*, 62(10):2316–2331, 2022.

[73] N. M. Lim, L. Wang, R. Abel, and D. L. Mobley. Sensitivity in binding free energies due to protein reorganization. *Journal of Chemical Theory and Computation*, 12(9):4620–4631, 2016.

[74] C. A. Lipinski, F. Lombardo, B. W. Dominy, and P. J. Feeney. Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Advanced Drug Delivery Reviews*, 23(1):3–25, 1997.

[75] J. Lyu, S. Wang, T. E. Balius, I. Singh, A. Levit, Y. S. Moroz, M. J. O'Meara, T. Che, E. Algaa, K. Tolmachova, A. A. Tolmachev, B. K. Shoichet, B. L. Roth, and J. J. Irwin. Ultra-large library docking for discovering new chemotypes. *Nature*, 566(7743):224–229, 2019.

[76] R. Ma, G. H. S. Dreiman, F. Ruggiu, A. J. Riesselman, B. Liu, K. James, M. Sultan, and D. Koller. Regression modeling on DNA encoded libraries. *NeurIPS 2021 AI for Science Workshop*, 2021.

[77] A. B. MacConnell and B. M. Paegel. Poisson statistics of combinatorial library sampling predict false discovery rates of screening. *ACS Comb. Sci.*, 19(8):524–532, 2017.

[78] J. A. Maier, C. Martinez, K. Kasavajhala, L. Wickstrom, K. E. Hauser, and C. Simmerling. ff14sb: Improving the accuracy of protein side chain and backbone parameters from ff99sb. *J Chem Theory Comput*, 11(8):3696–3713, 2015.

[79] L. Mannocci, Y. Zhang, J. Scheuermann, M. Leimbacher, G. De Bellis, E. Rizzi, C. Dumelin, S. Melkko, and D. Neri. High-throughput sequencing allows the identification of binding molecules isolated from DNA-encoded chemical libraries. *Proc. Natl. Acad. Sci USA*, 105(46):17670–17675, 2008.

[80] Y. C. Martin, J. L. Kofron, and L. M. Traphagen. Do structurally similar molecules have similar biological activity? *Journal of Medicinal Chemistry*, 45(19):4350–4358, 2002-09-01.

[81] A. Martín, C. A. Nicolaou, and M. A. Toledo. Navigating the DNA encoded libraries chemical space. *Communications Chemistry*, 3(1):1–9, 2020.

[82] K. McCloskey, E. A. Sigel, S. Kearnes, L. Xue, X. Tian, D. Moccia, D. Gikunju, S. Bazzaz, B. Chan, M. A. Clark, J. W. Cuozzo, M.-A. Guié, J. P. Guilinger, C. Huguet, C. D. Hupp, A. D. Keefe, C. J. Mulhern, Y. Zhang, and P. Riley. Machine learning on DNA-encoded libraries: A new paradigm for hit finding. *Journal of Medicinal Chemistry*, 63(16):8857–8866, 2020.

[83] R. T. McGibbon, K. A. Beauchamp, M. P. Harrigan, C. Klein, J. M. Swails, C. X. Hernández, C. R. Schwantes, L.-P. Wang, T. J. Lane, and V. S. Pande. MDTraj: A modern open library for the analysis of molecular dynamics trajectories. *Biophysical Journal*, 109(8):1528–1532, 2015.

[84] L. McInnes, J. Healy, and S. Astels. hdbscan: Hierarchical density based clustering. *Journal of Open Source Software*, 2(11):205, 2017.

[85] L. McInnes, J. Healy, and J. Melville. UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv*, (arXiv:1802.03426), 2020.

[86] L. McInnes, J. Healy, N. Saul, and L. Großberger. UMAP: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861, 2018.

[87] M. Merski, M. Fischer, T. E. Balius, O. Eidam, and B. K. Shoichet. Homologous ligands accommoyeard by discrete conformations of a buried cavity. *Proceedings of the National Academy of Sciences*, 112(16):5039–5044, 2015.

[88] R. Minkwitz and M. Meldal. Application of a photolabile backbone amide linker for cleavage of internal amides in the synthesis towards melanocortin subtype-4 agonists. *QSAR & Combinatorial Science*, 24(3):343–353, 2005.

[89] D. L. Mobley and K. A. Dill. Binding of small-molecule ligands to proteins: "what you see" is not always "what you get". *Structure*, 17(4):489–498, 2009.

[90] D. L. Mobley, A. P. Graves, J. D. Chodera, A. C. McReynolds, B. K. Shoichet, and K. A. Dill. Predicting absolute ligand binding free energies to a simple model site. *Journal of Molecular Biology*, 371(4):1118–1134, 2007.

[91] D. L. Mobley and P. V. Klimovich. Perspective: Alchemical free energy calculations for drug discovery. *The Journal of Chemical Physics*, 137(23):230901, 2012.

[92] J. Mondal, N. Ahalawat, S. Pandit, L. E. Kay, and P. Vallurupalli. Atomic resolution mechanism of ligand binding to a solvent inaccessible cavity in t4 lysozyme. *PLOS Computational Biology*, 14(5):e1006180, 2018.

[93] 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.

[94] S. W. Muchmore, D. A. Debe, J. T. Metz, S. P. Brown, Y. C. Martin, and P. J. Hajduk. Application of belief theory to similarity data fusion for use in analog searching and lead hopping. *Journal of Chemical Information and Modeling*, 48(5):941–948, 2008.

[95] A. Nicholls, G. B. McGaughey, R. P. Sheridan, A. C. Good, G. Warren, M. Mathieu, S. W. Muchmore, S. P. Brown, J. A. Grant, J. A. Haigh, N. Nevins, A. N. Jain, and B. Kelley. Molecular shape and medicinal chemistry: A perspective. *Journal of Medicinal Chemistry*, 53(10):3862–3886, 2010.

[96] J. W. M. Nissink, S. Bazzaz, C. Blackett, M. A. Clark, O. Collingwood, J. S. Disch, D. Gikunju, K. Goldberg, J. P. Guilinger, E. Hardaker, E. J. Hennessy, R. Jetson, A. D. Keefe, W. McCoull, L. McMurray, A. Olszewski, R. Overman, A. Pflug, M. Preston, P. B. Rawlins, E. Rivers, M. Schimpl, P. Smith, C. Truman, E. Underwood, J. Warwicker, J. Winter-Holt, S. Woodcock, and Y. Zhang. Generating selective leads for mer kinase inhibitors—example of a comprehensive lead-generation strategy. *J. Med. Chem.*, 64(6):3165–3184, 2021.

[97] OpenEye. OpenEye Toolkits 2021 OpenEye Scientific Software, Santa Fe, NM. http://www.eyesopen.com.

[98] B. Ozenne, F. Subtil, and D. Maucort-Boulch. The precision–recall curve overcame the optimism of the receiver operating characteristic curve in rare diseases. *Journal of Clinical Epidemiology*, 68(8):855–859, 2015.

[99] M. L. Peach and M. C. Nicklaus. Combining docking with pharmacophore filtering for improved virtual screening. *J Cheminform*, 1:6, 2009.

[100] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[101] B. Ramsundar, P. Eastman, P. Walters, V. Pande, K. Leswing, and Z. Wu. *Deep learning for the life sciences.* O'Reilly Media, 2019.

[102] J. Reback and W. McKinney. pandas-dev/pandas: Pandas 1.2.1, 2021.

[103] D. Reker and G. Schneider. Active-learning strategies in computer-assisted drug discovery. *Drug Discovery Today*, 20(4):458–465, 2015.

[104] J.-L. Reymond and M. Awale. Exploring chemical space for drug discovery using the chemical universe database. *ACS Chemical Neuroscience*, 3(9):649–657, 2012.

[105] D. Rogers and M. Hahn. Extended-connectivity fingerprints. *J Chem Inf Model*, 50(5):742–754, 2010.

[106] G. A. Ross, G. M. Morris, and P. C. Biggin. One size does not fit all: The limits of structure-based models in drug discovery. *J. Chem. Theory Comput.*, 9(9):4266–4274, 2013.

[107] P. J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.

[108] M. D. Ryan, A. L. Parkes, D. Corbett, A. P. Dickie, M. Southey, O. A. Andersen, D. B. Stein, O. R. Barbeau, A. Sanzone, P. Thommes, J. Barker, R. Cain, C. Compper, M. Dejob, A. Dorali, D. Etheridge, S. Evans, A. Faulkner, E. Gadouleau, T. Gorman, D. Haase, M. Holbrow-Wilshaw, T. Krulle, X. Li, C. Lumley, B. Mertins, S. Napier, R. Odedra, K. Papadopoulos, V. Roumpelakis, K. Spear, E. Trimby, J. Williams,

M. Zahn, A. D. Keefe, Y. Zhang, H. T. Soutter, P. A. Centrella, M. A. Clark, J. W. Cuozzo, C. E. Dumelin, B. Deng, A. Hunt, E. A. Sigel, D. M. Troast, and B. L. M. DeJonge. Discovery of novel UDP-n-acetylglucosamine acyltransferase (LpxA) inhibitors with activity against pseudomonas aeruginosa. *J. Med. Chem.*, 64(19):14377–14425, 2021.

[109] T. Sander, J. Freyss, M. von Korff, and C. Rufener. DataWarrior: An open-source program for chemistry aware data visualization and analysis. *J. Chem. Inf. Model.*, 55(2):460–473, 2015.

[110] A. L. Satz. DNA encoded library selections and insights provided by computational simulations. *ACS Chemical Biology*, 10(10):2237–2245, 2015.

[111] A. L. Satz. Simulated screens of DNA encoded libraries: The potential influence of chemical synthesis fidelity on interpretation of structure–activity relationships. *ACS Combinatorial Science*, 18(7):415–424, 2016.

[112] A. L. Satz, R. Hochstrasser, and A. C. Petersen. Analysis of current DNA encoded library screening data indicates higher false negative rates for numerically larger libraries. *ACS Combinatorial Science*, 19(4):234–238, 2017.

[113] M. K. Scherer, B. Trendelkamp-Schroer, F. Paul, G. Pérez-Hernández, M. Hoffmann, N. Plattner, C. Wehmeyer, J.-H. Prinz, and F. Noé. PyEMMA 2: A software package for estimation, validation, and analysis of markov models. *Journal of Chemical Theory and Computation*, 11(11):5525–5542, 2015.

[114] G. Schneider. Automating drug discovery. *Nat Rev Drug Discov*, 17(2):97–113, 2018.

[115] Schrödinger. Evaluating large ligand libraries with active learning glide https://www.schrodinger.com/training/, 2023.

[116] K. Shmilovich, B. Chen, T. Karaletsos, and M. M. Sultan. DEL-dock: Molecular docking-enabled modeling of DNA-encoded libraries. *Journal of Chemical Information and Modeling*, 63(9):2719–2727, 2023.

[117] T. Sivula, L. Yetukuri, T. Kalliokoski, H. Käsnänen, A. Poso, and I. Pöhner. Machine learning-boosted docking enables the efficient structure-based virtual screening of giga-scale enumerated chemical libraries. *J. Chem. Inf. Model.*, 63(18):5773–5783, 2023.

[118] J. S. Smith, B. Nebgen, N. Lubbers, O. Isayev, and A. E. Roitberg. Less is more: Sampling chemical space with active learning. *The Journal of Chemical Physics*, 148(24):241733, 2018.

[119] R. R. Stoll. *Set Theory and Logic.* Courier Corporation, 1979.

[120] W. Su, R. Ge, D. Ding, W. Chen, W. Wang, H. Yan, W. Wang, Y. Yuan, H. Liu, M. Zhang, J. Zhang, Q. Shu, A. L. Satz, and L. Kuai. Triaging of DNA-encoded library selection results by high-throughput resynthesis of DNA–conjugate and affinity selection mass spectrometry. *Bioconjugate Chemistry*, 32(5):1001–1007, 2021.

[121] J. Sun, L. Carlsson, E. Ahlberg, U. Norinder, O. Engkvist, and H. Chen. Applying mondrian cross-conformal prediction to estimate prediction confidence on large imbalanced bioactivity data sets. *Journal of Chemical Information and Modeling*, 57(7):1591–1598, 2017.

[122] D. L. Theobald. Rapid calculation of RMSDs using a quaternion-based characteristic polynomial. *Acta Crystallographica Section A*, 61(4):478–480, 2005.

[123] J. Thompson, W. P. Walters, J. A. Feng, N. A. Pabon, H. Xu, M. Maser, B. B. Goldman, D. Moustakas, M. Schmidt, and F. York. Optimizing active learning for free energy calculations. *Artificial Intelligence in the Life Sciences*, 2:100050, 2022.

[124] A. Tomberg and J. Boström. Can easy chemistry produce complex, diverse, and novel molecules? *Drug Discovery Today*, 25(12):2174–2181, 2020.

[125] D. Van Tilborg and F. Grisoni. Traversing chemical space with active deep learning: A computational framework for low-data drug discovery. *ChemRxiv*, 2024.

[126] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *arXiv*, (arXiv:1706.03762), 2023.

[127] D. F. Veber, S. R. Johnson, H.-Y. Cheng, B. R. Smith, K. W. Ward, and K. D. Kopple. Molecular properties that influence the oral bioavailability of drug candiyears. *Journal of Medicinal Chemistry*, 45(12):2615–2623, 2002.

[128] J. J. N. Veerman, Y. B. Bruseker, E. Damen, E. H. Heijne, W. van Bruggen, K. F. W. Hekking, R. Winkel, C. D. Hupp, A. D. Keefe, J. Liu, H. A. Thomson, Y. Zhang, J. W. Cuozzo, A. J. McRiner, M. J. Mulvihill, P. van Rijnsbergen, B. Zech, L. M. Renzetti, L. Babiss, and G. Müller. Discovery of 2,4-1h-imidazole carboxamides as potent and selective TAK1 inhibitors. *ACS Med. Chem. Lett.*, 12(4):555–562, 2021.

[129] M. K. Warmuth, J. Liao, G. Rätsch, M. Mathieson, S. Putta, and C. Lemmen. Active learning with support vector machines in the drug discovery process. *Journal of Chemical Information and Computer Sciences*, 43(2):667–673, 2003.

[130] E. W. Weisstein. Full width at half maximum https://mathworld.wolfram.com/fullwidthathalfmaximum.html.

[131] Wes McKinney. Data structures for statistical computing in python. In S. van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010.

[132] S. A. Wildman and G. M. Crippen. Prediction of physicochemical parameters by atomic contributions. *Journal of Chemical Information and Computer Sciences*, 39(5):868–873, 1999.

[133] Z. Wu, B. Ramsundar, E. N. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing, and V. Pande. MoleculeNet: a benchmark for molecular machine learning. *Chemical Science*, 9(2):513–530, 2018.

[134] K. Yang, K. Swanson, W. Jin, C. Coley, P. Eiden, H. Gao, A. Guzman-Perez, T. Hopper, B. Kelley, M. Mathea, A. Palmer, V. Settels, T. Jaakkola, K. Jensen, and R. Barzilay. Analyzing learned molecular representations for property prediction. *Journal of Chemical Information and Modeling*, 59(8):3370–3388, 2019-08-26.

[135] Y. Yang, K. Yao, M. P. Repasky, K. Leswing, R. Abel, B. K. Shoichet, and S. V. Jerome. Efficient exploration of chemical space with docking and deep learning. *Journal of Chemical Theory and Computation*, 17(11):7106–7119, 2021.

[136] J. Yu, X. Li, and M. Zheng. Current status of active learning for drug discovery. *Artificial Intelligence in the Life Sciences*, 1:100023, 2021.

[137] L. H. Yuen and R. M. Franzini. Achievements, challenges, and opportunities in DNA-encoded library research: An academic point of view. *ChemBioChem*, 18(9):829–836, 2017.

[138] Y. Zabolotna, D. M. Volochnyuk, S. V. Ryabukhin, D. Horvath, K. S. Gavrilenko, G. Marcou, Y. S. Moroz, O. Oksiuta, and A. Varnek. A close-up look at the chemical space of commercially available building blocks for medicinal chemistry. *J. Chem. Inf. Model.*, 62(9):2171–2185, 2022.

[139] C. Zhang, M. Pitman, A. Dixit, S. Leelananda, H. Palacci, M. Lawler, S. Belyanskaya, L. Grady, J. Franklin, N. Tilmans, and D. L. Mobley. Building block-based binding predictions for DNA-encoded libraries. *Journal of Chemical Information and Modeling*, 63(16):5120–5132, 2023.

[140] H. Zhu, T. L. Foley, J. I. Montgomery, and R. V. Stanton. Understanding data noise and uncertainty through analysis of replicate samples in DNA-encoded library selection. *J. Chem. Inf. Model.*, 62(9):2239–2247, 2022.

Appendix A

Supporting Information: Characterizing Discrete Binding Conformations of T4 L99A via Markov State Modeling

A.1 Additional Results

A.1.1 Implied Timescales Plots for all systems

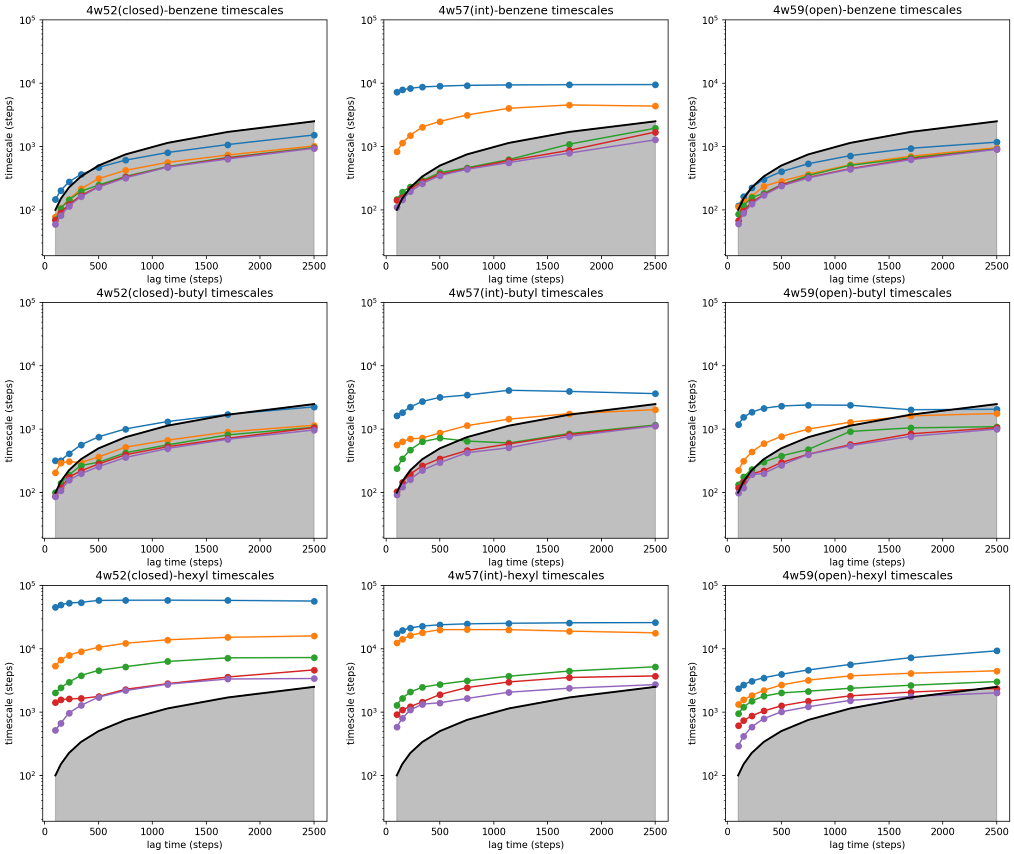


Figure A.1: Implied timescales plots for all systems.

A.1.2 Slowest motions for benzene-bound systems

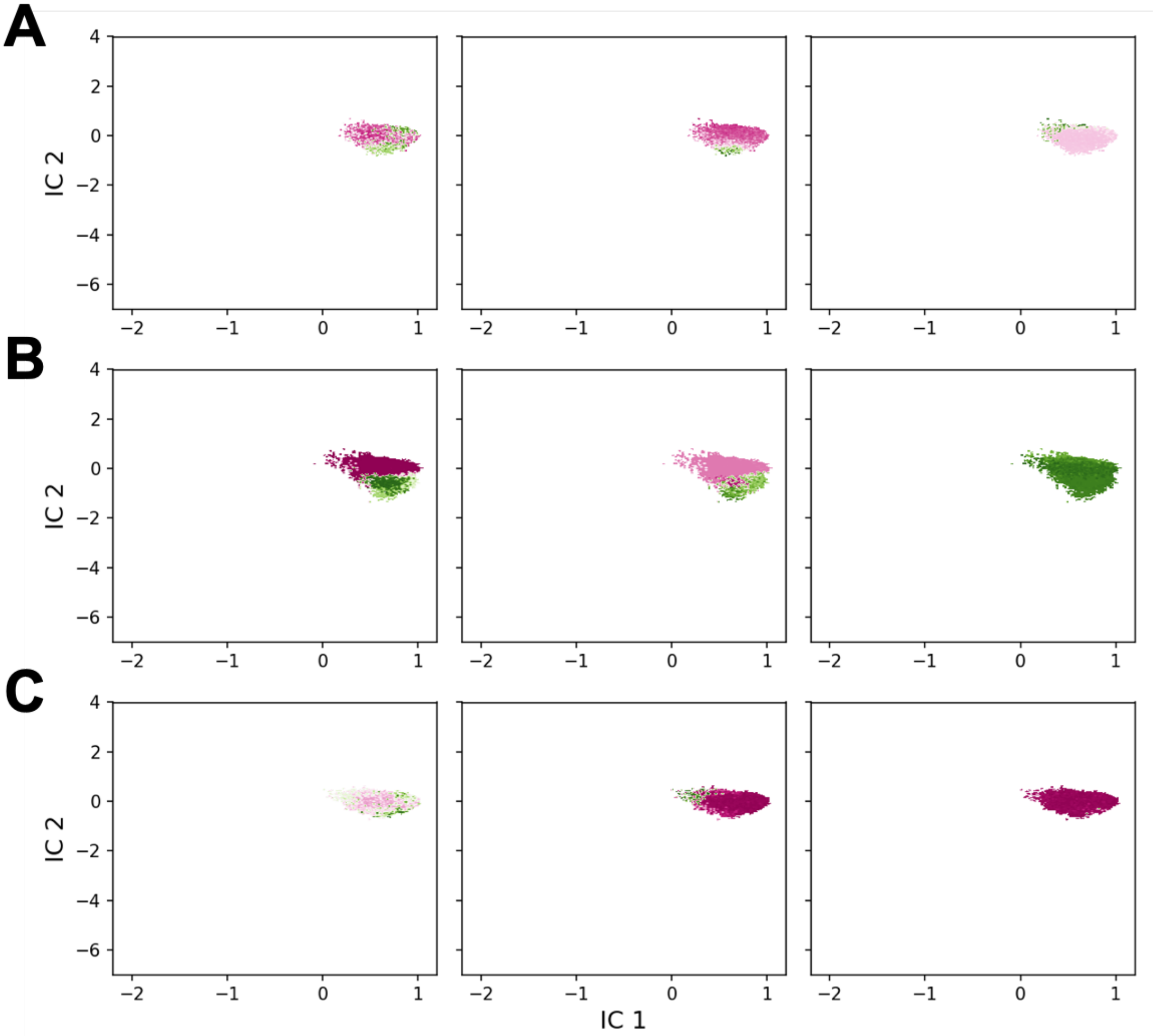


Figure A.2: Top three slowest motions for all benzene-bound systems. Shown are the slowest motions for (A) 4w52–benzene, (B) 4w57–benzene and (C) 4w59–benzene. Darker pink indicates flux out, darker green indicates flux in.

A.1.3 Slowest motions for butylbenzene-bound systems

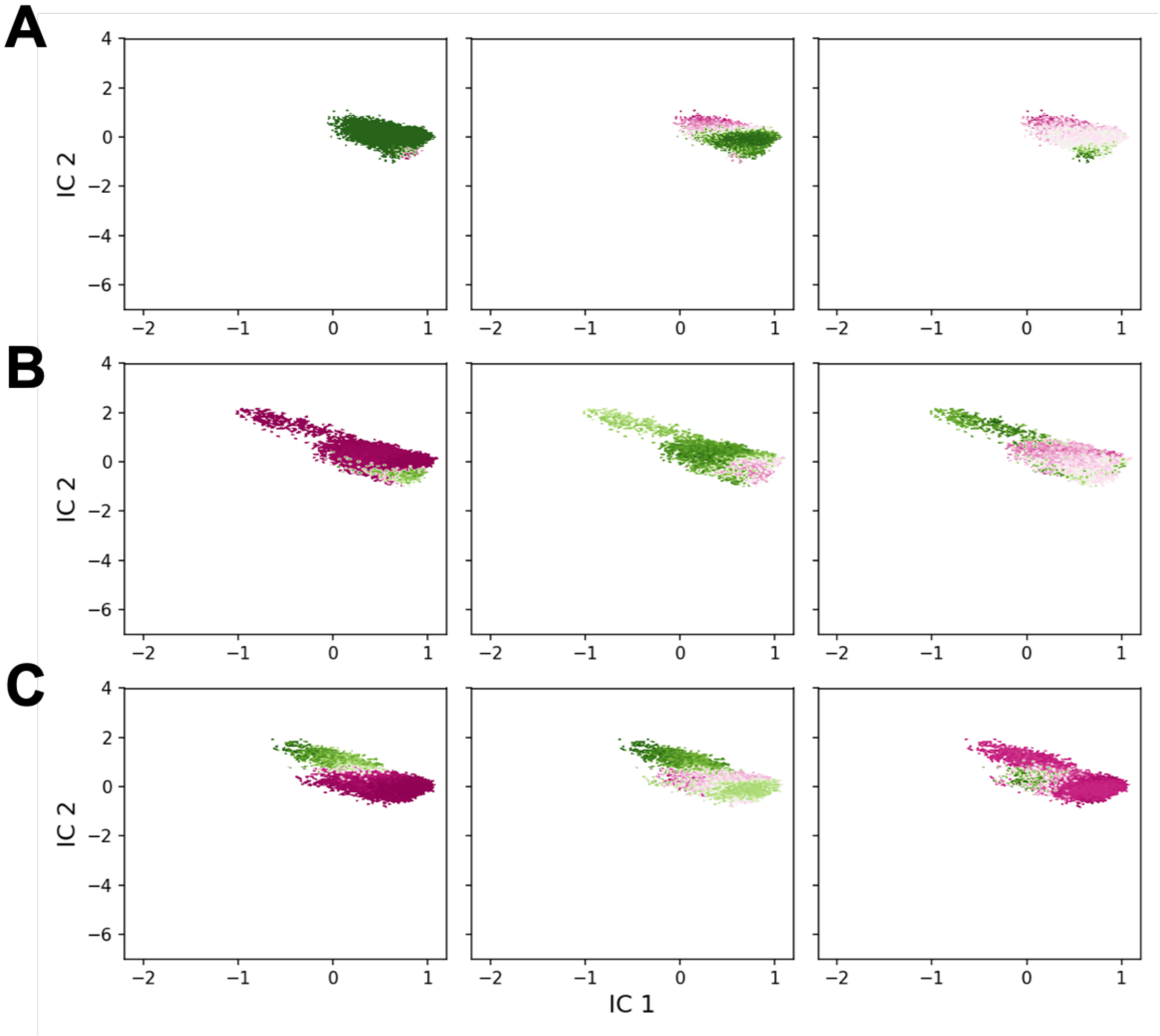


Figure A.3: Top three slowest motions for all butylbenzene-bound systems. Shown are the slowest motions for (A) 4w52–butylbenzene, (B) 4w57–butylbenzene and (C) 4w59–butylbenzene. Darker pink indicates flux out, darker green indicates flux in.

A.1.4 Slowest motions for hexylbenzene-bound systems

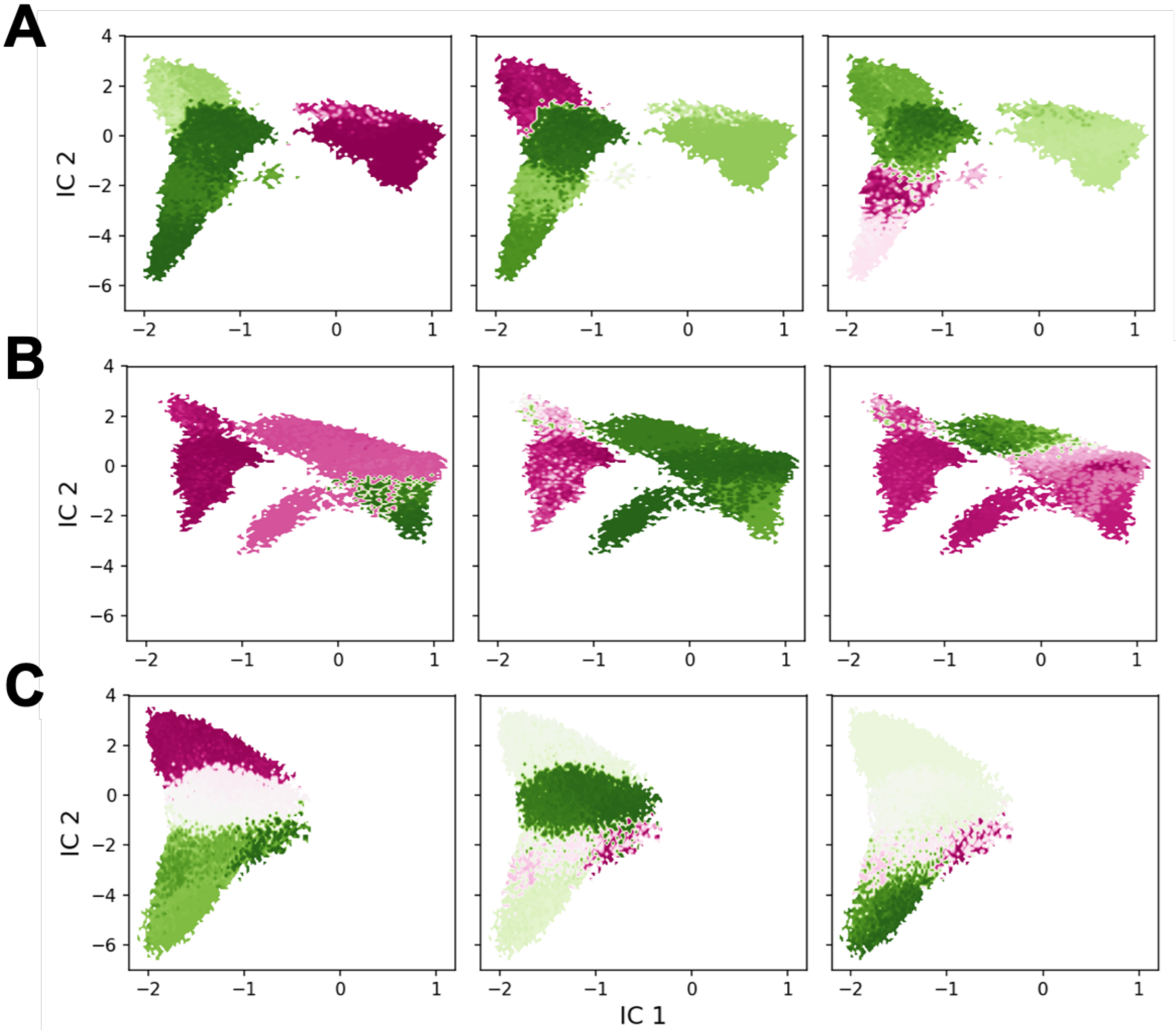


Figure A.4: Top three slowest motions for all hexylbenzene-bound systems. Shown are the slowest motions for (A) 4w52–hexylbenzene, (B) 4w57–hexylbenzene and (C) 4w59–hexylbenzene. Darker pink indicates flux out, darker green indicates flux in.

Appendix B

Supporting Information: Building Block-Based Binding Predictions for DNA-Encoded Libraries

B.1 Additional Results

B.1.1 Effect of Sampling on Calculation of P(bind) metric

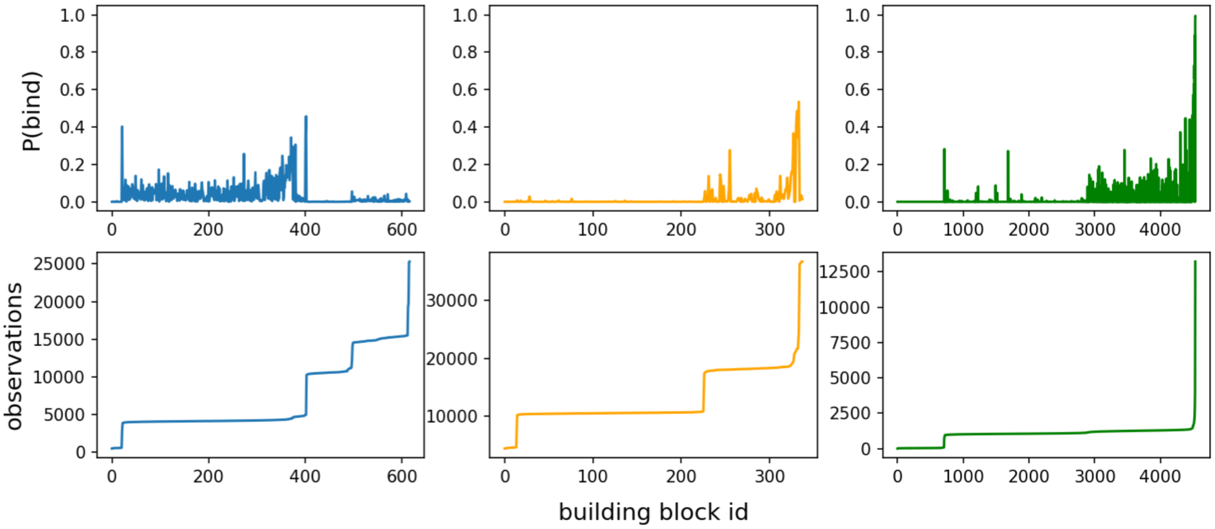


Figure B.1: P(bind) value and number of observations for each building block in the library. With the exception of 10 building blocks in position 3, every building block occurs in a statistically significant number of compounds (N>30) [68].

B.1.2 Structural Similarities between Productive Fragments and Known sEH Inhibitors

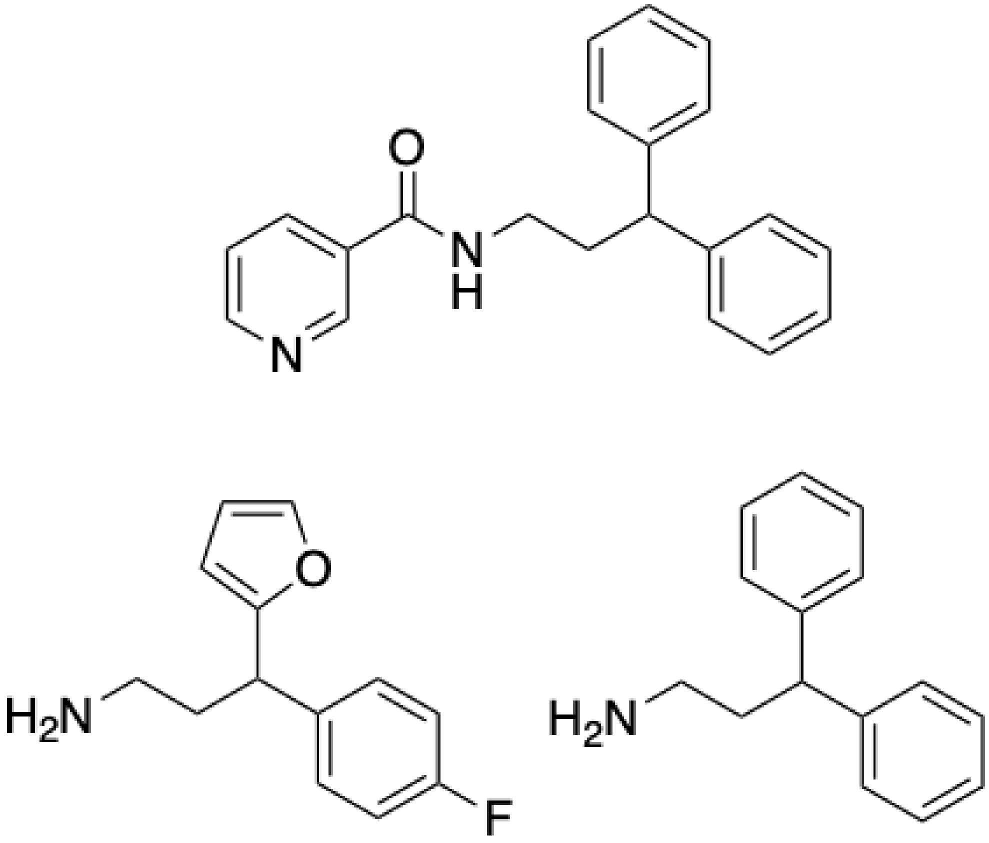


Figure B.2: Structure of a known potent inhibitor of sEH from experiment [34] (top) and the structures of the top two most productive building blocks in position 3 identified through P(bind) analysis (bottom).

B.1.3 Physicochemical Properties of Productive Building Blocks

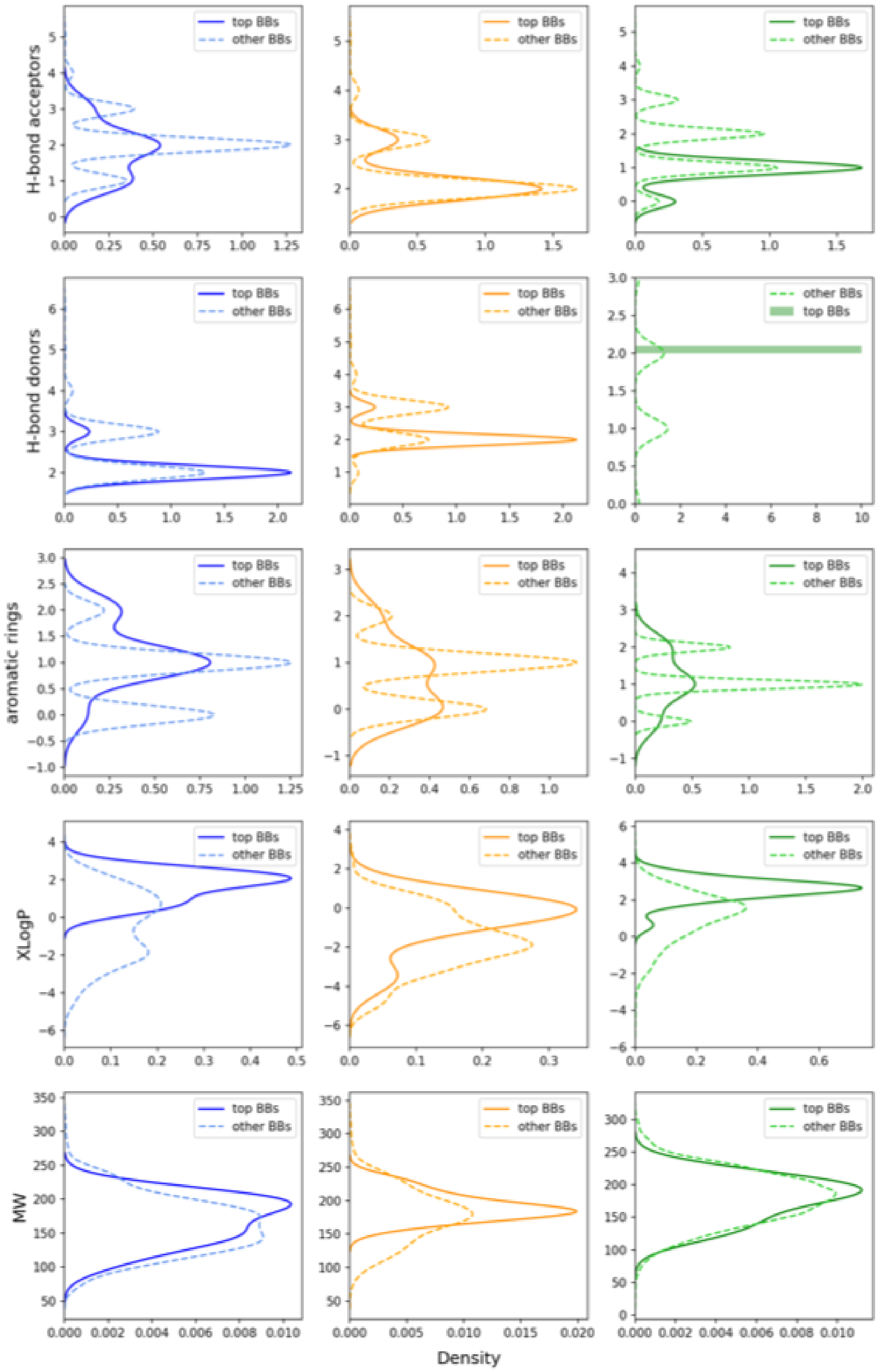


Figure B.3: Distribution of physicochemical properties for top building blocks by P(bind) value to all other building blocks at each position. We defined the top building blocks at each position as the top 20 by P(bind) value.

B.1.4 Tables of values for pairwise analysis of building block bins

p_2 P(bind) bin	p_1 P(bind) bin		
	$[0.00, 0.20)$	$[0.20, 0.40)$	$[0.40, 0.60)$
$[0.40, 0.60)$	0.4452	0.9451	0.9771
$[0.20, 0.40)$	0.2946	0.9104	0.9697
$[0.00, 0.20)$	0.0100	0.1706	0.2930

Table B.1: Matrix entries for pairwise analysis of building block bins in positions p_1 and p_2

p_3 P(bind) bin	p_1 P(bind) bin		
	$[0.00, 0.20)$	$[0.20, 0.40)$	$[0.40, 0.60)$
$[0.80, 1.00]$	0.9417	0.9973	1.0000
$[0.60, 0.80)$	0.6412	0.9813	1.0000
$[0.40, 0.60)$	0.4329	0.9403	0.9768
$[0.20, 0.40)$	0.2579	0.9037	0.9815
$[0.00, 0.20)$	0.0036	0.1666	0.2922

Table B.2: Matrix entries for pairwise analysis of building block bins in positions p_1 and p_3

p_3 P(bind) bin	p_2 P(bind) bin		
	$[0.00, 0.20)$	$[0.20, 0.40)$	$[0.40, 0.60)$
$[0.80, 1.00]$	0.9301	0.9994	0.9989
$[0.60, 0.80)$	0.4631	0.9953	0.9981
$[0.40, 0.60)$	0.2261	0.9769	0.9908
$[0.20, 0.40)$	0.1185	0.9301	0.9827
$[0.00, 0.20)$	0.0012	0.1325	0.2630

Table B.3: Matrix entries for pairwise analysis of building block bins in positions p_2 and p_3

B.1.5 UMAP Projection using 2D Tanimoto Similarity

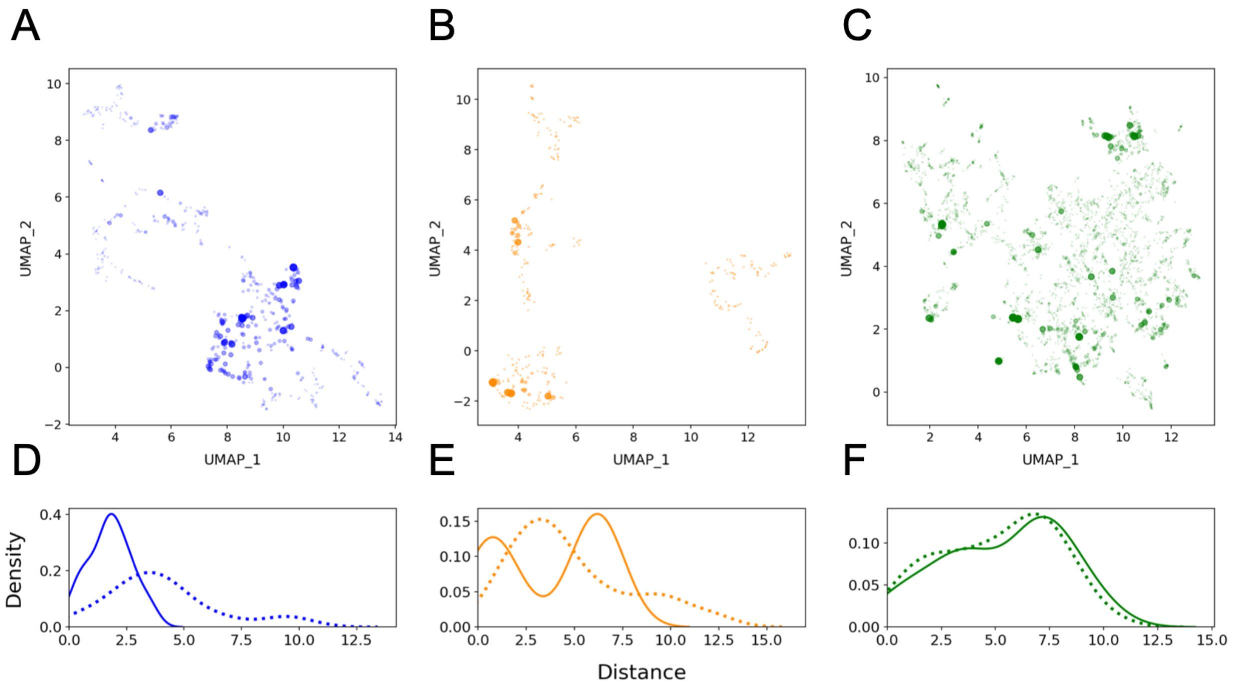


Figure B.4: (A–C) UMAP projections of chemical space for each library position using 2D Tanimoto. Pictured are building blocks in (A) p_1, (B) p_2 and (C) p_3. (D–F) Distributions of distances in UMAP space between the top 10 building blocks by P(bind) and randomly selected building blocks. Pictured are the distances between top 10 to top 10 (solid line) and top 10 to random (dotted line) building blocks for (D) p_1, (E) p_2 and (F) p_3. Some separation between high and low P(bind) building blocks still occurs for 2D Tanimoto, but is decreased compared to when using 3D Tanimoto combo.

B.1.6 Distances Between Building Blocks in UMAP space

Position	top - top distance	top - random distance
1	3.4877	6.0072
2	1.2091	4.7872
3	4.6037	8.2252

Table B.4: Distance between top P(bind) building blocks to other top P(bind) building blocks and to random building blocks using 3D Tanimoto

Position	top - top distance	top - random distance
1	1.7771	4.0865
2	3.7956	4.8249
3	5.3657	5.1005

Table B.5: Distance between top P(bind) building blocks to other top P(bind) building blocks and to random building blocks using 2D Tanimoto

B.1.7 Table of values for pairwise analysis of building block clusters

p_1 cluster id	p_2 cluster id									
	0	1	2	3	4	5	6	7	8	9
0	0.0001	0.0016	0.0000	0.0066	0.0000	0.0600	0.0001	0.0001	0.0261	0.0024
1	N/A	N/A	N/A	0.0230	N/A	0.1493	N/A	N/A	0.1469	0.0227
2	N/A	N/A	N/A	0.0264	N/A	0.0836	N/A	N/A	0.0383	0.0050
3	N/A	N/A	N/A	0.0343	N/A	0.1051	N/A	N/A	0.0595	0.0101
4	0.0000	0.0001	0.0000	0.0011	0.0000	0.0151	0.0000	0.0000	0.0090	0.0004
5	0.0003	0.0004	0.0000	0.0011	0.0000	0.0241	0.0002	0.0002	0.0186	0.0018
6	0.0001	0.0001	0.0000	0.0011	0.0002	0.0414	0.0001	0.0001	0.0327	0.0074

Table B.6: Matrix entries for pairwise analysis of building block clusters in positions p_1 and p_2

p_1 cluster id	p_3 cluster id																		
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
0	0.0340	0.0002	0.0003	0.0002	0.0002	0.0077	0.0000	0.0002	0.0082	0.0001	0.0003	0.0163	0.0185	0.0025	0.0001	0.0001	0.0250	0.0715	0.0058
1	0.1039	N/A	N/A	N/A	N/A	N/A	N/A	0.0238	0.0366	N/A	N/A	N/A	0.1154	0.0583	0.0000	0.0000	0.0908	0.2967	0.0483
2	0.0482	N/A	N/A	N/A	N/A	N/A	N/A	0.0000	0.0088	N/A	N/A	N/A	0.0359	0.0130	0.0000	0.0000	0.0401	0.1075	0.0065
3	0.0628	N/A	N/A	N/A	N/A	N/A	N/A	0.0048	0.0153	N/A	N/A	N/A	0.0546	0.0229	0.0000	0.0000	0.0563	0.1630	0.0133
4	0.0106	0.0000	0.0000	0.0000	0.0000	0.0013	0.0000	0.0000	0.0002	0.0000	0.0000	0.0018	0.0030	0.0001	0.0000	0.0000	0.0033	0.0076	0.0000
5	0.0190	0.0001	0.0001	0.0001	0.0000	0.0028	0.0000	0.0000	0.0000	0.0000	0.0007	0.0031	0.0029	0.0005	0.0000	0.0001	0.0076	0.0101	0.0000
6	0.0291	0.0003	0.0001	0.0000	0.0003	0.0007	0.0000	0.0000	0.0138	0.0001	0.0001	0.0012	0.0196	0.0052	0.0000	0.0001	0.0176	0.0554	0.0117

Table B.7: Matrix entries for pairwise analysis of building block clusters in positions p_1 and p_3

p_1 cluster id	p_3 cluster id									
	0	1	2	3	4	5	6	7	8	9
0	0.0000	0.0000	0.0001	0.0000	0.0000	0.0004	0.0000	0.0000	N/A	0.0000
1	0.3013	0.0002	0.0008	0.0009	0.0006	0.0027	0.0000	0.0000	N/A	0.0001
2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	N/A	0.0000
3	0.0292	0.0006	0.0003	0.0000	0.0000	0.0378	0.0000	0.0011	0.0000	0.0000
4	0.0000	0.0000	0.0000	0.0000	0.0007	0.0002	0.0000	0.0000	N/A	0.0002
5	0.0875	N/A	N/A	N/A	N/A	N/A	N/A	0.0069	0.0216	N/A
6	0.0000	0.0001	0.0001	0.0000	0.0001	0.0000	0.0000	0.0000	N/A	0.0002
7	0.0000	0.0003	0.0001	0.0000	0.0000	0.0003	0.0000	0.0000	N/A	0.0000
8	0.0464	N/A	N/A	N/A	N/A	N/A	N/A	0.0170	0.0158	N/A
9	0.0102	N/A	N/A	N/A	N/A	N/A	N/A	0.0000	0.0010	N/A

p_1 cluster id	p_3 cluster id								
	10	11	12	13	14	15	16	17	18
0	0.0005	0.0000	0.0000	0.0002	0.0002	0.0002	0.0000	N/A	0.0000
1	0.0001	0.0089	0.0000	0.0000	0.0001	0.0003	0.0000	N/A	0.0000
2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	N/A	0.0000
3	0.0013	0.0696	0.0064	0.0000	0.0000	0.0000	0.0082	0.0094	0.0000
4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	N/A	0.0000
5	N/A	N/A	0.0731	0.0300	0.0000	0.0000	0.0736	0.2044	0.0147
6	0.0000	0.0005	0.0001	0.0005	0.0000	0.0000	0.0027	N/A	0.0000
7	0.0002	0.0014	0.0000	0.0000	0.0000	0.0000	0.0000	N/A	0.0000
8	N/A	N/A	0.0538	0.0241	0.0000	0.0000	0.0514	0.1503	0.0220
9	N/A	N/A	0.0030	0.0010	0.0000	0.0000	0.0013	0.0027	0.0002

Table B.8: Matrix entries for pairwise analysis of building block clusters in positions p_2 and p_3

B.1.8 Full Width at Half Maximum Values

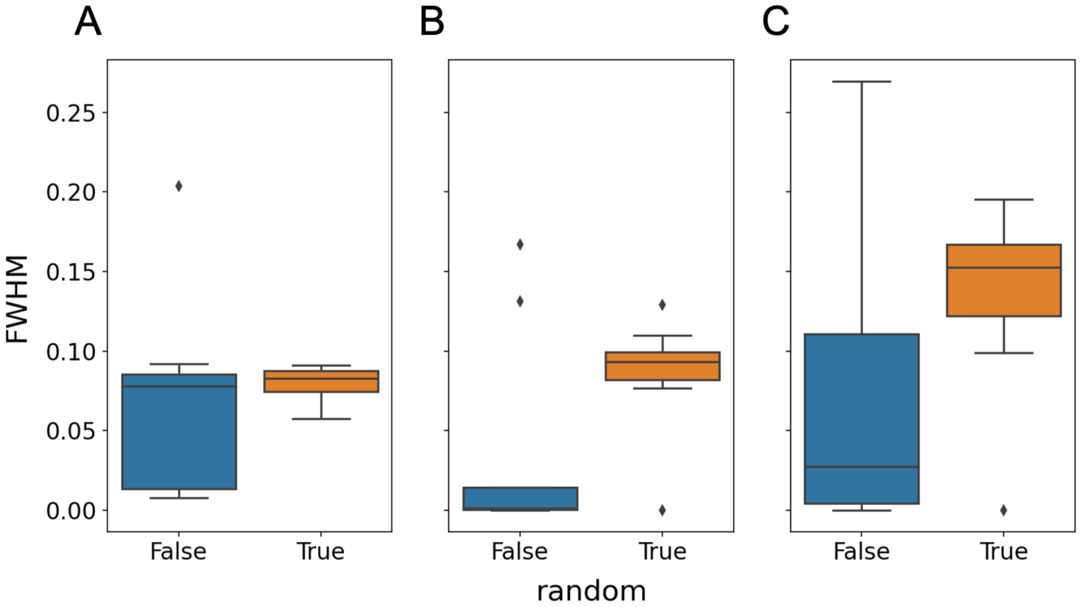


Figure B.5: Boxplot of FHWM values for the distribution of P(bind) values in clusters at each building block position. The blue boxplot indicates the FWHM for the P(bind) distributions in each cluster generated using HDBSCAN. The boxplot in orange is the average FWHM of 50 different random initializations of each cluster P(bind) distribution. We observe no statistically significant difference between the FWHM of the P(bind) distributions from HDBSCAN and random clustering in (A) position 1, but do for (B) position 2 (p-value=0.0395) and (C) position 3 (p-value=1.20e-3).

B.1.9 Area Under the Curve Across Multiple Random Trials

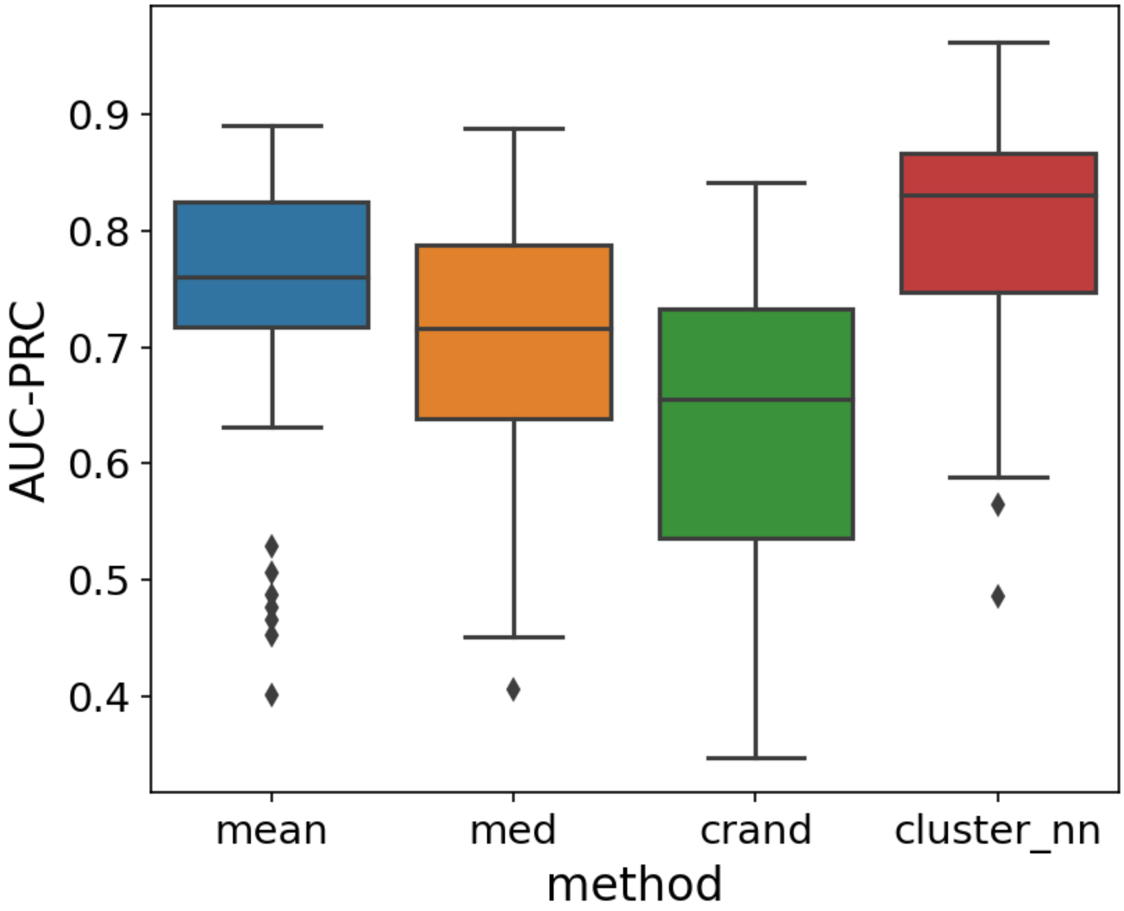
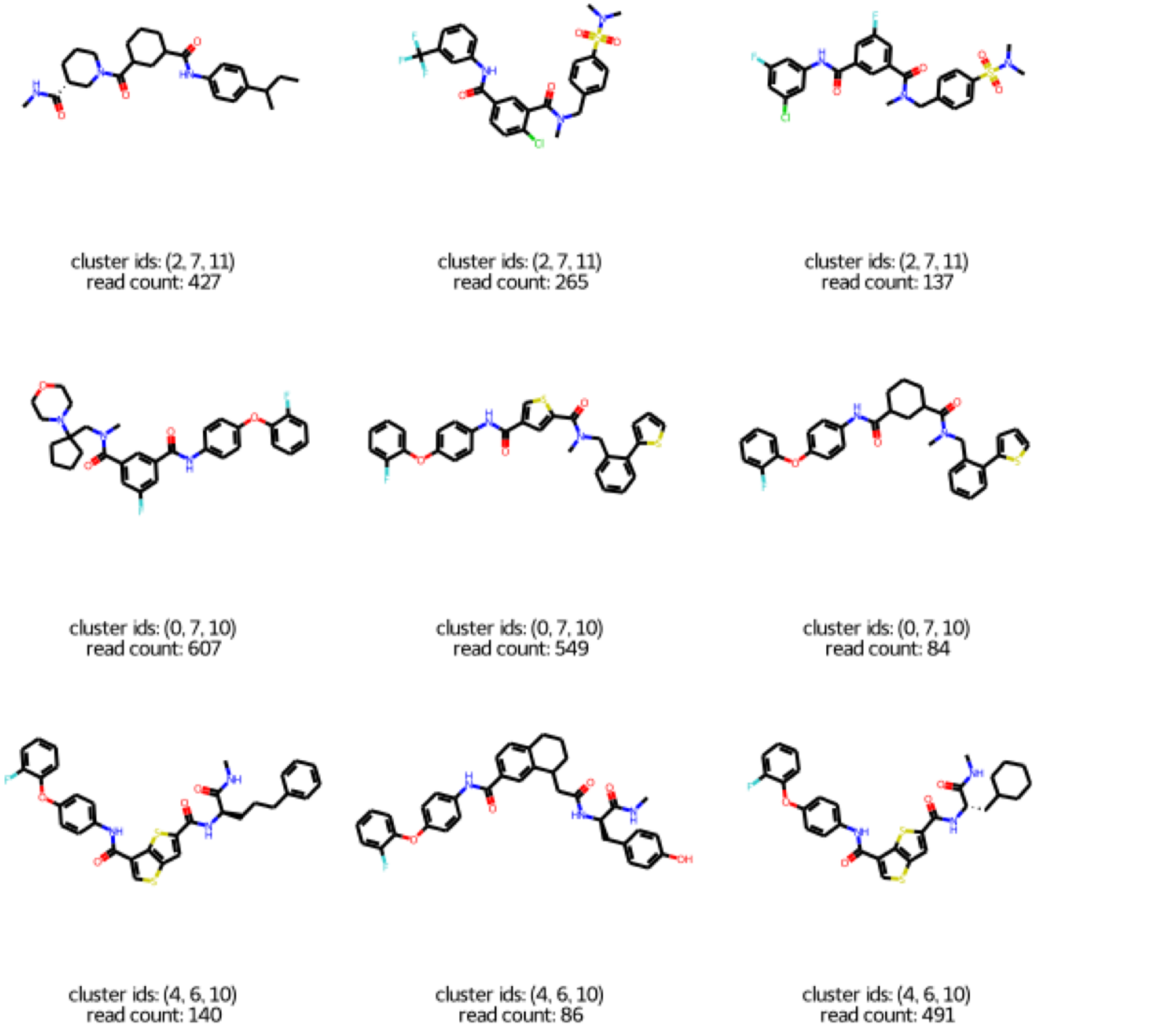


Figure B.6: Performance of different prediction methods across 50 random trials.

B.1.10 Diversity of Holdout Set Compound Predictions



cluster ids: (2, 7, 11)
read count: 427

cluster ids: (2, 7, 11)
read count: 265

cluster ids: (2, 7, 11)
read count: 137

cluster ids: (0, 7, 10)
read count: 607

cluster ids: (0, 7, 10)
read count: 549

cluster ids: (0, 7, 10)
read count: 84

cluster ids: (4, 6, 10)
read count: 140

cluster ids: (4, 6, 10)
read count: 86

cluster ids: (4, 6, 10)
read count: 491

Figure B.7: Examples of holdout set trisynthons predicted to bind to sEH. We trained a decision tree model to predict whether compounds containing at least one out-of-sample building block would bind to sEH. Cluster ids are given as a triplet of values, corresponding to the cluster ID for each position in the library. We did not provide read count values when training our model, but show them here to demonstrate that all the following model predictions are correct.

B.1.11 Distribution of Read Count Values for Binders

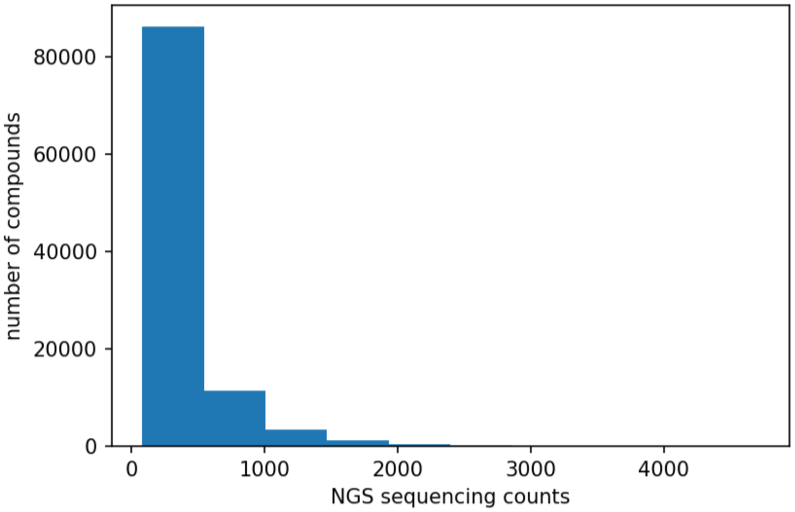


Figure B.8: Distribution of NGS sequencing counts (read counts) for compounds designated as binders. From a complete set of DEL selection data screened against sEH, we selected the top 10K compounds based on read count to analyze, giving us a set of highly enriched binders. The minimum read count in our set of binders is 81 and the maximum is 4712.

Read count threshold	Fraction of binders	Fraction of non-binders
1	0.0233	0.9767
50	0.0233	0.9767
100	0.0198	0.9802
500	0.0043	0.9957
1000	0.0013	0.9987

Table B.9: Fraction of compounds classified as binders and non-binders with varying read count threshold

B.2 Additional Methods

B.2.1 HDBSCAN Objective Function

We devised an empirical objective function to evaluate the performance of the HDBSCAN algorithm. We tried using three different metrics to evaluate clustering quality – silhouette score [107], calinski-harabasz score [16] and davies-bouldin score [25] – and found that for every metric, there was either not enough cluster resolution, too many noise points, or too many small clusters formed (Figure B.9). Thus, we created our own objective function that would score any HDBSCAN run by the number of noise (unclustered) points and the average intracluster distance. The goal was to find a set of hyperparameters that would generate compact groups of highly similar compounds, but not overfit such that many points would be left unclassified. We arrived at an empirical formula for the objective function, L

$$L = n_{noise} + 10 * ICD \tag{B.1}$$

We found that there was a global minimum value for the objective function across the sampled hyperparameters, meaning we could use those values as the best initialization for each HDBSCAN run (Figure B.10). We selected the set of HDBSCAN parameters which would minimize the objective function.

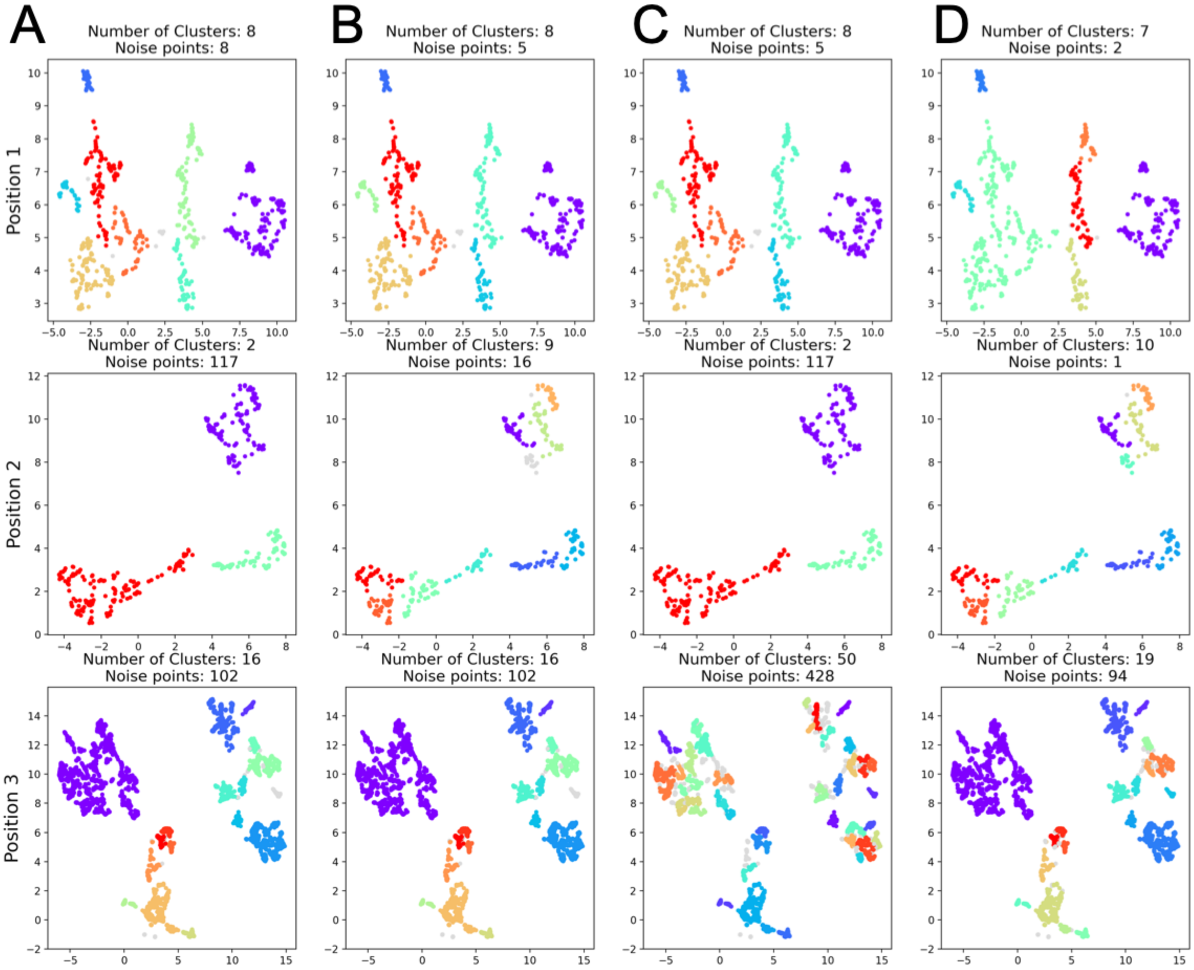


Figure B.9: Cluster results for UMAP projections using different metrics. We compare clustering results when optimizing via various metrics. From left to right, (A) silhouette score, (B) calinski-harabasz score, (C) davies-bouldin score and (D) our empirical objective function. Our objective function results in the greatest number of building blocks clustered (fewest noise points) for each building block position.

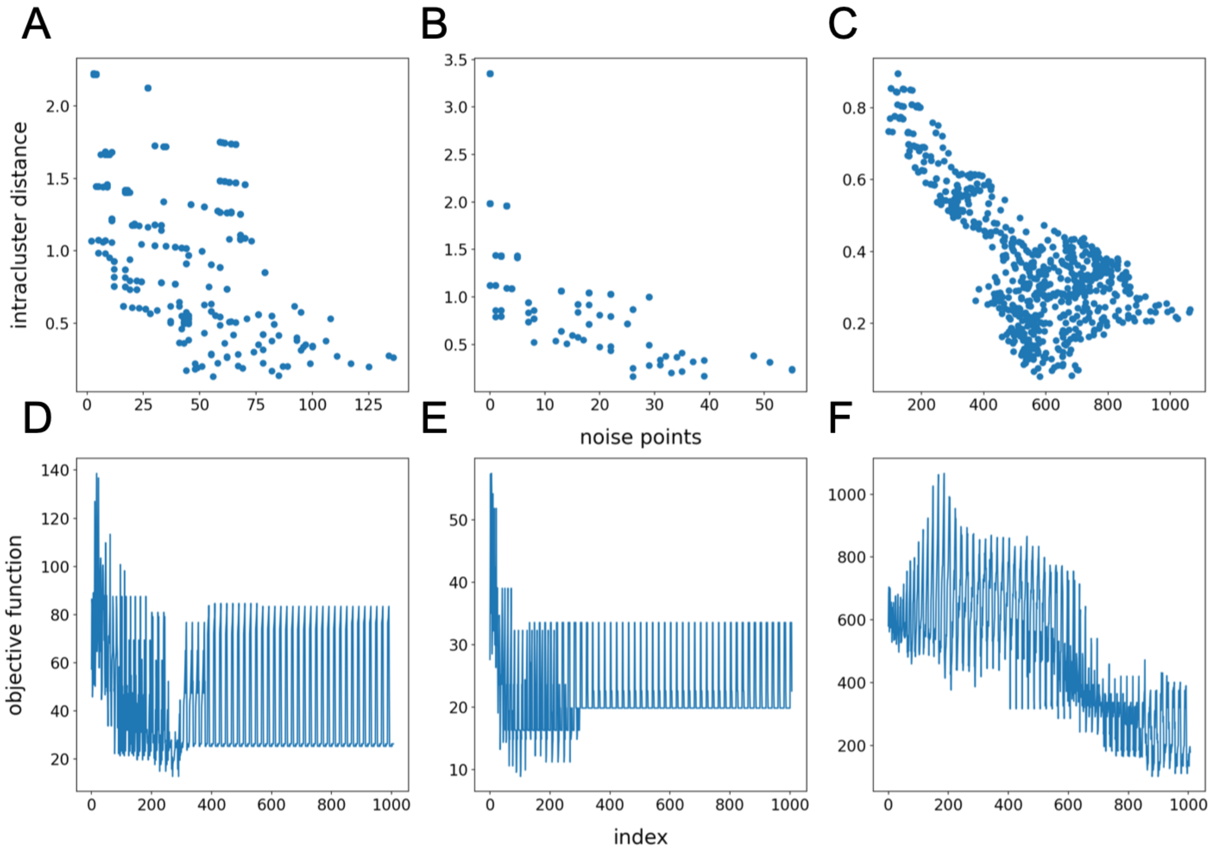


Figure B.10: Relationship between the number of noise points and intracluster distance for HDBSCAN initializations. (A–C) For a given HDBSCAN initialization, we count the number of noise points and calculate the average distance between points within a cluster (intracluster distance). There is no discernable correlation between the number of noise points and intracluster distance for (A) p_1 (B) p_2 or (C) p_3. (D–F) We construct an objective function which accounts for both the number of noise points and the intracluster distance of a cluster. The index variable on the x-axis represents a distinct set of hyperparameters for HDBSCAN. We observe that the objective function has a global minimum for (D) p_1 (E) p_2 and (F) p_3. For each position, we take the set of hyperparameters corresponding to the global minimum to be the optimal set of parameters for that position.

B.2.2 Selecting Decision Tree Parameter

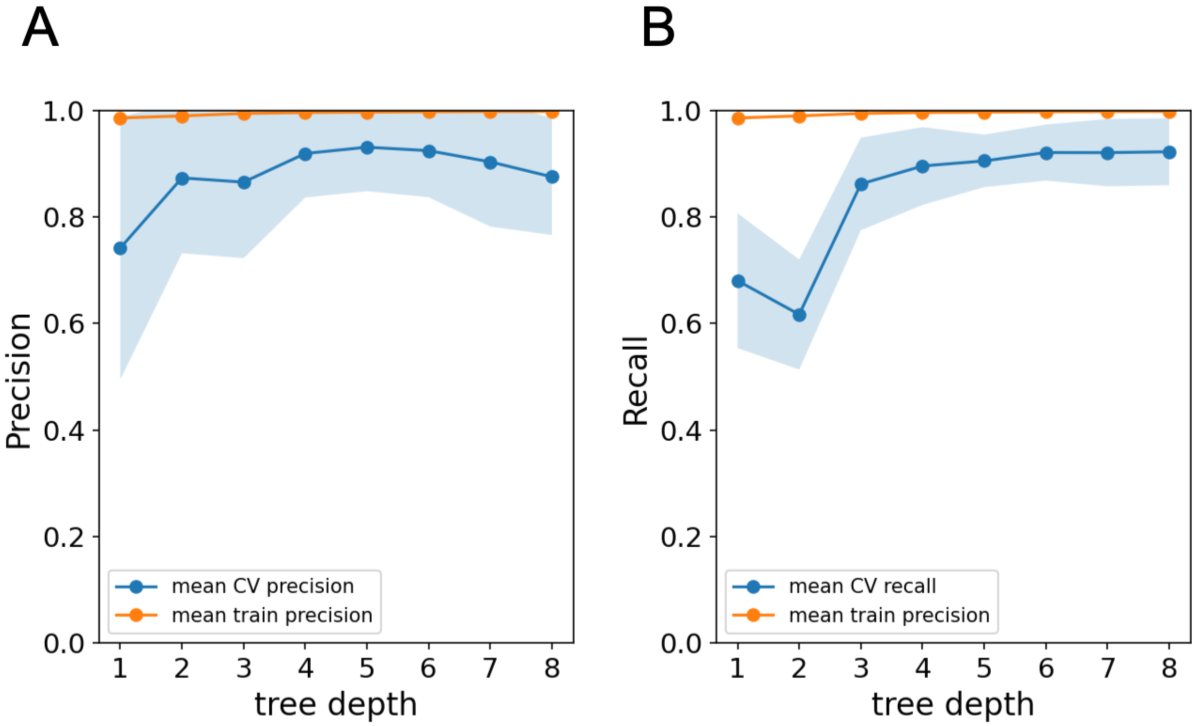


Figure B.11: Hyperparameter selection for the decision tree model. We used 5-fold cross validation to find the best choice for the depth of the decision tree. For each value of tree depth, we trained a model on 4/5ths of the training data and evaluated the precision and recall of the model on the remaining 1/5th. This process was repeated 5 times to include all training points once in the validation set. We report the mean and standard deviation of both (A) precision and (B) recall across the 5 cross validation folds for each value of tree depth. We selected depth=5 as our optimal tree depth – beyond this point, the model appears to overfit, indicated by the gradual drop in model precision.