©Copyright 2022
Chase Armer

Deep Learning and Cloud Science for the Engineering of Biological Molecules and Systems

Chase Armer

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science

University of Washington

2022

Committee:

Herbert Sauro

Jesse Zalatan

Program Authorized to Offer Degree:
Bioengineering

University of Washington

Abstract

Deep Learning and Cloud Science for the
Engineering of Biological Molecules and Systems

Chase Armer

Chair of the Supervisory Committee:
Herbert Sauro
Department of Bioengineering

Solving many of the greatest challenges facing our industrial landscape and therapeutic development will require innovations in both computational and experimental biology alike. Deep learning has demonstrated it capacity to model complex systems in Natural Language Processing, Computer Vision, and Protein Modeling, and is poised to continue pioneering state-of-the-art results for modeling biological molecules and networks. Furthermore, cloud science laboratories have the potential to become a significant driving force in the acquisition of massive biological datasets, as well as in the future of reproducible and accessible life science research. The following document discusses a list of research projects focused on these emerging fields. Chapters 1 and 2 investigate novel deep learning models and workflows for the design of protein minibinders. Chapter 3 considers a new protocol for the computational design of protein-binding macrocycle therapeutics. Chapter 4 explores the utility of cloud laboratories for experimental biology, and illustrates a low-cost, high-throughput protocol for enzyme characterization. Finally, Chapter 5 details a new deep learning approach for simulating metabolic systems.

TABLE OF CONTENTS

LIST OF FIGURES

GLOSSARY

ALPHAFOLD: A deep learning model for protein structure prediction, developed by Deepmind.

FASTDESIGN: A protein design method which optimizes a protein sequence in the context of a binding complex through iterative cycles of rotamer repacking, backbone and side chain torsion minimization, and sequence sampling, developed in Rosetta.

GALIGANDDOCK: A molecular docking method which leverages a genetic algorithm, grid-based force fields, and conformer sampling to dock ligands to proteins.

PREDICTED ALIGNED ERROR (PAE): An AlphaFold metric quantifying the model's predicted error at each residue in the protein, thereby serving as a confidence metric for the predicted structure.

PROTEINMPNN: A deep learning-based, protein sequence design model based on Message Passing Neural Networks.

RIFDOCK: A docking and design method for scaffolding proteins to ligands or other proteins.

ROSETTAFOLD: A deep learning model for protein structure prediction, developed by the Baker Lab.

TELLURIUM: A python environment for building, simulating, and analyzing models in systems biology.

ACKNOWLEDGMENTS

The author wishes to express sincere appreciation to University of Washington's Biological Physics, Structure and Design Program, where he has had the opportunity to work with David Baker, Frank DiMaio, Jesse Zalatan, and Herbert Sauro.

Chapter 1

A HYBRID APPROACH: DEEP LEARNING AND ROSETTA FOR MINIBINDER DESIGN

1.1 What are Minibinders?

Minibinders are small proteins of around fifty amino acids in length that are designed to bind tightly to a target protein. By binding to and altering the function of specific proteins, minibinders have a vast array of healthcare applications from cancer therapeutics to antibacterial agents[1, 2]. Designing these mini-proteins to possess high expressibility, stability, and specificity for arbitrary protein targets is an ongoing challenge in the field of protein engineering.

1.2 Minibinder Design Pipeline

The original minibinder design pipeline in use when I arrived at the David Baker laboratory was a three step process. First, RifDock, a docking and design method for scaffolding proteins to ligands or other proteins[3], generates a large list of minibinder protein backbones which are predicted to bind to the target protein. Secondly, Rosetta's FastDesign protocol performs successive cycles of sequence design and structure refinement to iteratively optimize each minibinder backbone within the context of its predicted binding pocket. Finally, the optimized minibinder designs are individually submitted into AlphaFold 2[4] along with the target protein, wherein the 3D structures of the two proteins will be predicted and successful designs will be filtered.

When provided with two distinct proteins, AlphaFold will either predict the two proteins to be bound together in a single 3D complex or will place a large separation between the two folded proteins. Practitioners have found that AlphaFold's predictions of a single 3D complex correlate strongly with binder success. Furthermore, researchers in the Baker lab have identified that AlphaFold's Predicted Aligned Error (PAE), an output metric measur-

ing the confidence of the predicted structures, correlates strongly with binder success as well. Therefore, AlphaFold is utilized to filter the optimized minibinder designs by selecting only those which are predicted by AlphaFold to be complexed with the target protein and possess an AlphaFold PAE of less than ten. This computational pipeline is used to generate a list of promising minibinder designs which are then ordered and experimentally characterized for their expressibility, stability, and specificity for the protein target.

1.3 ProteinMPNN: A New Approach

ProteinMPNN is a deep learning-based protein sequence design model which leverages a Message Passing Neural Network (MPNN) to generate protein sequences that satisfy a 3D protein backbone[5]. Our ambition was to improve the efficacy of the minibinder design pipeline by replacing Rosetta's FastDesign with ProteinMPNN, thereby creating a hybrid Rosetta and deep learning-driven design pipeline for minibinder generation. We found that ProteinMPNN sequence design was 40 times faster than the FastDesign approach, and we sought to directly compare the ability of these two methods to generate viable minibinder designs. Since compute time was the limiting resource in computational design, we believed the most accurate means of comparing the two approaches would be to allot the same amount of compute time for both methods and to compare the number of successful minibinder designs as determined by AlphaFold filtering. Therefore, the number of designs that would be evaluated by AlphaFold would be the same between both of the approaches. Because ProteinMPNN would generate 40 times more minibinders than FastDesign when given the same amount of compute time, we needed a fast method to filter out the top ProteinMPNN designs so that both methods would present the same number of designs for AlphaFold filtering. Therefore, we utilized the Rosetta Predictor function to relax the backbone and predict the change in binding free energies (DDG) for each of our minibinder designs. This DDG was used to filter out the top designs and keep computational time equal between the two methods. The computational cost of the predictor step was negligible in comparison to the other computational methods, and therefore was not included in the calculations for allotting equal computational time.

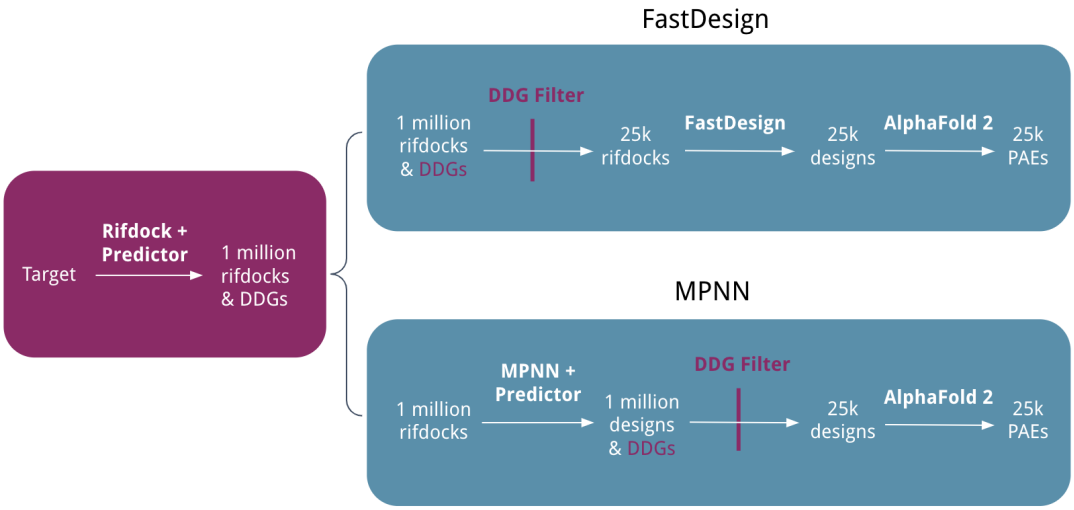

Figure 1.1: Diagram of FastDesign (top) and MPNN (bottom) comparison experiment. Both models use RifDock and the Predictor to create one million RifDock scaffolds and their associated DDGs. The FastDesign approach filters the top twenty-five thousand RifDocks to be designed. The MPNN approach generates designs for all one million RifDocks before filtering down to twenty-five thousand designs.

1.4 Results and Implications

We conducted the minibinder design comparison experiment on three unique protein targets, known as CD86, Kit, and ROR1. Our experiments, shown in Figure 1.2, demonstrated that ProteinMPNN generated anywhere from 60 to 460 more successful binders than FastDesign as determined by AlphaFold, resulting in a 2 to 5 fold improvement in binder design. These results of this project represent a dramatic improvement in minibinder design and have established this new hybrid approach, of combining Rosetta scaffolding with deep learning driven sequence generation, as the new paradigm for minibinder design within the lab.

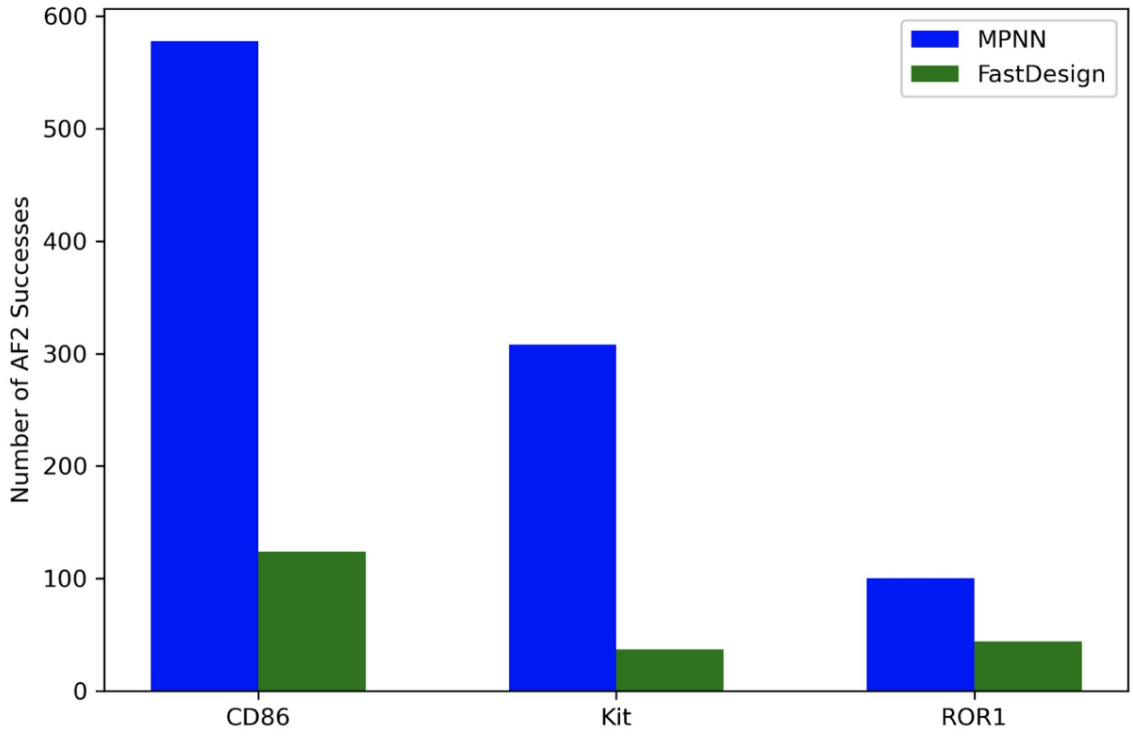

Figure 1.2: Comparison of the ProteinMPNN and FastDesign for designing successful minibinders against three unique protein targets. Success is defined as a design with an AlphaFold 2 PAE of less than or equal to 10.

Chapter 2

END-TO-END DEEP LEARNING FOR MINIBINDER DESIGN

2.1 Generative Modeling for Peptides and Binders

To further leverage the advantages of deep learning for minibinder design, I designed and trained a deep learning model, PeptideDesigner, to generate small peptides that bind to specific locations on a given target protein. More specifically, I fine-tuned the RoseTTAFold model to receive as input both the sequence and structure of a target protein along with a 3D coordinate at a user-specified location on the protein's surface, and to produce as output a protein complex involving the original target protein in addition to a newly designed, 7-residue peptide that is bound onto the surface of the protein at the location of the 3D input coordinate. The purpose of this model was to generate a peptide binder at a specified location of a protein's surface, and then leverage the inpainting and constrained hallucination deep learning techniques to generate the full protein scaffold from the designed peptide, thereby creating the first end-to-end deep learning pipeline for protein minibinder design. With this approach, researchers could arrange for the peptide to bind a specific binding pocket of interest by sampling the 3D coordinate in different locations within the pocket. Conversely, if researchers wanted to design a binder with no specific binding pocket in mind, the researchers could sample the input 3D coordinate uniformly across the surface of the protein to generate a diverse list of candidate designs.

2.2 What is RoseTTAFold?

RoseTTAFold is a deep neural network developed in the Baker lab to predict the three-dimensional structure of proteins from their sequence[6]. RoseTTAFold is a three track model which processes a one-dimensional sequence, a two-dimensional distance map, and a three-dimensional coordinate graph in parallel, allowing the three modules to transmit

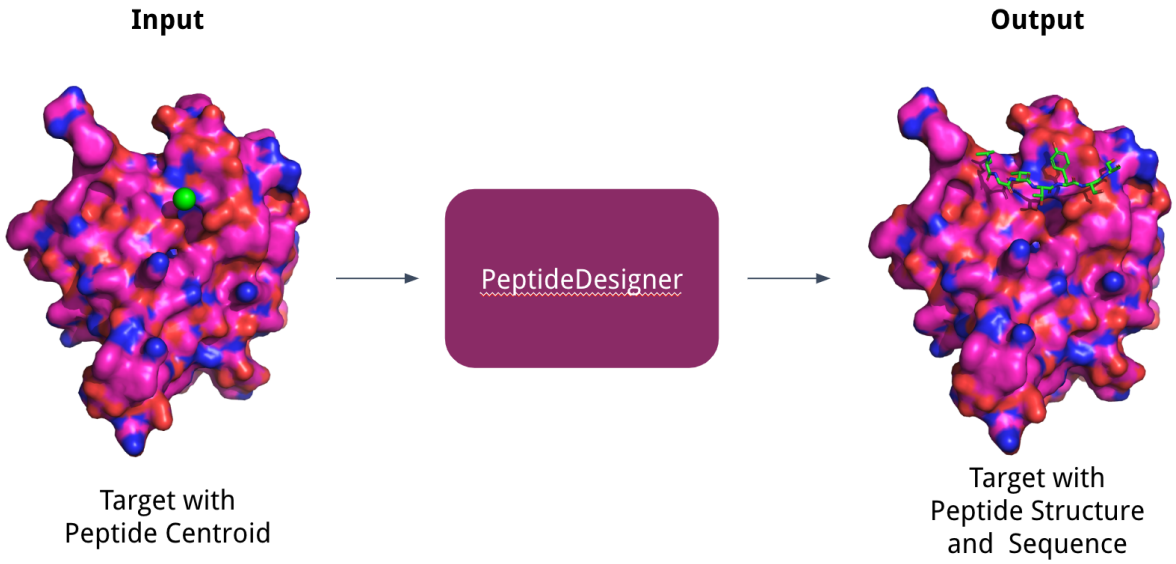

Figure 2.1: Peptide Designer

information between one another. The input to the RoseTTAFold model is the protein sequence of interest, a multiple sequence alignment (MSA) of evolutionary related sequences, and the 3D structures of similar sequences within the Protein Data Bank (PDB). The input MSA has 15% of its amino acids replaced with masked tokens, and a loss function tracks the model's ability to predict the identity of these masked amino acids. The output of the model includes the predicted MSA, as well as the predicted 3D structure of the protein.

2.3 Inpainting and Constrained Hallucination

The protein sequence inpainting and constrained hallucination techniques both provide the opportunity to generate a full minibinder protein which supports the central, 7-residue peptide designed by the PeptideDesigner model[7]. The inpainting technique relies on a fine-tuned RoseTTAFold model trained to predict sequence and structure information from partially masked sequence-structure pairs. To produce a minibinder protein from our peptide, the model would be provided with the sequence and structure of the target protein along with an additional protein which contains our peptide embedded between masked amino acids, and would be tasked with producing a full minibinder sequence that stabilizes

and supports the peptide's interactions with the target protein. The constrained hallucination approach possesses the same inputs and outputs, but instead performs Monte Carlo sampling of the amino acid sequence in order to produce a viable protein scaffold which contains the functional site, as determined by a protein structure prediction model such as RoseTTAFold or trRosetta. The loss function of the hallucination protocol can be further modified to include additional terms which allow users to customize the sequence generation to match their specific use-case.

2.4 Building the Dataset

To build the dataset for this model, I used a protein-complex dataset which contains the PDB files for over 100,000 unique protein complexes. I isolated all interacting pairs of proteins within this dataset, resulting in a set of 270,000 protein pairs. For each protein pair, I defined the binding surface between the two proteins as every amino acid within 10 angstroms of another amino acid on the paired protein. Finally, I identified every contiguous, 7-residue sequence on the binding surface and extracted it as a unique datapoint. Each extracted datapoint was the sequence and structure of the isolated 7-residue peptide and the protein partner it was complexed to. This datapoint was then restructured into the input-output pair that would be provided to the neural network during model training. The output datapoint simply contained the full sequence and structure of the protein partner and the 7-residue peptide in complex. The input datapoint, however, replaced the 7-residue peptide with a 'peptide centroid', wherein the sequence of the peptide was represented as masked tokens, and the 3D coordinate of each amino acid was collapsed into the same position – the average 3D location of all amino acids in the peptide. Thus, the task of the model would be to 'expand' this masked centroid into the 7-residue peptide by defining its sequence and structure. This protocol resulted in a total of 645,000 unique input-output pairs for model training.

2.5 Building the Model

In order to fine-tune the RoseTTAFold model on this new task, it was necessary to provide the input data containing the target protein and stub centroid through the appropriate

channels of the original model inputs. One of the original inputs to the model was the three-dimensional structures of related sequences. I repurposed this input to instead provide the structure of the target protein and stub centroid. The original RoseTTAFold training included a masked MSA and an associated loss function for predicting the identity of the masked tokens. I repurposed this MSA prediction task to instead serve as our peptide sequence generation task. To do this, the MSA would contain only a single sequence: the product of concatenating the target protein sequence with the seven-length, masked stub centroid. We communicated to the model that the stub centroid was a separate peptide by increasing the index position, an additional input which gives an index value to each amino acid in the provided protein sequence, by a value of four-hundred – a value also used commonly during RoseTTAFold training to indicate separate sequences. Therefore, the model would not only predict the 3D coordinates of the peptide, but would also predict the identity of the masked amino acids, thus predicting the peptide's sequence. New data loaders were written to accommodate this new task, and the loss functions were also rewritten to provide a separate 'peptide loss' that upweighted the loss for the predictions of the peptide's sequence and structure.

2.6 Results and Future Work

The model was trained on this new task over a five-day period on the laboratory's GPU cluster. The training results demonstrated that the model was learning to accurately generate the sequence and structure of the binder peptides. Figure 2.2 shows the results of model training. Figure 2.3 shows a contact map comparison between a predicted and ground-truth protein complex. The lack of defined detail in the predicted peptide map suggests the model could be further improved.. Hyperparameter optimization, longer training regimes, and additional dataset curation could lead to improvements in model performance. Follow up research for building the full minibinder design pipeline by applying inpainting and constrained hallucination to the designed peptides would be the next step for investigating the full potential of this new approach.

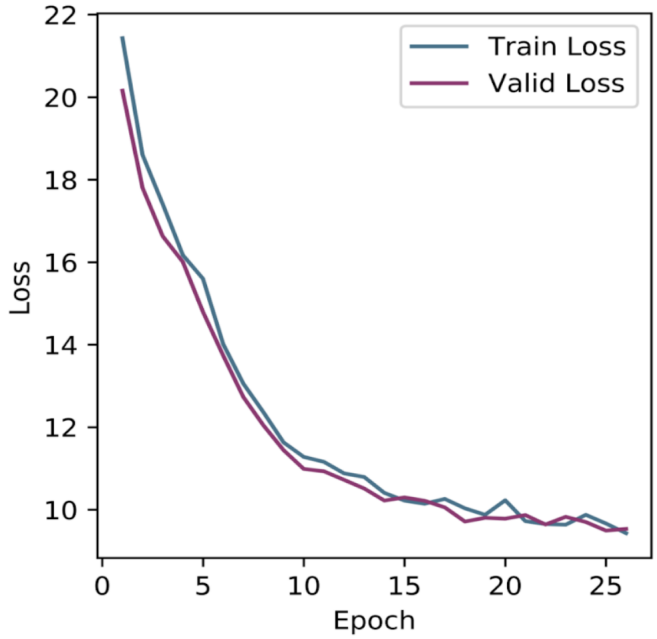

Figure 2.2: Training and validation loss for the PeptideDesigner model.

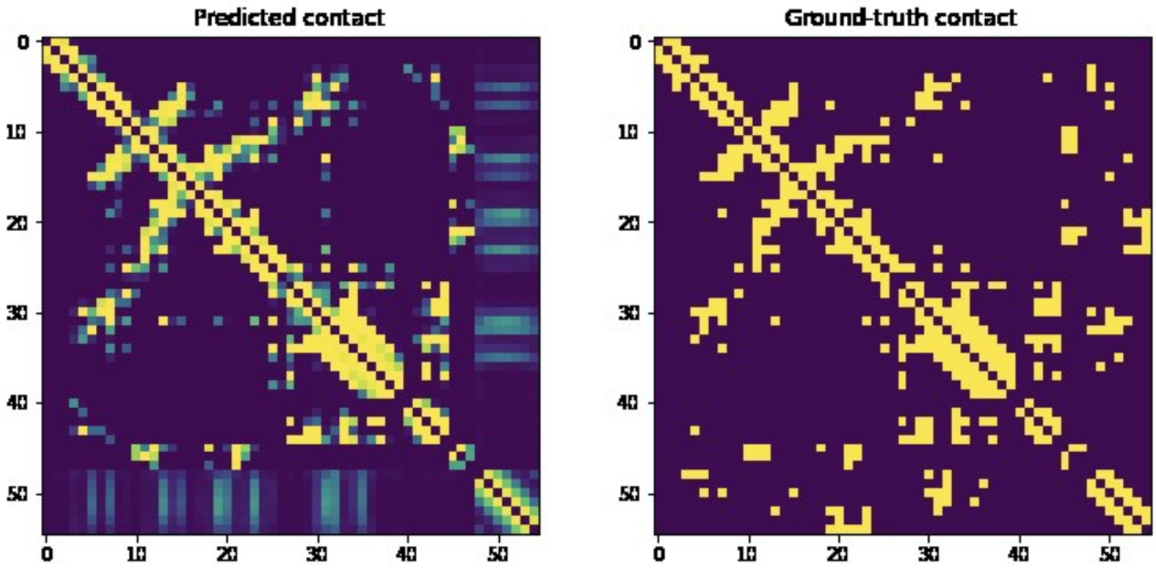

Figure 2.3: Contact Map of Predicted (left) and Ground-Truth (right) complex between the target protein and peptide. The peptide is represented in the final seven residues of the map.

Chapter 3

COMPUTATIONAL MACROCYCLE DESIGN

3.1 Macrocycle Introduction

Macrocycles are cyclic peptides of around ten amino acids in length. These cyclic peptides are attractive candidates for therapeutic development due to the innate strengths they draw from small molecules and proteins alike. More specifically, macrocycles combine the cell permeability of small molecules with the tunability and specificity of biologic therapeutics. However, the permeability of a given macrocycle depends on the sequence and stereochemistry of its amino acids, and thus not all macrocycles are highly permeable to cell membranes. Previous research combined large peptide libraries with in vitro permeability assays to identify a collection of 300,000 macrocycle backbones that are cell permeable[8–10]. The researchers found that the most important factor in determining cell permeability was the location and stereochemistry of prolines and glycines within the peptide. That is to say, when a given arrangement of prolines and glycines is present in the peptide, all other amino acids within the macrocycle can be changed with little to no impact on permeability. Therefore, researchers define these cell-permeable macrocycle backbones only by the location and stereochemistry of the prolines and glycines they contain.

3.2 A New Macrocycle Design Approach

The computational macrocycle design protocol established within the lab used the GALigandDock software to dock macrocycles onto a target protein. GALigandDock applies a genetic algorithm to iteratively evolve a collection of protein-ligand complexes, successively modulating the sequence and binding conformation of the ligand to minimize the predicted binding energy[11]. Previous results indicated this approach generated macrocycle designs which made strong contacts to the binding surface, but did so with only a few of the amino acids on the peptide, leaving most of the macrocycle backbone free-floating without any

contact point with the target protein. Therefore, the lab was interested in a new design approach that could generate macrocycles that make strong contacts across the entire peptide backbone. The new approach we devised was to separate macrocycle design into two separate stages. First, we would dock all 300,000 macrocycle backbones from the cell-permeable macrocycle dataset onto the target protein. All non-backbone amino acids (non proline or glycine amino acids) were replaced with alanines prior to docking. The purpose of this step was to ensure that the key backbone amino acids were making strong contracts with the binding surface. Following this, we would then apply GALigandDock optimization to modulate the non-backbone amino acids and identify the lowest energy sequence and conformation. The macrocycles generated from this dock-and-design process would then be filtered on key metrics that correlate to binder success, described further below.

3.3 Protocol Speed Improvements

The computational time necessary to dock and design all 300,000 macrocycles with this protocol was fairly expensive, and thus many projects using this approach would be forced to sample only a subset of the macrocycles. My ambition was to increase the speed of the dock-and-design protocol to allow the sampling of a greater number of macrocycles for a given computational budget. There are three key parameters in the GALigandDock protocol which were likely to have the largest impact on speed. The first parameter was Repeats, the number of rounds of evolution. The second was NPool, the number of variations per round. The third was MaxIter, the maximum number of iterations available for conformational optimization. While I could lower these values to increase the speed of the protocol, I needed a way to measure how lowering these variables would impact performance. Since the output of this protocol should represent the lowest energy conformation for a given macrocycle backbone, I defined the decrease in performance for a given parameter change as the average root-mean-squared deviation (RMSD) between the macrocycle conformations it produced and the original macrocycles produced by the default parameter values. To identify the optimal parameter values in this three-parameter search space, I created a collection of one hundred parameter sets, where the value of each parameter was randomly sampled. The default values and sampling range for the parameters are shown in Table 3.1.

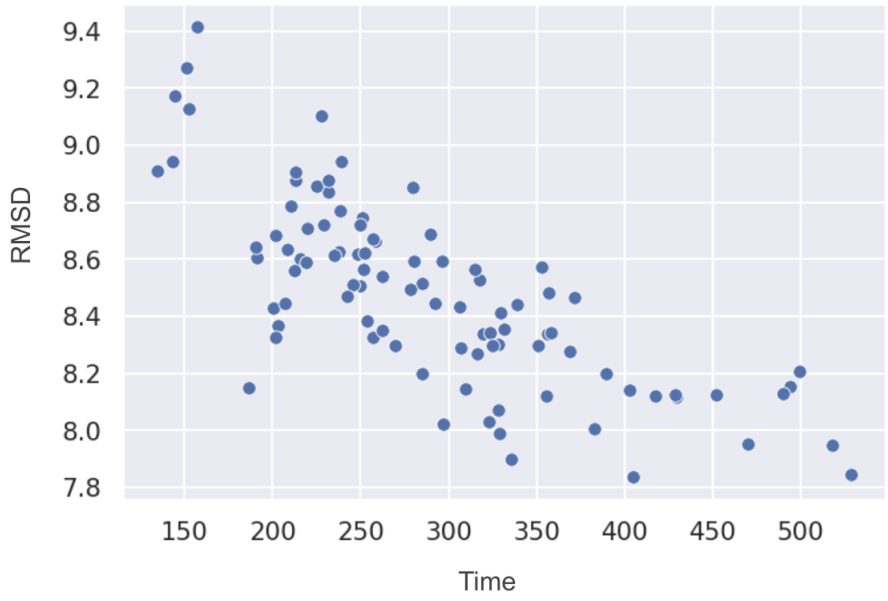

Figure 3.1: Plot of RMSD versus Time. Each point is a randomly sampled parameter set. RSMD is calculated by comparing the output of the random parameter set with the output of the default parameter set on a collection of 300 macrocycles.

For each of the parameter sets, I docked the same list of three hundred macrocycles and compared the speed and average RMSD to the default parameters. Figure 3.1 shows the results of this experiment, where each point represents a unique parameter set. Table 3.2 illustrates the Pearson Correlation of each parameter to the Time and RMSD.

	Repeats	**NPools**	**MaxIter**
Sample Range	1 - 10	10 - 100	10 - 100
Default	10	100	100

Table 3.1: Sample Range and Default Parameters for the Random Search Experiment.

A perplexing result of this experiment was that the lowest RMSD values were just over 7.8 angstroms, a significant distance in the context of macrocycle docking. To better understand this result, I ran a control experiment where I compared the RMSD between two

	Time	**RMSD**
Repeats	0.46	-0.12
NPools	0.82	-0.70
MaxIter	-0.04	0.05

Table 3.2: Pearson Correlation to Time and RMSD for each GALigandDock parameter, calculated from the 100 randomly sampled parameters sets.

docking experiments which both used the default parameters. These two identical parameters resulted in an average RMSD of 7.79, suggesting there was a fundamental stochasticity to the docking results of the GALigandDock protocol. Therefore, since the docking software appeared to be capable of identifying multiple low energy conformations for a given macrocycle, I decided a better alternative was to compare the average DDG of the generated macrocycles. A parameter change would thus be valuable if it increased the speed of docking whilst having only a slight change in the binding energy of its lowest-energy conformation relative to the default parameters. I repeated the one hundred parameter set random search experiment, this time only sampling different values for the number of Repeats and NPools. Since Table 3.2 indicated that MaxIter had a negligible small impact on time and performance, it was set back to the default value for all subsequent experiments. Figure 3.2 illustrates the results of this experiment, where the default parameter set is highlighted in red. The parameter set highlighted by a red '1', which contains the parameter values of Repeats = 5 and NPools = 80, was selected as the ideal balance between speed and performance. This parameter selection provided a 46% increase in docking speed for only a 3% decrease in average DDG. To further speed up the docking protocol, I rewrote the batching process to enable the lab's CPU cluster to perform parallel processing of the docking script more efficiently. The combination of the new parameters with the new batching resulted in a 72% increase in speed.

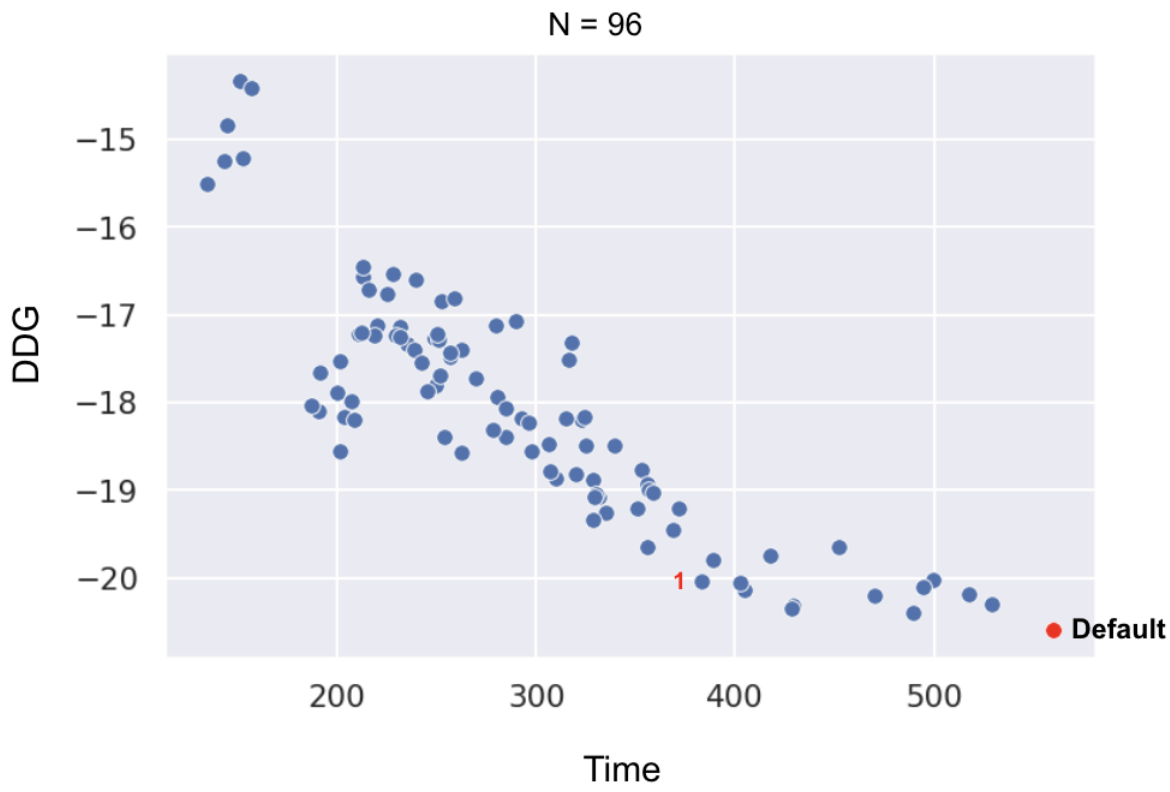

Figure 3.2: Plot of RMSD versus Time comparing each randomly sampled parameter set to the default for a collection of 300 macrocycles. Only the Repeat and NPool parameters were varied in the parameter sets. The default parameter set is highlighted in red, and the 'optimal' parameter set is marked with a red '1'. Only 96 of the 100 parameter sets were successfully completed; 4 experiments were lost to server instability.

3.4 SARS-CoV-2 Main Protease

The SARS-CoV-2 Main Protease has become a prominent drug target in coronavirus research because of the key role it plays in processing the protein products of the virus[12–14]. Due to the high specificity and permeability of macrocycles, I sought to design a macrocycle that would bind to this protein target. Therefore, I applied my improved protocol to dock and design all 300,000 macrocycles against the SARS-CoV-2 Main Protease. Figure 3.3 shows the results of this experiment.

The research lab where I conducted this research, the DiMaio Lab, developed internal criteria for defining good computational designs that are worthy of follow up experimental research. The criteria are as follows:

Surface area of Contact	$\geq$ 150 angstroms
Internal Hydrogen Bonds	$\geq$ 3
Buried Unsaturated Hydrogens	$\leq$ 1
Hydrogen Bond DDG	$\leq$ -1
DDG	$\leq$ -25

My dock-and-design protocol generated a total of over 18,000 macrocycles that surpassed this criteria, thereby providing a promising list of designs for the lab to experimentally investigate going forward.

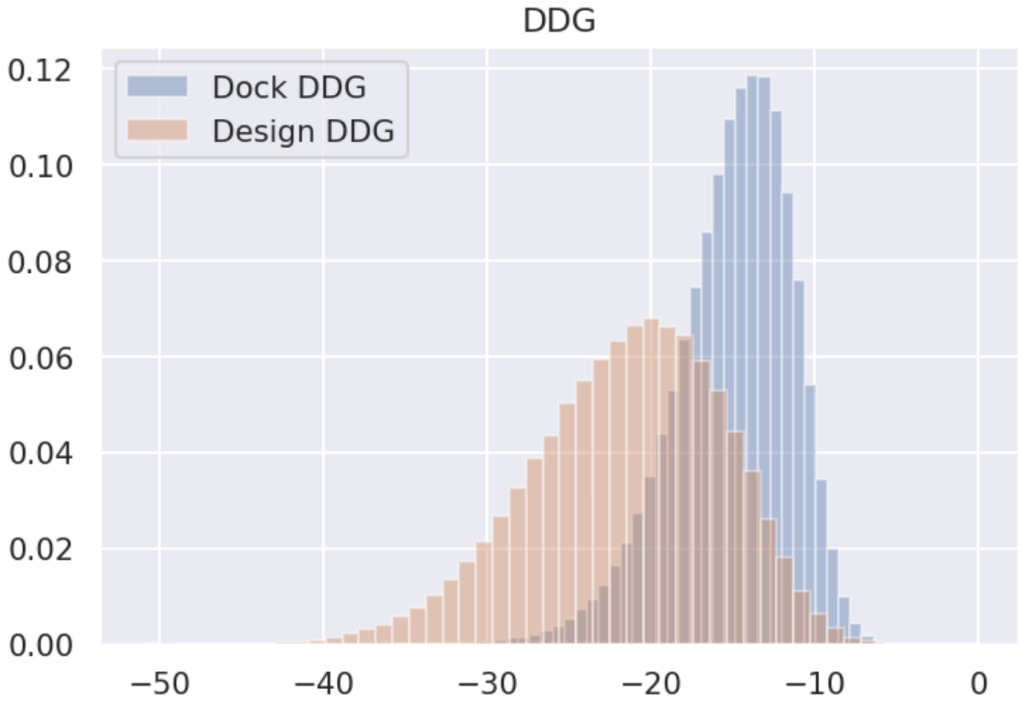

Figure 3.3: Results of the SARS-CoV-2 Main Protease dock-and-design experimen, containing the histogram of DDG values for the 300,000 docked macrocycle backbones (blue), and for the sequence-optimized macrocycle designs (orange).

Chapter 4

SCIENCE IN THE CLOUD

4.1 Introduction

Enzyme engineering possesses the opportunity to revolutionize the chemical manufacturing industry and provide novel solutions for therapeutic development[15, 16]. However, our ability to engineer enzymes is bottlenecked by the low-throughput processes with which we characterize enzyme designs. Current protocols for measuring enzyme efficacy are labor-intensive and slow, often limiting engineering campaigns to testing only dozens or hundreds of unique enzyme designs per round of evolution. This low throughput not only lowers the likelihood of finding valuable designs, it also prohibits the use of sophisticated machine learning algorithms which require large datasets for training. Given the unparalleled performance of machine learning in the field of protein folding, it is very likely that machine learning and large protein datasets will be the backbone of future enzyme engineering efforts. The first step towards realizing this goal, therefore, will be the development of high-throughput methods for enzyme characterization. Ideally this protocol would be reproducible, low-cost, and readily available to the scientific community at large. To achieve this goal, I set out to develop a high-throughput protocol for enzyme characterization on the Emerald Cloud Lab platform.

4.2 Cloud Laboratories

Cloud laboratories are robotically operated facilities which enable researchers to run automated experiments from anywhere in the world[17]. Researchers define their protocols in a symbolic lab language (SSL) which allows the cloud labs to interpret and execute the assays. These labs remove the up-front cost of purchasing and installing the expensive and complex robotics that are often pivotal to high-throughput, large-scale experimentation. Also, by formulating protocols with a symbolic lab language, researchers can more easily

share their protocols with other scientists to improve the reproducibility of their research. These laboratories are a powerful resource for developing high-throughput protocols, and I sought to leverage this tool for my enzyme characterization assay.

4.3 Bioautomation Challenge 2022

The Bioautomation Challenge is a non-profit program that provides researchers with full access to the Emerald Cloud lab platform in order to automate a key assay in their research. The vision of the Bioautomation Challenge is to accelerate academic use of cloud labs to promote reproducibility and big data collection in life science research. Emerald, a cloud lab company, provides software that allows scientists to program their assays in a symbolic lab language, execute experiments in Emerald's automated laboratory, and analyze the experimental measurements, environmental conditions, and robotic diagnostics generated during the experiment. Research teams can apply with a written application describing the protocol they aim to automate, and if accepted are granted one year of access to the Emerald Cloud Lab platform and full funding for assay development and reagents costs. I wrote an application for my automated enzyme characterization protocol and was accepted into the program.

4.4 Enzyme Characterization Protocol

We begin our enzyme characterization protocol by ordering a large library of DNA sequences encoding our enzyme designs. We use electroporation to transfect electrocompetent E. coli cells with our designed DNA and transfer the transfected cells into a 96 well plate. This step requires an electroporator. During the colony streaking step, we leverage a robotic colony picker to streak the transfected cells from the 96 well source plate into two 48-region QTrays containing selective agar. This requires a robotic colony picker that can streak cells from the liquid media of a source plate into the agar of a destination plate. During the colony plating step, we use the robotic colony picker to pick colonies from the incubated QTrays containing our transfected cells into a set of 96 well plates. The number of 96 well plates that will be used as destination plates for colony picking, referred to as N, is a variable that the user will specify on a per-experiment basis. Given the volumes of reagents specified

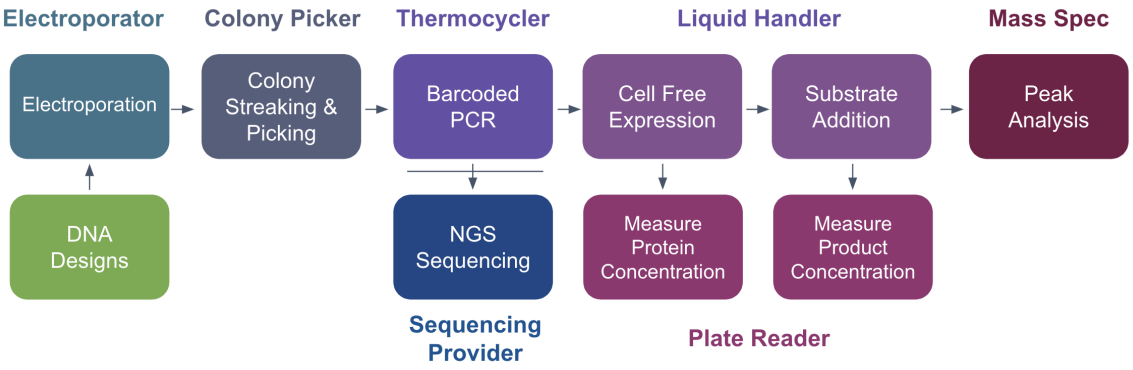

Figure 4.1: Diagram of the Enzyme Characterization Protocol

throughout the five key steps of this protocol, N can range from 1 to up to 20 plates.

During the Barcoded PCR step, we add primers to our 96-well plates and amplify our DNA. The transfected cells containing our DNA will naturally lyse during the PCR protocol. In order to determine the identity of each sequence in each well of our 96-well plates, we place plate and well-specific primer combination in each well prior to amplification. These primers provide an address that, following DNA sequencing, ties each sequence back to a specific plate and well location. The primers that identify the well position are composed of 96 forward primers and 96 reverse primers. Finally, there are the inner primers which are the same for all PCR reactions and are added to every well prior to amplification. This step requires a robotic liquid handler to transfer the primers and PCR reagents, as well as an automated thermocycler to run the PCR protocol. For the DNA Sequencing step we transfer an aliquot from each well of our 96-well plates of amplified DNA into a single 96-well plate, which is sequenced by NGS from an external sequencing provider. It will involve one set of pooled samples for each 96-well plate, which will be sequenced with 2x150bp reads.

With the amplified DNA contained in our 96-well plates, we now add and incubate a cell free expression kit to synthesize our proteins. Each expressed protein has an attached GFP tag to measure protein concentration. With our proteins expressed, we place the plate into a fluorescent plate reader to measure the protein concentration for each well. With our enzymes expressed, we now add in the substrate and any necessary cofactors to start reaction

catalysis. Following catalysis of the enzymatic reaction, we analyze our samples with mass spectroscopy in order to determine the concentrations of the product and thereby measure the turnover rate of the enzyme. For fluorescent products, we also have the opportunity to use a plate reader to quantify product concentrations and calculate the enzyme turnover.

4.5 Protocol Benefits

Mass spectrometers can measure substrate and product concentrations with high sensitivity and precision, providing a high-quality measurement of enzyme turnover[18]. In addition, this sensitivity allows for low-volume, low-concentration usage of substrates and reagents, lowering experimental costs. Mass spectroscopy is also theoretically generalizable across most enzyme reactions that involve small molecule substrates and products; as long as the device can detect the molecule within the sample, the assay can be run. Additionally, with the introduction of high-throughput mass spectrometers, such as the Bruker MicroFlex LRF, Bruker RapiFlex, Agilent Rapidfire 400, and the SciEx Echo MS, the speed of these measurements match the throughputs we hope to achieve.

The benefits of the protocol are as follows:

- Higher Throughput
- Ease of Use
- Transferability
- Volume Miniaturization

The increased throughput of this automated protocol is crucial for building large, high-quality enzyme datasets that will serve as the bedrock for training machine learning models. Having access to a larger screening library allows us to sample a higher proportion of enzyme sequence space, which will significantly enhance our ability to identify enzymes with new and improved properties and functions. Further, automating this protocol takes a significant amount of work off the hands of the scientists involved. By doing so, we free up time which may be better spent on the intellectual tasks of improving experimental design and analyzing data. Additionally, by removing the barrier of intensive human labor, we can make enzyme

engineering a more routine task, thereby leading to even more publicly available enzymatic datasets.

Automating and sharing this protocol also makes it immediately available to the scientific community at large. Typical sharing of protocols through published materials requires scientists to read through the protocol, obtain access to the necessary machines, and work through the protocol one step at a time to ensure that the results are similar. More often than not, researchers then must struggle with the reproducibility problems caused by unmentioned variables which lead to different results. The automation of this protocol on Emerald's platform, however, would allow researchers anywhere in the world to reproduce and build off of this work at the click of a button.

Lastly, it is possible that the sensitivity of our mass spectroscopy approach may allow us to use even lower volumes than we have estimated in our protocol. By leveraging acoustic liquid handling to miniaturize our experiments, we could experience dramatically decreased costs per experiment, and thus be able to test more designs for a given budget.

4.6 Protocol Cost and Throughput

By discussing the protocol details with experimental biologists and automation vendors, and by carefully analyzing each step of the process, I was able to generate an estimate for the cost and throughput of executing ten thousand enzyme characterization experiments with my protocol, using the robotic equipment discussed above. As shown in Figure 4.2, it is theoretically possible to characterize ten thousand enzymes in under one week for less than $50,000.

4.7 Protocol Development

With the Emerald Cloud Lab access I was provided by the Bioautomation Challenge, I focused on implementing three sub-protocols within the enzyme characterization workflow. More specifically, I worked on automating barcoded PCR amplification, cell-free expression followed by protein concentration measurements, and mass spectroscopy standard-curve analysis for small molecule substrates. Once I became proficient in writing protocols in Emerald's software, I was able to program the assays in Emerald's symbolic lab language.

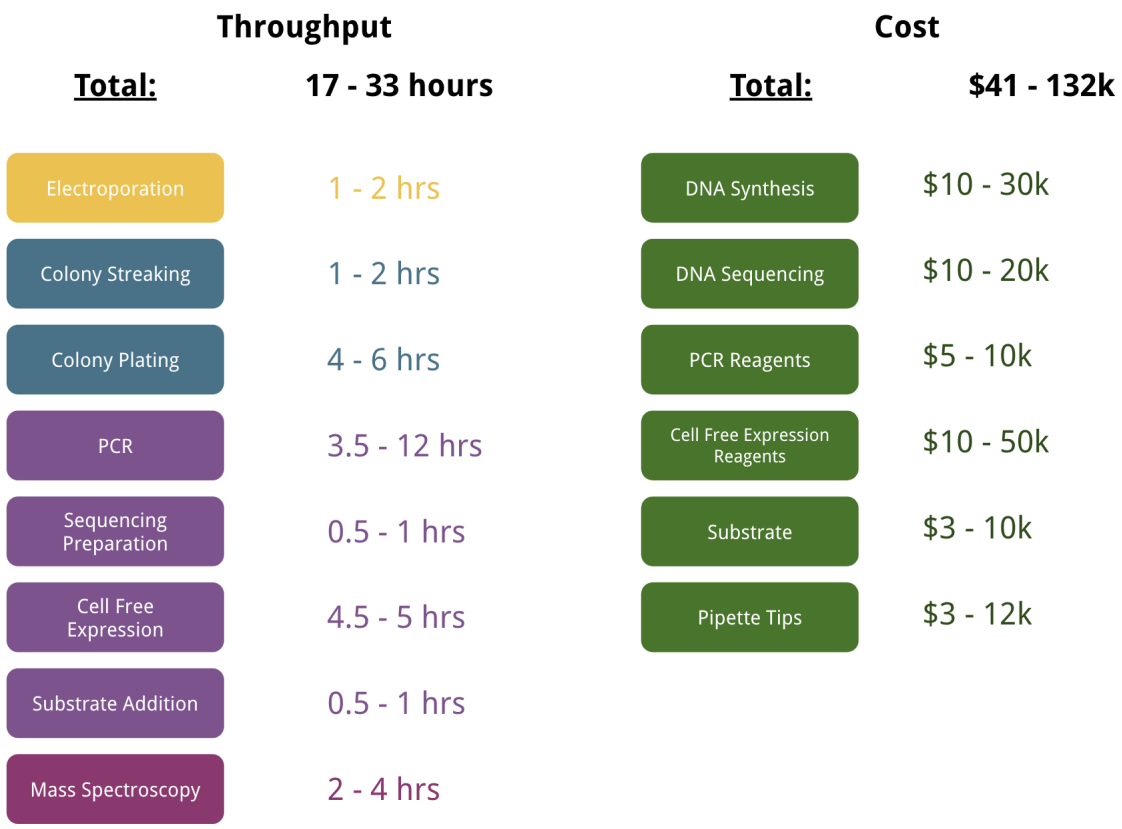

Figure 4.2: Cost and Throughput projections for characterizing 10,000 enzymes with the Enzyme Characterization Protocol

Further, I shipped my reagents to the facility, ran a series of assay development experiments, and analyzed the resulting data across the three sub-protocols. While the enzyme characterization protocol requires more assay development than I could provide within the three-month period I could dedicate to the project, I demonstrated meaningful progress towards this goal and in doing so solidified my belief that cloud laboratories are a powerful technology that will significantly improve the reproducibility, accessibility, and productivity of experimental life science research.

Chapter 5

DEEP LEARNING FOR METABOLIC SIMULATIONS

5.1 Introduction

Systems involving multiple bodies whose behavior is governed by differential equations do not possess any known closed-form solution that allows their behavior in the past or future to be modeled quickly. Currently, time-consuming simulations must be run to calculate the state of these systems from a given time point t to a given time point $t + p$. In the case of biological networks, where properties of the system can be measured experimentally with the desire to infer the parameters which govern the system (e.g. kinetic rates of enzymes or signaling proteins), these costly computations limit our ability to infer system parameters.

It is possible that there are mathematical equations that allow us to simulate these systems far faster than the differential equations we now use; however, such equations are currently not known. It is further possible that neural networks trained with gradient descent to model the states of these systems may uncover these equations during network training. Therefore, it is theoretically possible that neural networks could provide substantial speed increases to the modeling of n-body systems. The following work will investigate this question in state systems where units transfer between given states based on a graph described by differential equations. These systems are representative of biological networks, and here we focus on systems representing metabolomes.

Systems of differential equations representing a metabolome can be defined and simulated with Tellurium, a python library for building, simulating, and analyzing models in systems biology[19]. Figure 5.1 illustrates an example of defining a simulating a metabolic system in Tellurium. The core information that is relevant to the simulation are the following:

X_t: The vector containing $[x1_t, x2_t, \ldots, xn_t]$, where xi_t represents the total concentration of metabolite x_i present in the system at this given time point t.

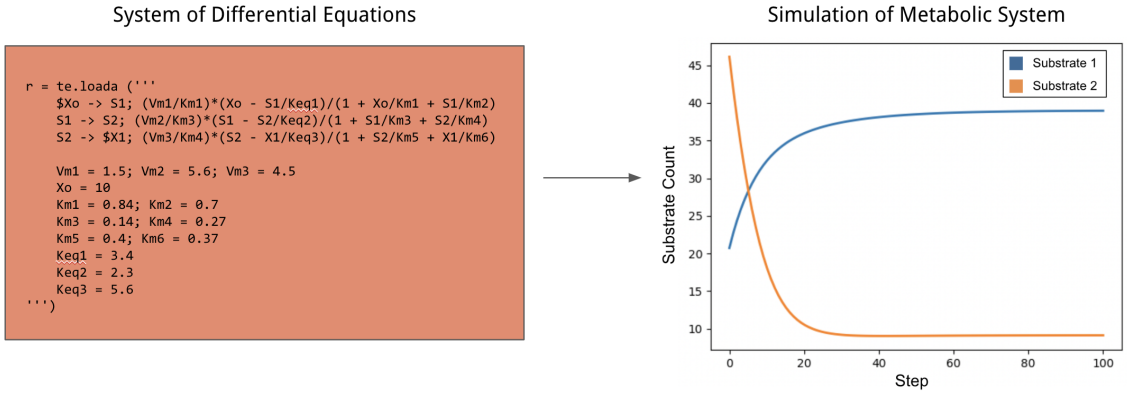

Figure 5.1: Example of defining and simulating a system of differential equations with the Tellurium systems biology software.

K: The vector containing $[k1, k2, \ldots, kn]$ where k_i represents a system parameter relevant to the equations which govern the system, such as $VMax$, Km, and Keq parameters.

D: The list of equations which govern the system, detailing how the X_{t+1} will be calculated given X_t and K.

The list of differential equations, D, will be held equal for the experiments. In order to train a deep learning model to simulate a metabolic system, we provide the network with X_t and K and request that it calculate X_{t+s}. The variable s refers to the step size of the dataset, that is to say how many differential equation calculations have occurred since X_t and X_{t+s}.

5.2 Building the Tellurium Dataset

The metabolic system we defined for these experiments are a linear six-step pathway governed by the following equations:

$x0 \rightarrow s1; (Vm1/Km1) * (x0 - s1/Keq1)/(1 + x0/Km1 + s1/Km2)$

$s1 \rightarrow s2; (Vm2/Km3) * (s1 - s2/Keq2)/(1 + s1/Km3 + s2/Km4)$

$s2 \rightarrow s3; (Vm3/Km5) * (s2 - s3/Keq3)/(1 + s2/Km5 + s3/Km6)$

$s3 \rightarrow s4; (Vm4/Km7) * (s3 - s4/Keq4)/(1 + s3/Km7 + s4/Km8)$

$s4 \rightarrow s5; (Vm5/Km9) * (s4 - s5/Keq5)/(1 + s4/Km9 + s5/Km10)$

$s5 \rightarrow x1; (Vm6/Km11) * (s5 - x1/Keq6)/(1 + s5/Km11 + x1/Km12)$

The variables x0 and x1 represent the source and sink, respectively. Variables s1 to s5 are the metabolites within the system. Variables *Vm1* to *Vm6*, *Keq1* to *Keq6*, and *Km1* to *Km12* are the system parameters.

Each datapoint in the dataset has the following structure:
Input: Initial substrate counts (s1, s2, s3, s3, s4, s5); Model parameters (*Vms*, *Kms*, *Keqs*, $x0$, $x1$)
Output: Next time-step substrate counts (y1, y2, y3, y4, y5)

5.3 Modeling Training and Simulations

As demonstrated in Figure 5.2, the model was successfully trained to perform next-state prediction for the simulated metabolic systems. While a single, next-state prediction was a valuable indicator of performance, it was pertinent to evaluate the model's ability to simulate a metabolic system over multiple time steps. To do this, I provided a randomly initialized set of system parameters and starting substrate concentrations to the model and calculated the predicted substrate concentrations for the next time step. The outputted metabolite concentrations were iteratively inputted back into the model to simulate the system over a total of one hundred time steps. This neural network-simulated system was compared to a Tellurium simulation with the same starting conditions to evaluate the quality of the model's simulation. Figure 5.3 shows two examples comparing the simulations provided by Tellurium and the neural network. The model's results were compared with those of Tellurium across one thousand randomly sampled parameter sets; the final concentrations of the system metabolites were compared between the two approaches. The neural network's final substrate concentrations had an average difference of 7.8% with the Tellurium-simulated values. The histogram of differences between the final substrate concentrations of these two approaches is shown in Figure 5.4.

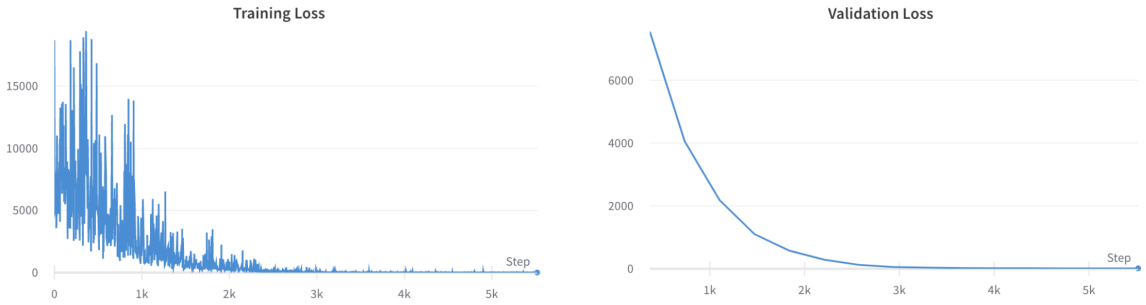

Figure 5.2: Training and Validation loss for the neural network's next-state prediction task.

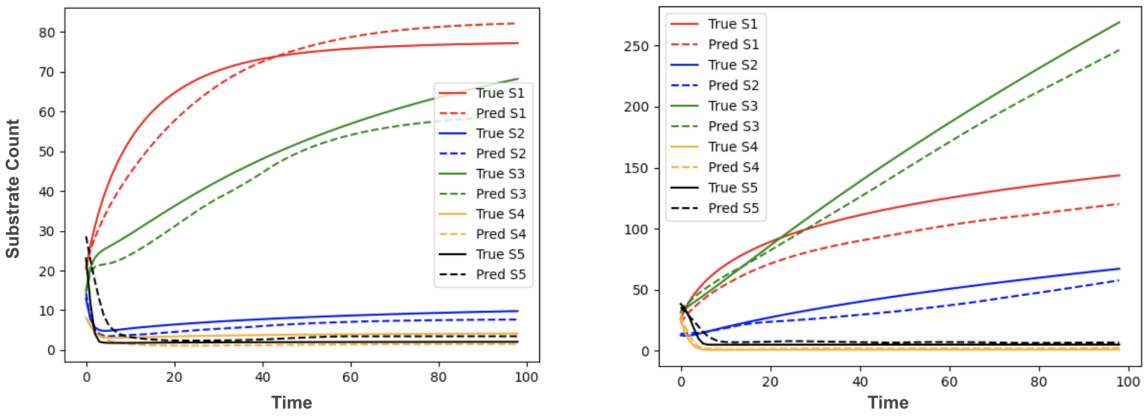

Figure 5.3: Two example simulations demonstrating the similarity between the Tellurium simulation (solid lines) and the neural network simulation (dotted lines) for the changes in concentration of the five system substrates over the course of one hundred time steps.

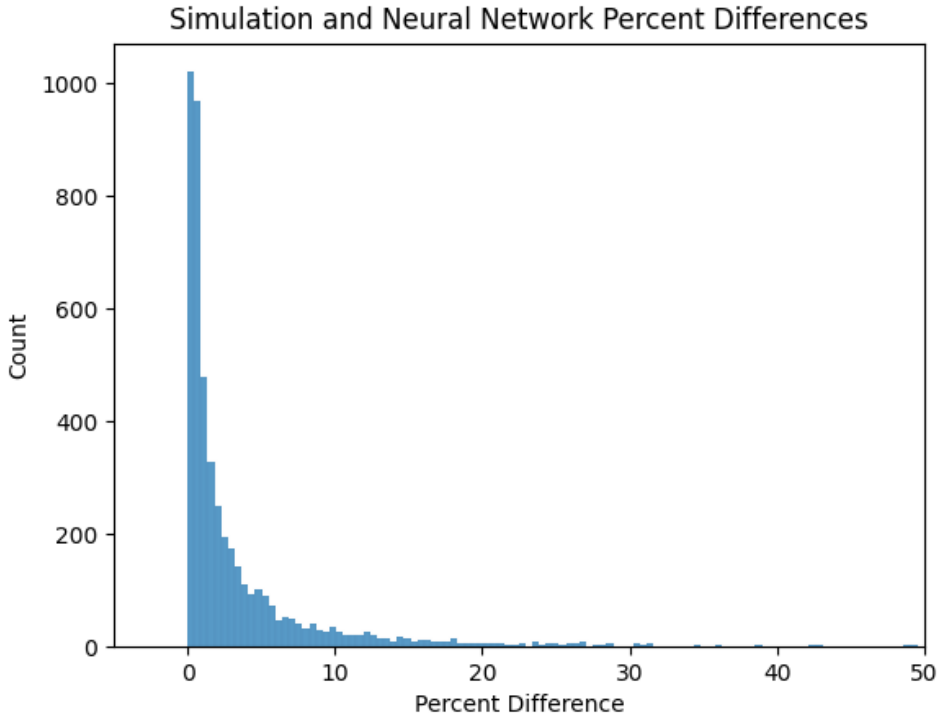

Figure 5.4: Histogram of percent differences between the final substrate concentrations in the neural network and Tellurium simulations. One thousand metabolomes were initialized with randomly sampled system parameters and starting substrate concentrations, and were simulated for one hundred time steps by Tellurium and the neural network independently.

Chapter 6

CONCLUSION

Deep learning promises to bring significant advances to the modeling and engineering of proteins and the many other biomolecules of the cell. Deep learning also poses to advance our understanding and simulations of biological systems, from metabolomes to entire cellular networks. The consolidation of these two approaches, of leveraging deep learning to both model the building blocks of the cell and to combine them into representations of entire cellular systems, is a promising avenue for future research.

Furthermore, academic access to cloud science facilities could be a significant boon for experimental and computational biology. The throughput and precision of these robotic systems combined with specificity and shareability of the symbolic lab language protocols that control them offer to dramatically improve the productivity, accessibility, and reproducibility of life science research. The scalability of protocols executed on these automated platforms could further enable the creation of large, high-quality biological datasets, further setting the stage for the successful development of biologically-focused deep learning models.

By combining deep learning-driven design with robotically-automated experimentation, the future of biological engineering will unlock a vast array of industrial and healthcare innovations, thereby creating a healthier and more productive tomorrow.

BIBLIOGRAPHY

1. Han, Y. *et al.* Covalently Engineered Protein Minibinders with Enhanced Neutralization Efficacy against Escaping SARS-CoV-2 Variants. *Journal of the American Chemical Society* **144.** PMID: 35212528, 5702–5707. eprint: https://doi.org/10.1021/jacs.1c11554. https://doi.org/10.1021/jacs.1c11554 (2022).

2. Cao, L. *et al.* De novo design of picomolar SARS-CoV-2 miniprotein inhibitors. *Science* **370,** 426–431. eprint: https://www.science.org/doi/pdf/10.1126/science.abd9909. https://www.science.org/doi/abs/10.1126/science.abd9909 (2020).

3. De novo design of a fluorescence-activating -barrel. *Nature* **561,** 485–491. eprint: https://www.nature.com/articles/s41586-018-0509-0. https://www.nature.com/articles/s41586-018-0509-0 (2018).

4. Jumper, J. *et al.* Highly accurate protein structure prediction with AlphaFold. *Nature* **596,** 583–589. https://doi.org/10.1038/s41586-021-03819-2 (July 2021).

5. Dauparas, J. *et al.* Robust deep learning–based protein sequence design using ProteinMPNN. *Science* **378,** 49–56. eprint: https://www.science.org/doi/pdf/10.1126/science.add2187. https://www.science.org/doi/abs/10.1126/science.add2187 (2022).

6. Baek, M. *et al.* Accurate prediction of protein structures and interactions using a three-track neural network. *Science* **373,** 871–876. https://doi.org/10.1126/science.abj8754 (Aug. 2021).

7. Wang, J. *et al.* Scaffolding protein functional sites using deep learning. *Science* **377,** 387–394. https://doi.org/10.1126/science.abn2100 (July 2022).

8. Hosseinzadeh, P. *et al.* Comprehensive computational design of ordered peptide macrocycles. *Science* **358,** 1461–1466. https://doi.org/10.1126/science.aap7577 (Dec. 2017).

9. Bhardwaj, G. *et al.* Accurate de novo design of hyperstable constrained peptides. *Nature* **538,** 329–335. https://doi.org/10.1038/nature19791 (Sept. 2016).

10. Bhardwaj, G. *et al.* Accurate de novo design of membrane-traversing macrocycles. *Cell* **185,** 3520–3532.e26. https://doi.org/10.1016/j.cell.2022.07.019 (Sept. 2022).

11. Park, H., Zhou, G., Baek, M., Baker, D. & DiMaio, F. Force Field Optimization Guided by Small Molecule Crystal Lattice Data Enables Consistent Sub-Angstrom Protein–Ligand Docking. *Journal of Chemical Theory and Computation* **17,** 2000–2010. https://doi.org/10.1021/acs.jctc.0c01184 (Feb. 2021).

12. Ullrich, S. & Nitsche, C. The SARS-CoV-2 main protease as drug target. *Bioorganic &amp Medicinal Chemistry Letters* **30,** 127377. https://doi.org/10.1016/j.bmcl.2020.127377 (Sept. 2020).

13. Mengist, H. M., Dilnessa, T. & Jin, T. Structural Basis of Potential Inhibitors Targeting SARS-CoV-2 Main Protease. *Frontiers in Chemistry* **9.** https://doi.org/10.3389/fchem.2021.622898 (Mar. 2021).

14. Zhang, L. *et al.* Crystal structure of SARS-CoV-2 main protease provides a basis for design of improved -ketoamide inhibitors. *Science* **368,** 409–412. https://doi.org/10.1126/science.abb3405 (Apr. 2020).

15. Wu, S., Snajdrova, R., Moore, J. C., Baldenius, K. & Bornscheuer, U. T. Biocatalysis: Enzymatic Synthesis for Industrial Applications. *Angewandte Chemie International Edition* **60,** 88–119. https://doi.org/10.1002/anie.202006648 (Aug. 2020).

16. De la Fuente, M. *et al.* Enzyme Therapy: Current Challenges and Future Perspectives. *International Journal of Molecular Sciences* **22,** 9181. https://doi.org/10.3390/ijms22179181 (Aug. 2021).

17. Arnold, C. Cloud labs: where robots do the research. *Nature* **606,** 612–613. https://doi.org/10.1038/d41586-022-01618-x (June 2022).

18. Pluchinsky, A. J., Wackelin, D. J., Huang, X., Arnold, F. H. & Mrksich, M. High Throughput Screening with SAMDI Mass Spectrometry for Directed Evolution. *Jour-*

nal of the American Chemical Society **142,** 19804–19808. https://doi.org/10.1021/jacs.0c07828 (Nov. 2020).

19. Choi, K. *et al.* Tellurium: An extensible python-based modeling environment for systems and synthetic biology. *Biosystems* **171,** 74–79. https://doi.org/10.1016/j.biosystems.2018.07.006 (Sept. 2018).