NAVIGATING PROTEIN FITNESS LANDSCAPES WITH MACHINE LEARNING AND BIOLOGICAL INSIGHTS

By

Mehrsa Mardikoraem

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Chemical Engineering—Doctor of Philosophy

2024

ABSTRACT

Proteins are essential biomolecules addressing challenges in medicine, nanotechnology, and industry. Protein engineering designs and optimizes these molecules for specific functions, such as catalyzing reactions or facilitating drug delivery. However, designing proteins with desired properties is extremely challenging due to unpredictable mutation effects and complex fitness landscapes, which depict the relationship between a protein's sequence, structure, and function. Traditional methods like directed evolution and rational design have limitations in exploring vast sequence spaces and modeling amino acid interactions. Recent advances in machine learning (ML) and the increasing availability of biological data have shifted protein engineering from a theory-driven to a data-driven approach. Despite progress, challenges remain, such as capturing nuanced protein behaviors under distinct biological conditions, enhancing data quality and diversity, and developing models that handle complex protein-ligand interactions.

This dissertation explores innovative protein engineering approaches by integrating machine learning (ML) and computational tools with biological insights. It addresses designing proteins with desired properties, enhancing their numerical representations, and modeling protein-drug interactions. Also, methodologies are developed to generate new-to-nature proteins with desired properties and optimize experimental design strategies. Protein representation methods were optimized by combining traditional encodings with protein sequence language models. This ensemble approach achieved a 94% F1 score, enhancing sequence-function predictions by capturing diverse protein fitness aspects. Performance varied for larger proteins and different protein properties suggesting the need for specialized biologically aware ML methodologies. Additionally, the study addressed the critical challenge of modeling protein-drug interactions, focusing on organic anion-transporting polypeptides (OATPs). OATPs are crucial for drug absorption and distribution, and significantly impact drug efficacy and safety. A comprehensive pipeline was developed, combining AlphaFold structure prediction, molecular docking, and a novel Heterogeneous Graph Neural Network model named HIPO. This model captured complex inter and intra-molecular interactions, outperforming existing methods for OATP inhibition prediction. By identifying key drug

attributes influencing these interactions, the study demonstrated the effectiveness of structure-based approaches in elucidating protein-drug interactions, contributing to advancements in drug development and toxicity prediction. In addition, key drug attributes affecting these interactions were identified, emphasizing the need for structure-based methods. Advancing beyond protein representation and drug interaction modeling, this work addresses the generation of novel protein sequences with desired properties. It integrates evolutionary information into generative ML models through a dual approach: combining ancestral sequence reconstruction (ASR) with a Variational Autoencoder (VAE) for sequence generation and utilizing ASR-derived data to fine-tune language models for improved protein representations. This methodology explores sequences from evolutionary history not observed in modern organisms, accessing a vast, unexplored protein space. This data-centric approach leverages ASR to provide a rich source of information beyond extant species, emphasizing the crucial role of biologically diverse datasets in machine learning frameworks. The result is the generation of proteins with enhanced diversity and stability, particularly in thermal properties.

By synthesizing evolutionary insights with advanced ML techniques, this work expands the possibilities for engineering proteins with unprecedented characteristics, In conclusion, this thesis presents a comprehensive framework integrating ML with protein engineering, advancing the design and optimization of biomolecules, and addressing specific biological challenges for improved therapeutics and diagnostics applications.

ACKNOWLEDGEMENTS

The PhD journey has been one of my most challenging yet enlightening pursuits, filled with valuable life lessons. Looking back on the past five years, I see myself stronger and more adaptable to life's challenges. This experience has taught me patience, the importance of trusting the process, and keeping my curiosity alive while preserving the joy of learning. None of this would have been possible without the support of my mentors, colleagues, and friends. I owe my deepest gratitude to my PhD advisor, Dr. Daniel Woldring. Under his guidance, I've learned to maintain curiosity, actively pursue optimal solutions, and embody the essence of a true scientist. Dr. Woldring's support throughout my PhD journey has been nothing short of extraordinary. Our lengthy brainstorming meetings and discussions about future methodologies have been invaluable, and I will cherish these memories as I move forward in my career.

I'm deeply grateful to my PhD committee members, Dr. S. Patrick Walton, Dr. Jose L. Mendoza-Cortes, and Dr. Arjun Krishnan, for their invaluable mentorship and insightful feedback. Their critical guidance, especially during my comprehensive exam, was instrumental in shaping my final projects and refining my graduation goals. Their expertise and dedication have significantly enriched my academic journey and contributed immensely to my growth as a researcher. I extend my heartfelt appreciation to my wonderful lab mates in the Woldring lab, whose professionalism and punctuality made our collaborations both enjoyable and productive. Their camaraderie and support have been integral to my research journey. Equally important has been the invaluable support and resources provided by the Chemical and Material Science Department, which have been crucial in facilitating my research and academic progress throughout my PhD studies. I'm also thankful for my dear friends Reza, Sunanda, and Parisa. Your friendship has been a source of comfort and openness as always.

My family deserves special mention - my parents and my two lovely sisters, Mahsa and Melina. Your constant support, patience, and presence have meant everything. From my mother, I learned determination, and from my father, ambition. Thank you for raising me to believe in myself and pursue my goals.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Proteins are biological machines involved in nearly all life processes [1, 2, 3, 4]. They are composed of chains of amino acids that fold into three-dimensional structures, allowing them to perform essential biological functions. The diverse chemical properties of the twenty natural amino acids enable proteins to maintain cellular structure, facilitate chemical reactions [5, 6, 7], and transport drugs [8, 9]. Protein engineering is vital for modifying proteins to meet modern industrial and medical needs by directing their natural evolution [10, 11, 12, 13]. This field uses advanced techniques like recombinant DNA technology [14], CRISPR-Cas [15], and high-throughput screening [16] to design proteins with specific functions, such as developing antiviral peptides, engineering antibodies, and creating personalized therapies. Protein engineering enhances enzyme specificity and functionality under various conditions and helps develop protein therapeutics with improved pharmacokinetics, pharmacodynamics, and reduced immunogenicity [17, 18, 19, 20, 21].

1.1 Fundamentals of Protein Engineering Guided by the Sequence-Structure-Function Paradigm

Protein fitness [22, 23, 24], which measures how well a protein performs a specific task, is influenced by its amino acid sequence, structural stability, and interactions with other molecules (e.g., regulatory interactions [25], transport interactions [9], inhibition [26], and binding [27]). Protein engineering alters protein functions through experimental methods inspired by natural selection, aiming to yield high-fitness proteins that perform well in relevant tests. The interplay between protein sequences and their functional outcomes highlights the transformative potential of protein engineering. This sets the groundwork for innovative methods to tackle biological problems, guided by the fundamental paradigm called the "sequence-structure-function" relationship [28, 29]. However, this is challenging due to the vast number of possible mutations. For example, a small protein with 100 amino acids has 20^{100} possible sequences, more than the number of atoms in the observable universe. Additionally, the complex relationship between protein sequences and

their fitness creates a sparse and rugged fitness landscape, making it difficult to identify optimal mutations [30, 31]. This rugged landscape means that even one mutation can drastically affect a protein's function, causing significant variability in fitness. Ultimately, the challenges posed by the vast sequence space and rugged fitness landscape underscore the need for more strategic approaches in protein engineering.

Protein engineers approach biological problems by employing a diverse array of methodologies, all guided by the foundational concept of protein sequence-structure-function relationships[28, 29]. This concept articulates that the sequence of amino acids dictates a protein's structure, and in turn, its structure determines its function. A key task in this field is structure prediction, exemplified by breakthroughs like AlphaFold [32], which reveals how sequences fold into intricate three-dimensional shapes, enabling advancements in drug design and disease understanding. Conversely, when a function or interaction is predefined, engineers focus on designing novel (*de novo*) proteins tailored to fold into structures that execute specific functions [33]. This creative process addresses the challenge of the vast and sparse search space of amino acid sequences. Additionally, protein engineering extends to fitness optimization, aimed at refining an existing protein's stability and activity for enhanced function[34, 35]. Collectively, these efforts exemplify how leveraging a broad spectrum of information beyond mere structure can lead to groundbreaking advances in protein engineering.

Addressing complex biological challenges involves tools like rational design [36, 37, 38] and directed evolution (DE) [39]. Rational design utilizes already-known information—including protein sequences, structures, phylogenetic data, knowledge of active sites, and predictions from simulations—to predict new, functional proteins. Techniques like X-ray crystallography [40], NMR spectroscopy [41], cryo-electron microscopy (Cryo-EM) [42], and machine learning (ML) models like AlphaFold [32] further aid this process. For example, rational design has created novel enzymes to break down antibiotic-resistant bacterial walls, improving drug efficacy, and stabilizing therapeutic proteins for enhanced drug delivery [43, 44, 45]. DE mimics natural selection in the lab to iteratively improve proteins. Starting with a parent protein, random mutations create a library of

variants, which are screened for desirable traits. DE has been impactful in enhancing the binding of affibody proteins for improved HER2-positive breast cancer diagnosis [46], developing therapeutic protein candidates with selective biological impact for treating a wide array of human disorders including cancer and autoimmune/inflammatory diseases [47, 48], and improving nitrilases for greener pharmaceutical synthesis [49]. This approach has significantly advanced therapeutic efficacy, diagnostic accuracy, and industrial sustainability. Combining rational design and DE has led to breakthroughs like enzymes for plastic degradation [50]. These methods are now enhanced by machine learning and data-driven approaches. The fusion of traditional methods with computational models and high-throughput screening is paving the way for innovative solutions in medicine, diagnostics, industry, and environmental science.

1.2 A New Era in Protein Engineering: The Integration of Machine Learning

Recent advancements in ML techniques, along with the growing availability of biological data, have fundamentally transformed protein engineering, shifting it from a theory-driven to a data-driven discipline [51, 52]. Traditionally, theory-driven approaches relied on domain knowledge and mathematical models to capture the attributes and physics of proteins. However, ML methods focus on modeling observed data, which is particularly beneficial for the complex, high-dimensional challenges encountered in protein engineering [53]. ML models extract meaningful features from labeled protein sequence data, significantly enhancing predictive accuracy and generalization [54]. These models identify patterns and relationships within the data, enabling the forecasting of protein properties such as binding affinity, solubility, and stability [51, 55, 56, 57, 58, 59]. Despite the abundance of large-scale sequence data, the lack of experimental fitness annotations necessitates the use of more advanced ML models, such as self-supervised and unsupervised methods. Self-supervised learning techniques, including large language models (LLMs) [60, 61] like AlphaFold [32] and Evolutionary Scale Modeling (ESM) [62], leverage inherent patterns in unlabeled sequences to decipher the biological "language" of proteins, enabling accurate predictions of their structure and function [62, 63]. Generative models, such as ProGen [64] and ProtVAE [65], ProtGPT2 [66], and ProteinGAN [67], learn the underlying data distribution to create novel

proteins with desired characteristics. They use various neural network architectures, including variational autoencoders, transformers, and adversarial neural networks [68]. LLM-based models like TAPE [69], UniRep [63], and ProtTrans [70] capture dependencies within protein sequences, resulting in rich semantic and syntactic numerical representations. By integrating these advanced ML techniques with traditional protein engineering methods, researchers can effectively navigate the high-dimensional landscape of protein design and engineering. This comprehensive toolkit not only enhances our understanding of protein data but also paves the way for innovative solutions in industrial, medical, and research applications.

The continued developments in ML have significantly enhanced our ability to analyze extensive datasets, predict protein mutations, and generate novel functional proteins. However, these techniques often face challenges due to the complex nature of biological problems. In protein engineering, effectively predicting protein properties or designing novel proteins requires integrating biological knowledge with ML tools. The protein family and specific property being studied can drastically influence the problem-solving strategy [24]. While protein language models have been successful, training simpler ML models on one protein family at a time with evolutionary tools like alignment has sometimes yielded more promising results [69, 71]. Additionally, LLMs trained on public datasets may lack the high-quality, specialized data needed for protein engineering, where data quality and diversity are crucial for reliable predictions [72, 73, 74]. Furthermore, the property under investigation, whether stability or binding, also dictates the complexity of the ML models required [24]. By identifying and addressing these issues, we can achieve considerable progress in the field, paving the way for more precise and effective protein engineering solutions.

1.3 Thesis Overview: ML Solutions for Biological Challenges

This thesis investigates optimization methods for navigating protein fitness space by integrating ML and computational tools to address domain-specific challenges. The five chapters tackle critical issues such as reducing data noise, optimizing ML algorithms for next-generation sequencing (NGS) data [75], and employing advanced techniques in protein engineering. They delve into the use of generative models, data-centric approaches, and evolutionary algorithms to boost ML

performance and improve protein fitness predictions. Chapter Two (Mardikoraem, Yeast Surface Display,2022) [76] explores the latest ML algorithm advancements and their application to predict sequence mutations before experiments, integrating ML into directed evolution techniques. Chapter Three (Mardikoraem, Pharmaceutics 2023) [24] focuses on enhancing ML model performance by refining protein representations, managing specific properties like sequence length, and addressing imbalanced datasets. Chapter Four examines Organic Anion Transporter Proteins (OATPs), crucial in drug development, using multiple ML algorithms to predict their behavior and highlight important physicochemical properties despite the scarcity of experimental structures. Chapter Five (Mardikoraem, bioinformatics, 2023) [68] provides a comprehensive overview of ML-driven protein engineering, particularly language modeling and generative models for protein sequence modeling, identifying gaps and familiarizing researchers with this specialized field. Chapter Six (Mardikoraem, ICLR 2024 Workshop on Generative and Experimental Perspectives for Biomolecular Design, 2024) [74] emphasizes the importance of high-quality domain-specific data and data-centric methods, highlighting how evolutionary information and ancestral sequence reconstruction can enhance generative models. Overall, this thesis explores the integration of advanced ML techniques, generative models, data-centric approaches, and evolutionary algorithms to contribute to the ongoing advancements in protein engineering.

In Chapter Two, I developed a robust methodology for effectively mapping next-generation sequencing (NGS) [75] data to experimental annotations from high-throughput techniques. This incorporates cleaning the data, preprocessing, algorithm selection, and post-processing. For cleaning, I carefully chose and combined multiple software (PEAR [77], SEQTK [78]) to minimize experimental noise and increase the signal-to-noise ratio (SNR) [79] in the initial data. Then, we do preprocessing, such as identifying how many times sequences appeared in the obtained data and ranking them from highly functional to low functional. Other ML processing methods such as the choice of protein numerical representations to make it compatible with ML and ML algorithm selection were explored. The most significant contribution of this book chapter aside from a comprehensive guideline on the implementation of ML in NGS data, is how simple ML

models can contribute to protein engineering paradigms. ML-driven methodologies offer several key advantages: (i) They provide a better starting point for directed evolution techniques, which are highly path-dependent and rely on the initial sequence. (ii) They help explore previously uncharted regions of fitness space, revealing the relationship between sequence data and its properties. (iii) They assist in experimental design, including the design of oligonucleotides.

In Chapter Three [24], I investigated the performance of language models in predicting protein properties, comparing them to simpler representations like one-hot encoding and physicochemical encoding. This study aimed to understand how different representations capture various aspects of protein fitness. I proposed an ensemble technique that combines these methods to enhance ML algorithm performance. The study evaluated four representation methods: one-hot encoding, physicochemical encoding, UniRep [63], and ESM [62]. It also examined the impact of protein size, revealing significant differences between small proteins ($\leq$120 amino acids) and large proteins (400–1500 amino acids) . For small proteins, one-hot encoding significantly boosted classification metrics, achieving a mean F1-score of 94% when combined with other encodings. However, it proved problematic for large proteins due to sparsity. The findings demonstrated that SMOTE [80] outperformed undersampling for imbalanced datasets when using one-hot, UniRep, and ESM representations. Ensemble learning increased predictive performance by 4 % in affinity-based datasets and achieved high F1 scores in stability prediction using ESM. By employing advanced sampling techniques and multiple-criteria decision-analysis (MCDA) [81], the study ensured statistical rigor. This approach not only benchmarks and compares protein representation models but also highlights the benefits of ensemble modeling in capturing distinct aspects of protein fitness. Critical features for distinguishing functional proteins were identified, providing a solid foundation for future research in protein engineering and significant advancements in predictive modeling.

Another significant focus of this thesis in Chapter Four is overcoming the challenges associated with OATPs [82, 83]. Drug-drug interactions (DDIs) pose substantial challenges in pharmacology, potentially leading to severe side effects and reduced therapeutic efficacy. The FDA's guidelines emphasize the importance of predictive DDI models in drug regulation. While predictive models

for metabolic DDIs involving cytochrome P450 enzymes are well-developed, there is a notable gap in transporter-mediated DDI (tDDI) models, particularly for OATPs. These transport proteins are crucial in ADME processes (absorption, distribution, metabolism, and excretion) and play a vital role in drug disposition and pharmacokinetics. The lack of high-resolution structural data for OATPs and significant variability in available in vitro data have historically limited the development of accurate predictive models. This research employs a multimodal approach, combining advanced neural network architectures like GNNs (graph neural networks) and molecular modeling tools to integrate diverse information sources effectively. GNNs model complex molecular interactions and spatial relationships within proteins, enabling the development of novel, generalizable methods for modeling OATP-drug interactions and filling a critical gap in transporter-DDI prediction models.

Chapter Five delves into advanced sequence modeling techniques, expanding on the previous chapter's focus on model improvement and language model performance in predicting protein properties, which offers a more holistic approach to examining current ML model development and its success in the protein engineering domain [68]. With the rapid expansion of high-throughput omics technologies, the volume of protein sequence data has surged, necessitating sophisticated ML methods. This chapter systematically reviews promising neural networks for protein sequences, covering core mathematical concepts and their implementation in protein engineering tasks. The discussion begins with language models, ideal for protein sequence data due to their ability to handle variable sequence lengths and track long-term dependencies while maintaining token order. It covers recurrent neural networks (RNNs) [84], the self-attention mechanism [85], and transformers [86]. The chapter then explores generative models: variational autoencoders (VAEs) [87], generative adversarial networks (GANs) [88], and diffusion models [89]. VAEs use a probabilistic approach to learn data distribution, GANs involve competing neural networks to generate realistic samples, and diffusion models add noise to data and reverse the process to generate new samples. Additionally, current challenges and future directions in applying these models to protein engineering are discussed, providing practical insights for researchers.

In Chapter Six [74], a data-centric approach is adopted by integrating evolutionary data,

particularly through ancestral sequence reconstruction (ASR) [90], to enhance the generative and language models discussed in Chapter Five. ASR provides high-quality training datasets from evolutionary trajectories and diverse functional adaptations, improving the robustness and generalizability of ML models. Evolutionary insights are valuable because ancient proteins often exhibit desirable traits: stability, adaptability, and pharmacologically relevant properties [90, 91]. By utilizing statistical models and phylogenetic tree reconstruction, extensive, high-quality datasets are created for training generative models. These datasets, generated with the AP-LASR [91] tool, include homogeneous and diverse sets to enhance sequence variability and model robustness. Instead of relying solely on the most probable sequences, sampling from a posterior probability distribution captures a wide spectrum of evolutionary insights [92]. Both generative models and language models are employed to utilize evolutionary data: the former to generate novel hyperstable and diverse protein sequences, and the latter to create competitive and sometimes superior protein representations for downstream stability prediction tasks, informed by evolutionary information. VAE is employed for generating novel protein sequences, transforming them into a computationally suitable format using one-hot encoding, and processing them through a 1D convolutional neural network (CNN) [93] layer to capture local sequence patterns. The VAE's architecture, with a latent space dimensionality of 100, captures the nuances of protein sequence variability. The generated sequences are evaluated using AlphaFold2 for 3D structure prediction and FoldX [94] for stability calculations, ensuring structural integrity and thermal stability. Additionally, a new protein representation aware of their evolutionary path is introduced using the ESM2 protein language model. This representation is extracted to refine protein family classification tasks with KNN [95], Random Forest [96], and XGBoost algorithms [97], resulting in competitive and sometimes better performance in stability prediction tasks. This comprehensive framework demonstrates that addressing specific biological problems requires a nuanced understanding of both the biological context and the strategic application of appropriate ML tools, leading to more efficient and effective protein design.

BIBLIOGRAPHY

[1] Hamid Ramezani and Hendrik Dietz. Building machines with dna molecules. *Nature Reviews Genetics*, 21(1):5–26, 2020.

[2] Paul C Whitford and José N Onuchic. What protein folding teaches us about biological function and molecular machines. *Current Opinion in Structural Biology*, 30:57–62, 2015.

[3] David C Miller, Soumitra V Athavale, and Frances H Arnold. Combining chemistry and protein engineering for new-to-nature biocatalysis. *Nature Synthesis*, 1(1):18–23, 2022.

[4] Jie Zhu, Nicole Avakyan, Albert Kakkis, Alexander M Hoffnagle, Kenneth Han, Yiying Li, Zhiyin Zhang, Tae Su Choi, Youjeong Na, Chung-Jui Yu, et al. Protein assembly by design. *Chemical reviews*, 121(22):13701–13796, 2021.

[5] Tristan Bepler and Bonnie Berger. Learning the protein language: Evolution, structure, and function. *Cell systems*, 12(6):654–669, 2021.

[6] Isabell Bludau and Ruedi Aebersold. Proteomic and interactomic insights into the molecular basis of cell functional diversity. *Nature Reviews Molecular Cell Biology*, 21(6):327–340, 2020.

[7] Elizabeth L Lieu, Tu Nguyen, Shawn Rhyne, and Jiyeon Kim. Amino acids in cancer. *Experimental & molecular medicine*, 52(1):15–30, 2020.

[8] Ehsan Kianfar. Protein nanoparticles in drug delivery: animal protein, plant proteins and protein cages, albumin nanoparticles. *Journal of Nanobiotechnology*, 19(1):159, 2021.

[9] Mattia D Pizzagalli, Ariel Bensimon, and Giulio Superti-Furga. A guide to plasma membrane solute carrier proteins. *The FEBS journal*, 288(9):2784–2835, 2021.

[10] Bruno S. Silvestre and Diana Mihaela Ţîrcă. Innovations for sustainable development: Moving toward a sustainable future. *Journal of Cleaner Production*, 208:325–332, 2019.

[11] Ali Miserez, Jing Yu, and Pezhman Mohammadi. Protein-based biological materials: Molecular design and artificial production. *Chemical Reviews*, 123(5):2049–2111, 2023.

[12] Chenyi Li, Ruihua Zhang, Jian Wang, Lauren Marie Wilson, and Yajun Yan. Protein engineering for improving and diversifying natural product biosynthesis. *Trends in biotechnology*, 38(7):729–744, 2020.

[13] Sasha B Ebrahimi and Devleena Samanta. Engineering protein-based therapeutics through structural and chemical design. *Nature Communications*, 14(1):2411, 2023.

[14] Bernard R Glick and Cheryl L Patten. *Molecular biotechnology: principles and applications*

of recombinant DNA. John Wiley & Sons, 2022.

[15] Fuguo Jiang and Jennifer A Doudna. Crispr–cas9 structures and mechanisms. *Annual review of biophysics*, 46:505–529, 2017.

[16] Christopher B Black, Thomas D Duensing, Linda S Trinkle, and R Terry Dunlay. Cell-based screening using high-throughput flow cytometry. *Assay and drug development technologies*, 9(1):13–20, 2011.

[17] Linghui Qian, Xuefen Lin, Xue Gao, Rizwan Ullah Khan, Jia-Yu Liao, Shubo Du, Jingyan Ge, Su Zeng, and Shao Q Yao. The dawn of a new era: targeting the "undruggables" with antibody-based therapeutics. *Chemical reviews*, 123(12):7782–7853, 2023.

[18] Sasha B Ebrahimi and Devleena Samanta. Engineering protein-based therapeutics through structural and chemical design. *Nature Communications*, 14(1):2411, 2023.

[19] Rohitash Yadav, Jitendra Kumar Chaudhary, Neeraj Jain, Pankaj Kumar Chaudhary, Supriya Khanra, Puneet Dhamija, Ambika Sharma, Ashish Kumar, and Shailendra Handu. Role of structural and non-structural proteins and therapeutic targets of sars-cov-2 for covid-19. *Cells*, 10(4):821, 2021.

[20] Timothy M Caradonna and Aaron G Schmidt. Protein engineering strategies for rational immunogen design. *npj Vaccines*, 6(1):154, 2021.

[21] Naveen K Mehta, Roma V Pradhan, Ava P Soleimany, Kelly D Moynihan, Adrienne M Rothschilds, Noor Momin, Kavya Rakhra, Jordi Mata-Fink, Sangeeta N Bhatia, K Dane Wittrup, et al. Pharmacokinetic tuning of protein–antigen fusions enhances the immunogenicity of t-cell vaccines. *Nature biomedical engineering*, 4(6):636–648, 2020.

[22] Chloe Hsu, Hunter M. Nisonoff, Clara Fannjiang, and Jennifer Listgarten. Learning protein fitness models from evolutionary and assay-labeled data. *Nature Biotechnology*, 2022.

[23] Mehrsa Mardikoraem and Daniel R. Woldring. Protein fitness prediction is impacted by the interplay of language models, ensemble learning, and sampling methods. *Pharmaceutics*, 2023.

[24] Mehrsa Mardikoraem and Daniel Woldring. Protein fitness prediction is impacted by the interplay of language models, ensemble learning, and sampling methods. *Pharmaceutics*, 15(5), 2023.

[25] Jacques Monod, Jeffries Wyman, and Jean-Pierre Changeux. On the nature of allosteric transitions: A plausible model. *Journal of Molecular Biology*, 12(1):88–118, 1965.

[26] Lauren E Powell and Paul A Foster. Protein disulphide isomerase inhibition as a potential cancer therapeutic strategy. *Cancer medicine*, 10(8):2812–2825, 2021.

[27] Alexander S. Hauser, Maggie M. Attwood, Mathias Rask-Andersen, Helgi B. Schiöth, and David E. Gloriam. Pharmacogenomics of gpcr drug targets. *Cell*, 172(1–2):41–54, 2017.

[28] Julia Koehler Leman, Pawel Szczerbiak, P Douglas Renfrew, Vladimir Gligorijevic, Daniel Berenberg, Tommi Vatanen, Bryn C Taylor, Chris Chandler, Stefan Janssen, Andras Pataki, et al. Sequence-structure-function relationships in the microbial protein universe. *Nature communications*, 14(1):2351, 2023.

[29] Noelia Ferruz, Michael Heinzinger, Mehmet Akdel, Alexander Goncearenco, Luca Naef, and Christian Dallago. From sequence to function through structure: Deep learning for protein design. *Computational and Structural Biotechnology Journal*, 21:238–250, 2023.

[30] José Nelson Onuchic, Zaida Luthey-Schulten, and Peter G Wolynes. Theory of protein folding: the energy landscape perspective. *Annual review of physical chemistry*, 48(1):545–600, 1997.

[31] Frances H Arnold. Design by directed evolution. *Accounts of chemical research*, 31(3):125–131, 1998.

[32] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Zıdek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A.A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with alphafold. *Nature*, 596:583–589, 2021.

[33] Po-Ssu Huang, Scott E Boyken, and David Baker. The coming of age of de novo protein design. *Nature*, 537(7620):320–327, 2016.

[34] Jason Yang, Julie Ducharme, Kadina E Johnston, Francesca-Zhoufan Li, Yisong Yue, and Frances H Arnold. Decoil: Optimization of degenerate codon libraries for machine learning-assisted protein engineering. *ACS Synthetic Biology*, 12(8):2444–2454, 2023.

[35] Ariel Goldenzweig and Sarel J. Fleishman. Automated structure- and sequence-based design of proteins for high bacterial expression and stability. *Methods in Enzymology*, 580:281–315, 2016.

[36] Xuefei Wang, Duan Ni, Yaqin Liu, and Shaoyong Lu. Rational design of peptide-based inhibitors disrupting protein-protein interactions. *Frontiers in chemistry*, 9:682675, 2021.

[37] Vincent Frappier and Rafael J. Najmanovich. Rational design of proteins and enzymes for biocatalysis: Recent advances and future perspectives. *Biotechnology Advances*, 36(8):2230–2241, 2018.

[38] Jonathan Weinstein, Olga Khersonsky, and Sarel J Fleishman. Practically useful protein-design methods combining phylogenetic and atomistic calculations. *Current Opinion in Structural Biology*, 63:58–64, 2020. Membranes, Engineering and Design.

[39] Frances H Arnold. Directed evolution, bringing new chemistry to life. *Angewandte Chemie International Edition*, 57(16):4143–4148, 2018.

[40] Andrea Ilari and Carmelinda Savino. *Protein Structure Determination by X-Ray Crystallography*, pages 63–87. Humana Press, Totowa, NJ, 2008.

[41] Kurt Wüthrich. Protein structure determination in solution by nmr spectroscopy. *Journal of Biological Chemistry*, 265(36):22059–22062, 1990.

[42] Ka Man Yip, Niels Fischer, Elham Paknia, Ashwin Chari, and Holger Stark. Atomic-resolution protein structure determination by cryo-em. *Nature*, 587(7832):157–161, 2020.

[43] Li Tang and Anne Besson. Rational design of chemically controlled antibodies and protein therapeutics. *ACS Chemical Biology*, 2023.

[44] Aisaraphon Phintha and Pimchai Chaiyen. Rational and mechanistic approaches for improving biocatalyst performance. *Chem catalysis*, 2022.

[45] Shima Ghaedizadeh, Majid Zeinali, Bahareh Dabirmanesh, Behnam Rasekh, Ali Mohammad Banaei-Moghaddam, et al. Rational design engineering of a thermostable α-carbonic anhydrase from sulfurihydrogenibium yellowstonense to improve its thermostability for industrial applications. 2022.

[46] Mikaela Friedman, Anna Orlova, Eva Johansson, Tove L.J. Eriksson, Ingmarie Höidén-Guthenberg, Vladimir Tolmachev, Fredrik Y. Nilsson, and Stefan Ståhl. Directed evolution to low nanomolar affinity of a tumor-targeting epidermal growth factor receptor-binding affibody molecule. *Journal of Molecular Biology*, 376(5):1388–1402, 2008.

[47] S. D. Levin, L. S. Evans, S. Bort, E. Rickel, K. E. Lewis, R. P. Wu, J. Hoover, S. MacNeil, D. La, M. F. Wolfson, M. W. Rixon, S. R. Dillon, M. G. Kornacker, R. Swanson, and S. L. Peng. Novel immunomodulatory proteins generated via directed evolution of variant igsf domains. *Frontiers in Immunology*, 10:3086, 2020.

[48] Ron Amon, Ronit Rosenfeld, Shahar Perlmutter, Oliver C. Grant, Sharon Yehuda, Aliza Borenstein-Katz, Ron Alcalay, Tal Marshanski, Hai Yu, Ron Diskin, Robert J. Woods, Xi Chen, and Vered Padler-Karavani. Directed evolution of therapeutic antibodies targeting glycosylation in cancer. *Cancers*, 12(10), 2020.

[49] Keith A Powell, Sandra W Ramer, Stephen B Del Cardayré, Willem PC Stemmer, Matthew B Tobin, Pascal F Longchamp, and Gjalt W Huisman. Directed evolution and biocatalysis. *Angewandte Chemie International Edition*, 40(21):3948–3959, 2001.

[50] Valentina Pirillo, Marco Orlando, Caren Battaglia, Loredano Pollegioni, and Gianluca Molla. Efficient polyethylene terephthalate degradation at moderate temperature: a protein engineering study of lc-cutinase highlights the key role of residue 243. *The FEBS journal*, 290(12):3185–3202, 2023.

[51] Chang Feng. Potential of deep representative learning features to interpret the sequence information in proteomics. *Proteomics*, 2021.

[52] O. Simeone. A very brief introduction to machine learning with applications to communication systems. *IEEE Transactions on Cognitive Communications and Networking*, 4:648–664, 2018.

[53] Kevin K Yang, Zachary Wu, and Frances H Arnold. Machine-learning-guided directed evolution for protein engineering. *Nature methods*, 16(8):687–694, 2019.

[54] Ameya Harmalkar, Roshan Rao, Yuxuan Richard Xie, Jonas Honer, Wibke Deisting, Jonas Anlahr, Anja Hoenig, Julia Czwikla, Eva Sienz-Widmann, Doris Rau, et al. Toward generalizable prediction of antibody thermostability using machine learning on sequence and structure features. In *MAbs*, volume 15, page 2163584. Taylor & Francis, 2023.

[55] T. B. Alakus and I. Turkoglu. Prediction of protein-protein interactions with lstm deep learning model. In *3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT 2019) - Proceedings*, pages 1–5, 2019.

[56] A. Ahmed, B. Mam, and R. Sowdhamini. Deelig: A deep learning-based approach to predict protein-ligand binding affinity. 2020.

[57] L. Heo and M. Feig. High-accuracy protein structures by combining machine-learning with physics-based refinement. *Proteins: Structure, Function, and Bioinformatics*, 88:637–642, 2020.

[58] S. Khurana, R. Rawi, K. Kunji, et al. Deepsol: A deep learning framework for sequence-based protein solubility prediction. *Bioinformatics*, 34:2605–2613, 2018.

[59] M. Giollo, A. J. M. Martin, I. Walsh, et al. Neemo: A method using residue interaction networks to improve prediction of protein stability upon mutation. *BMC Genomics*, 15:1–11, 2014.

[60] Lawrence Fong. Learning meaningful representations of protein sequences. *Nature Communications*, 2022.

[61] Prakash M Nadkarni, Lucila Ohno-Machado, and Wendy W Chapman. Natural language processing: an introduction. *Journal of the American Medical Informatics Association*, 18(5):544–551, 2011.

[62] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin,

Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

[63] Ethan C Alley, Grigory Khimulya, Surojit Biswas, Mohammed AlQuraishi, and George M Church. Unified rational protein engineering with sequence-based deep representation learning. *Nature methods*, 16(12):1315–1322, 2019.

[64] Ali Madani, Bryan McCann, Nikhil Naik, Nitish Shirish Keskar, Namrata Anand, Raphael R Eguchi, Po-Ssu Huang, and Richard Socher. Progen: Language modeling for protein generation. *arXiv preprint arXiv:2004.03497*, 2020.

[65] Emre Sevgen, Joshua Moller, Adrian Lange, John Parker, Sean Quigley, Jeff Mayer, Poonam Srivastava, Sitaram Gayatri, David Hosfield, Maria Korshunova, et al. Prot-vae: protein transformer variational autoencoder for functional protein design. *bioRxiv*, pages 2023–01, 2023.

[66] Noelia Ferruz, Steffen Schmidt, and Birte Höcker. Protgpt2 is a deep unsupervised language model for protein design. *Nature communications*, 13(1):4348, 2022.

[67] Donatas Repecka, Vykintas Jauniskis, Laurynas Karpus, Elzbieta Rembeza, Irmantas Rokaitis, Jan Zrimec, Simona Poviloniene, Audrius Laurynenas, Sandra Viknander, Wissam Abuajwa, et al. Expanding functional protein sequence spaces using generative adversarial networks. *Nature Machine Intelligence*, 3(4):324–333, 2021.

[68] Mehrsa Mardikoraem, Zirui Wang, Nathaniel Pascual, and Daniel Woldring. Generative models for protein sequence modeling: recent advances and future directions. *Briefings in Bioinformatics*, 24(6):bbad358, 2023.

[69] Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Peter Chen, John Canny, Pieter Abbeel, and Yun Song. Evaluating protein transfer learning with tape. *Advances in neural information processing systems*, 32, 2019.

[70] Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, et al. Prottrans: Toward understanding the language of life through self-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(10):7112–7127, 2021.

[71] Amir Shanehsazzadeh, David Belanger, and David Dohan. Is transfer learning necessary for protein landscape prediction? *arXiv preprint arXiv:2011.03443*, 2020.

[72] Pascal Notin, Aaron W Kollasch, Daniel Ritter, Lood Van Niekerk, Steffan Paul, Han Spinner, Nathan J Rollins, Ada Shaw, Rose Orenbuch, Ruben Weitzman, Jonathan Frazer, Mafalda Dias, Dinko Franceschi, Yarin Gal, and Debora Susan Marks. Proteingym: Large-scale benchmarks for protein fitness prediction and design. In *Thirty-seventh Conference on Neural*

Information Processing Systems Datasets and Benchmarks Track, 2023.

[73] Christian Dallago, Jody Mou, Kadina E Johnston, Bruce Wittmann, Nick Bhattacharya, Samuel Goldman, Ali Madani, and Kevin K Yang. FLIP: Benchmark tasks in fitness landscape inference for proteins. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[74] Mehrsa Mardikoraem and Daniel R. Woldring. PROTEIN FITNESS LANDSCAPE NAVIGATION IS BOOSTED VIA INCORPORATING EVOLUTIONARY INFORMATION INTO MACHINE LEARNING MODELS. In *ICLR 2024 Workshop on Generative and Experimental Perspectives for Biomolecular Design*, 2024.

[75] Emily E Wrenbeck, Matthew S Faber, and Timothy A Whitehead. Deep sequencing methods for protein engineering and design. *Current opinion in structural biology*, 45:36–44, 2017.

[76] Mehrsa Mardikoraem and Daniel Woldring. *Machine Learning-driven Protein Library Design: A Path Toward Smarter Libraries*, pages 87–104. Springer US, New York, NY, 2022.

[77] Jiajie Zhang, Kassian Kobert, Tomáš Flouri, and Alexandros Stamatakis. Pear: a fast and accurate illumina paired-end read merger. *Bioinformatics*, 30(5):614–620, 2014.

[78] H Li. Github-lh3/seqtk: toolkit for processing sequences in fasta/q formats, 2021.

[79] Barry T Bosworth, W Robert Bernecky, James D Nickila, Berhane Adal, and G Clifford Carter. Estimating signal-to-noise ratio (snr). *IEEE Journal of Oceanic Engineering*, 33(4):414–418, 2008.

[80] B Kovács, F Tinya, C Németh, and P Ódor. Smote: synthetic minority over-sampling technique nitesh. *Ecol. Appl*, 30(2):321–357, 2020.

[81] Ralph L Keeney, Howard Raiffa, and David W Rajala. Decisions with multiple objectives: Preferences and value trade-offs. *IEEE transactions on Systems, man, and cybernetics*, 9(7):403–403, 1979.

[82] Jörg König, Annick Seithel, Ulrike Gradhand, and Martin F Fromm. Pharmacogenomics of human oatp transporters. *Naunyn-Schmiedeberg's archives of pharmacology*, 372:432–443, 2006.

[83] Klaas Nico Faber, Michael Müller, and Peter LM Jansen. Drug transport proteins in the liver. *Advanced drug delivery reviews*, 55(1):107–124, 2003.

[84] Robin M. Schmidt. Recurrent neural networks (rnns): A gentle introduction and overview. *CoRR*, abs/1912.05911, 2019.

[85] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. 2017.

[86] Richard E. Turner. An introduction to transformers, 2024.

[87] Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

[88] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[89] Hanqun Cao, Cheng Tan, Zhangyang Gao, Yilun Xu, Guangyong Chen, Pheng-Ann Heng, and Stan Z Li. A survey on generative diffusion models. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[90] Yvonne Joho, Vanessa Vongsouthi, Matthew A. Spence, Jennifer Ton, Chloe Gomez, Li Lynn Tan, Joe A. Kaczmarski, Alessandro T. Caputo, Santana Royan, Colin J. Jackson, and Albert Ardevol. Ancestral sequence reconstruction identifies structural changes underlying the evolution of ideonella sakaiensis petase and variants with improved stability and activity. *Biochemistry*, 62(2):437–450, 2023. PMID: 35951410.

[91] James VanAntwerp, Mehrsa Mardikoraem, Nathaniel Pascual, and Daniel Woldring. Ap-lasr: Automated protein libraries from ancestral sequence reconstruction. *bioRxiv*, 2023.

[92] Geeta N. Eick, Jamie T. Bridgham, Douglas P. Anderson, Michael J. Harms, and Joseph W. Thornton. Robustness of Reconstructed Ancestral Protein Functions to Statistical Uncertainty. *Molecular Biology and Evolution*, 34(2):247–261, 10 2016.

[93] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *CoRR*, abs/1511.08458, 2015.

[94] Joost Schymkowitz, Jesper Borg, Francois Stricher, Robby Nys, Frederic Rousseau, and Luis Serrano. The FoldX web server: an online force field. *Nucleic Acids Research*, 33(suppl2):W382–W388, 07 2005.

[95] Gongde Guo, Hui Wang, David Bell, Yaxin Bi, and Kieran Greer. Knn model-based approach in classification. pages 986–996, 2003.

[96] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[97] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

CHAPTER 2

MACHINE LEARNING-DRIVEN PROTEIN LIBRARY DESIGN: A PATH TOWARD SMARTER LIBRARIES

[1]

Abstract

Proteins are small yet valuable biomolecules that play a versatile role in therapeutics and diagnostics. The intricate sequence-structure-function paradigm in the realm of proteins opens the possibility for directly mapping amino acid sequences to function. However, the rugged nature of the protein fitness landscape and an astronomical number of possible mutations even for small proteins make navigating this system a daunting task. Moreover, the scarcity of functional proteins and the ease with which deleterious mutations are introduced, due to complex epistatic relationships, compound the existing challenges. This highlights the need for auxiliary tools in current techniques such as rational design and directed evolution. To that end, state-of-the-art machine learning can offer time and cost efficiency in finding high-fitness proteins, circumventing unnecessary wet-lab experiments. In the context of improving library design, machine learning provides valuable insights via its unique features such as high adaptation to complex systems, multi-tasking and parallelism, and the ability to capture hidden trends in input data. Finally, both the advancements in computational resources and the rapidly increasing number of sequences in protein databases will allow more promising and detailed insights delivered from machine learning to protein library design. In this chapter, fundamental concepts and a method for machine learning-driven library design leveraging deep sequencing datasets will be discussed. We elaborate on i) basic knowledge about machine learning algorithms, ii) the benefit of machine learning in library design, and iii) methodology for implementing machine learning in library design.

Keywords: Library Design, Directed Evolution, Machine Learning, Deep Learning

[1]This chapter is adapted from content published in "Machine Learning-driven Protein Library Design: A Path Toward Smarter Libraries," SpringerLink. All rights reserved. For more information, visit https://link.springer.com/protocol/10.1007/978-1-0716-2285-8_5.

2.1 Introduction

Proteins are molecules with a wide variety of applications in biological processes. They have many fundamental functions in living organisms such as being catalysts, receptors, structural elements, transporters, and regulators [1, 2, 3, 4, 5]. Accordingly, increased potential for mapping the protein sequence to its function will result in comprehension and regulation of biological processes associated with functional disorders. As a result, techniques to efficiently navigate the protein fitness landscape are at the forefront of protein engineering. Nevertheless, the complexity and ruggedness of the protein fitness landscape, and the high potential of failure in searching through the fitness landscape (mutations resulting in unstable and non-functional variants), make this task more demanding. A growing number of computational and experimental approaches (e.g. high-resolution stability calculations [reference to stability calculation in the book chapter], virtual screening [6], deep sequencing [7], cytometry-based selections [8], ancestral sequence reconstruction [ref to ASR in book chapter]) seek to address these gaps in knowledge.

Machine learning algorithms offer a platform for harnessing large, diverse datasets to understand natural protein features and guide protein engineering efforts. This provides the opportunity to map protein sequences to function without requiring explicit biophysical knowledge of individual sequences. Another advantage of such algorithms lies in their ability to expand the utility of experimentally derived sequences beyond the typically small subsets of lead variants. While directed evolution discards low-fitness protein variants, machine learning can leverage these sub-optimal variants to enhance the model's predictive performance [9]. Machine learning proves particularly advantageous for protein engineering campaigns characterized by low-throughput or labor-intensive selection processes. Therefore, its usefulness depends on factors such as library size, screening difficulty, fitness landscape ruggedness, and the accuracy of the predictive model. As a result, the ability to capture complicated trends among protein datasets, aided by nonlinear functionality to reveal important features, makes machine learning a powerful tool for guiding protein library design.

Machine learning has played an important role in protein engineering for more than 30 years,

yielding improved prediction of tertiary structure [10] and protein-protein interactions [11] using amino acid sequence information alone [10, 11]. Machine learning algorithms such as support vector machines (SVMs), random forest (RF), and gradient boosting machines (GBMs) have provided platforms for protein function and property prediction including stability, catalytic activity, and secondary structure [12, 13, 14, 15]. Deep learning (DL) [16], as a sub-field of machine learning, imitates human brain functionality in decision-making and learning experiences. Utilizing non-linear functions, the algorithm can learn and extract desired features from the provided input data, well suited for dealing with rich datasets with high dimensionality. This makes deep learning methods particularly promising for evaluating sequence–function trends among a rapidly growing number of protein sequences. Although practical and promising, developing a fine-tuned strategy to employ machine learning in the field demands an awareness of the existing challenges and capabilities. Here, we present a procedure for establishing deep-learning models that guide protein library design. Common challenges and best practices in this burgeoning field will be highlighted.

We consider multiple practical applications of machine learning within the context of protein library design:

- Combinatorial library design based on deep sequencing data following high-throughput directed evolution
- Process parallelization and parameterization of features within far-reaching parts of the fitness landscape
- The ability to sample the diversity applied by specific degenerate codon techniques and oligonucleotide combinations before experimental implementation.

2.1.1 Providing a better starting point for directed evolution

Directed evolution campaigns are initialized using a parent sequence to implement mutations. Therefore, the path-dependency of directed evolution benefits from a high-fitness or highly stable sequence as a starting point to increase the probability of finding optimal regions within a fitness landscape [17]. Machine learning can guide directed evolution by identifying a large collection of promising sequences based on curated input data. In a recent example, a machine learning model

was trained based on the initial library of fluorescent proteins to build a second small yet enriched protein library [18].

2.1.2 Investigating unexplored parts of the fitness landscape

Machine learning algorithms provide some unique advantages that enable finding unexplored variants that may possess high performance. Large datasets with high-complexity algorithms are not only able to predict functionality but can learn hidden characteristics and rules that exist in provided data. As an example, UniRep [19, 20] has been trained on protein sequences in Uniref50 [21] on 24 million amino acid sequences to obtain general trends in protein sequences and statistical representations of amino acids. Another factor that is effective in finding the unexplored high-fitness sequences is the application of parallelism and multi-tasking within machine learning. Multi-task learning is a subset of machine learning algorithms that trains multiple tasks simultaneously in one unique model [22]. This feature enables capturing the epistatic relationships in protein sequences by providing a path-independent search in the fitness landscape [23]. In this way, applying machine learning to protein engineering enables an increased likelihood of accessing undetected regions of the fitness landscape.

2.1.3 Estimating degenerate codon performance via fitness distribution analysis

Implementing a well-trained machine learning algorithm enables the evaluation of multiple design strategies and reduces experimental effort. Various experimental techniques have been developed to improve the efficiency of directed evolution such as gene shuffling [24] and neutral drift library screening [25] to manage the library size and increase the likelihood of finding the desired property. Another highly used method is to generate libraries based on degenerate codons to introduce tailored diversity at individual positions. As an example, while NNK is used to generate a library coding for all amino acids (with 3% stop codons), other combinations such as NYC (coding for hydrophobic residues), KST (coding for small residues), and NDT (coding for a balanced set of all amino acid properties) are available. In addition, several impressive computational attempts have been used to even go beyond these techniques and optimize the oligonucleotide combinations [26, 27]. Among these potential degenerate codon techniques for the library design, the user can

pre-analyze their performance for their desired protein.

Figure 2.1 proposes a comparison of candidate degenerate codons. As a base platform, an initial deep-learning model is trained on the objective protein. Subsequently, the algorithm can generate sequences that incorporate candidate degenerate codons and assign a predicted fitness score to each based on the previously trained model. Finally, the distribution of fitness scores is calculated for all variants of each degenerate codon strategy. Statistical tools such as Jensen-Shannon-Divergence [28] and survival function [29] provide a direct comparison between individual distributions in terms of diversity and fitness (see Note 1). It should be emphasized that the criteria for choosing the best performance depends on the particular application the platform is used for.

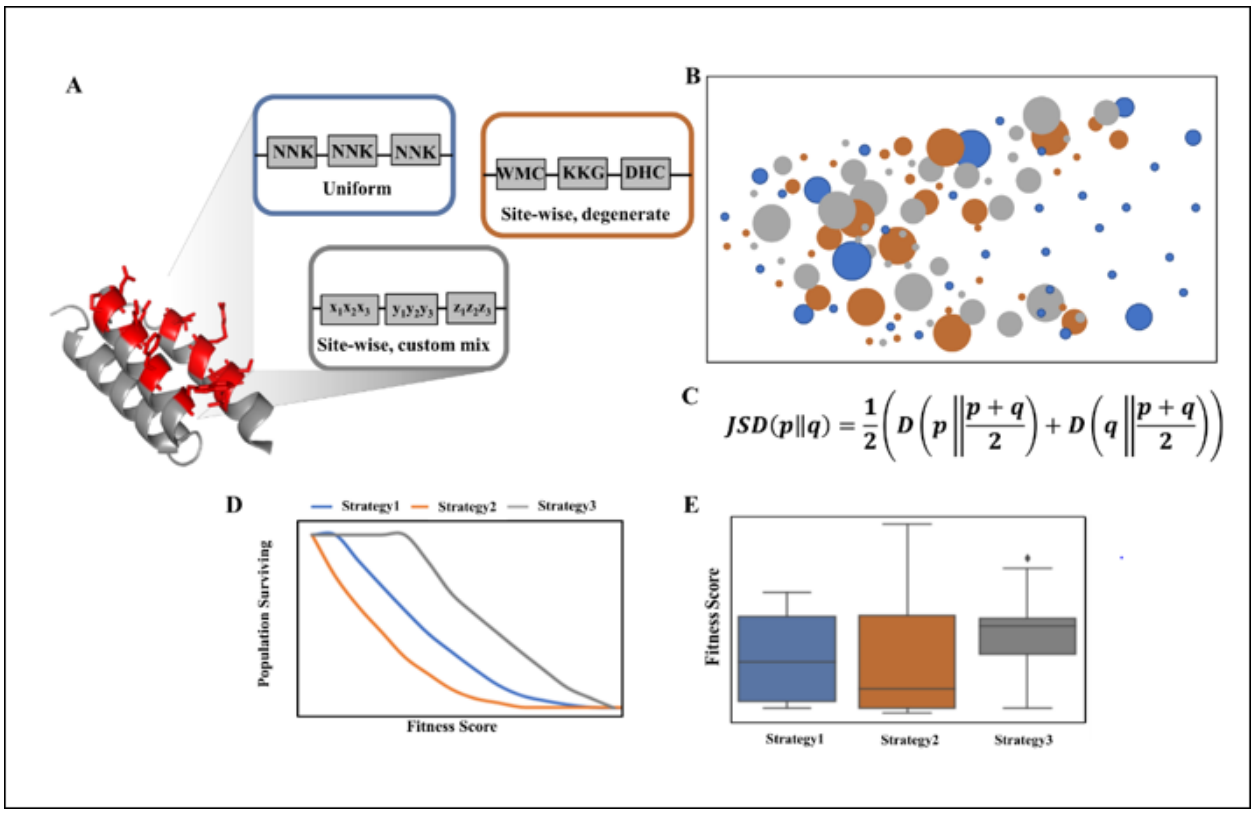


Figure 2.1 Evaluation of degenerate codon performance within multiple design strategies using the trained model based on experimental data on the protein of interest. (A) Elaboration of different hypothetical degenerate codon strategies in three different sites of the protein (the previously trained model needs to be trained on high-quality experimental data to predict the fitness of generated sequences in each technique.) The first utilizes the NNK technique in all sites while the second uses different degenerate codons at each site. The third strategy uses custom mix codons (versatile ratios of base pairs). Sequences compatible with the rubrics of each strategy can be generated and a fitness score for the generated sequences will be assigned to each sequence based on the previously trained model in the protein of interest. (B) The score function (transformed and standardized predicted scores) distribution will be obtained for each candidate library. One method to comprehend the predicted data is by clustering the generated data. The exemplary plot shown here is based on both the sequences and scores of the three strategies when each dot represents one sequence produced by one of the strategies. The radius of the dots represents their fitness scores, and the distance between any two dots represents how distant those amino acid sequences are from each other in sequence space. (C) The Jenson-Shannon-Divergence is useful for quantifying differences that exist between individual distributions and a reference distribution or the extent of similarity between candidates' distributions. (D) The survival function provides the probability of improved score functions compared to the current score function, providing more insight for the analysis of distributions. (E) Box plots can be produced based on the generated fitness scores and are informative for recognizing the quantiles and making comparisons between them in the three different strategies.

2.2 Materials

There are a plethora of software libraries and packages provided for implementing model optimization, machine learning, and deep learning algorithms. Choosing one depends on the specific application that the user has in mind. TensorFlow [30], Keras [31], PyTorch [32], and Scikit-learn [33] are among the most popular libraries with each having some trade-offs (See Note 2 for analysis). The discussion found here focuses on deep learning in Keras. Please refer to our GitHub (https://github.com/WoldringLabMSU/DeepLearning.git) for a collection of relevant Python scripts to guide model implementation as well.

- The latest version of Python[2] installed (Installing Anaconda[3] is highly suggested for beginners as it is straightforward to use).
- Spyder or Jupyter Notebook environments in Anaconda are both popular and practical in doing machine learning and deep learning projects.
- `pandas` [34] (For dealing with data frames)
- `numpy` [35] (For efficient mathematical operations)
- `scikit_learn` [36] (This is mostly used in machine learning but its preprocessing section has a myriad of useful functions for analysis and fine-tuning the data)
- `tensorflow` [37] (For using Keras, its backend should be installed (CNTK and Theano are other options, as well.)
- `keras` [31]

To use these libraries, use `pip install X` in the command prompt.

2.3 Methods

Below is a general workflow representing the required actions in building a deep learning algorithm from deep sequencing data. Here, the goal is to produce a supervised learning algorithm for predicting protein function based on amino acid sequence.

[2]https://www.python.org
[3]https://www.anaconda.com

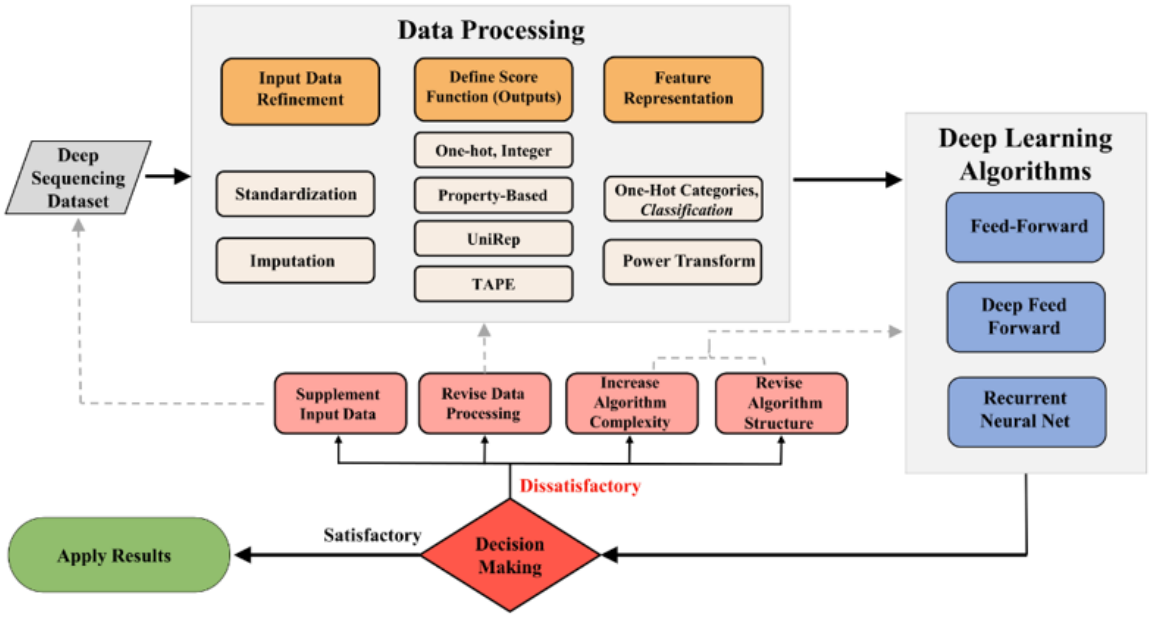


Figure 2.2 This figure provides the overall workflow of what the user may encounter when using deep sequencing data and wants to apply a machine learning algorithm. The main steps for using this path are data processing, choosing an appropriate learning algorithm, and deciding based on the algorithms' performance. A detailed explanation of the steps is mentioned in the content.

2.3.1 Data processing as an initial yet pivotal step in any DL algorithm

The main purpose of this step is to prepare the data to be fed into the deep learning algorithm. Importantly, what can be learned by the model strongly depends on what is provided to the algorithm as an input. If the aim is to map the sequence to function, the protein sequence should be as an input labeled with the desired functionality (output). Three important steps in data processing include input data refinement, input data representation, and output data representation (if dealing with supervised learning). See Note 2.

2.3.1.1 Input data refinement

First, one should input and standardize the features. One rationale behind this refinement is that scaling the input removes the "importance" of the raw magnitude of one factor relative to another and as a result reduces estimation errors and calculation times. One package which can be used for standardization is `StandardScaler` or `RobustScaler` (advantageous when dealing with outliers)

from the sklearn preprocessing package:

```
from sklearn.preprocessing import StandardScaler
scaler1 = StandardScaler()
scaler1.fit(Feature)
Feature_standardized1 = scaler1.transform(Feature)

From sklearn.preprocessing import RobustScaler
scaler2 = RobustScaler()
scaler2.fit(Feature)
Feature_standardized2 = scaler2.transform(Feature)
```

2.3.1.2 Input data representation

One-hot encoding (Note 3), integer encoding (Note 4), physiochemical property-based encoding (Note 5), UniRep encoding[4] [19], TAPE[5] [38] are notable options for representing amino acid sequence data. An example code for one-hot encoding the input data representation is mentioned in the following. The user can manually one-hot encode the sequences via defining dictionaries or using particular packages in Python (Refer to https://github.com/WoldringLabMSU/DeepLearning.git for more information on encoding).

```
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
Amino_Acids = ["A", "C", "D", "E", "F", "G", "H", "I", "K", "L", "M",
"N", "P", "Q", "R", "S", "T", "V", "W", "Y"]
label_encoder = LabelEncoder()
onehot_encoder = OneHotEncoder(sparse = False)
integer_encode = label_encoder.fit_transform(Amino_Acids)
```

[4]https://github.com/churchlab/UniRep
[5]https://github.com/songlab-cal/tape

```
integer_encoded = integer_encode.reshape(len(integer_encode), 1)
Amino_Acids_onehot = onehot_encoder.fit_transform(integer_encoded)
```

2.3.1.3 Output data representation

Following high throughput selection and deep sequencing of an initial combinatorial library, amino acid sequences can be labeled based on the observed enrichment ratios as a metric for relative fitness [39]. Depending on the experimental conditions for selection and depth of sequencing, the distribution of enrichment ratio data may take on various forms and require further refinement (see Note 6).

2.3.2 Deep learning algorithm selection requires an understanding of their structure

2.3.2.1 Overview

Artificial neural network (ANN) architecture is inspired by human brain function which can learn from various input data. The building blocks for this network (neurons) receive information from adjacent neurons, and then process this information with the aid of an activation function (see Note 7) before being sent to other downstream neurons. The effect and significance of each connection are proportional to its assigned weight.

There are many deep learning methods utilized in the field from the feed-forward neural network (FNN) to the convolutional neural network (CNN) [40] and recurrent neural network (RNN) [41]. FNNs are primary neural network algorithms that are rigorous and powerful in capturing high-dimensional features. It consists of three distinct layers (each composed of neurons): the input layer, the hidden layer, and output layer. The input layer receives the data features, the hidden layer transforms the input layer to the output by updating the weights iteratively to minimize the loss function. For each of these neural network algorithms, backpropagation is used to update model weights to minimize the loss function via gradient descent. Finally, the output layer captures the results. Figure 3 illustrates one example structure of a feed-forward neural network.

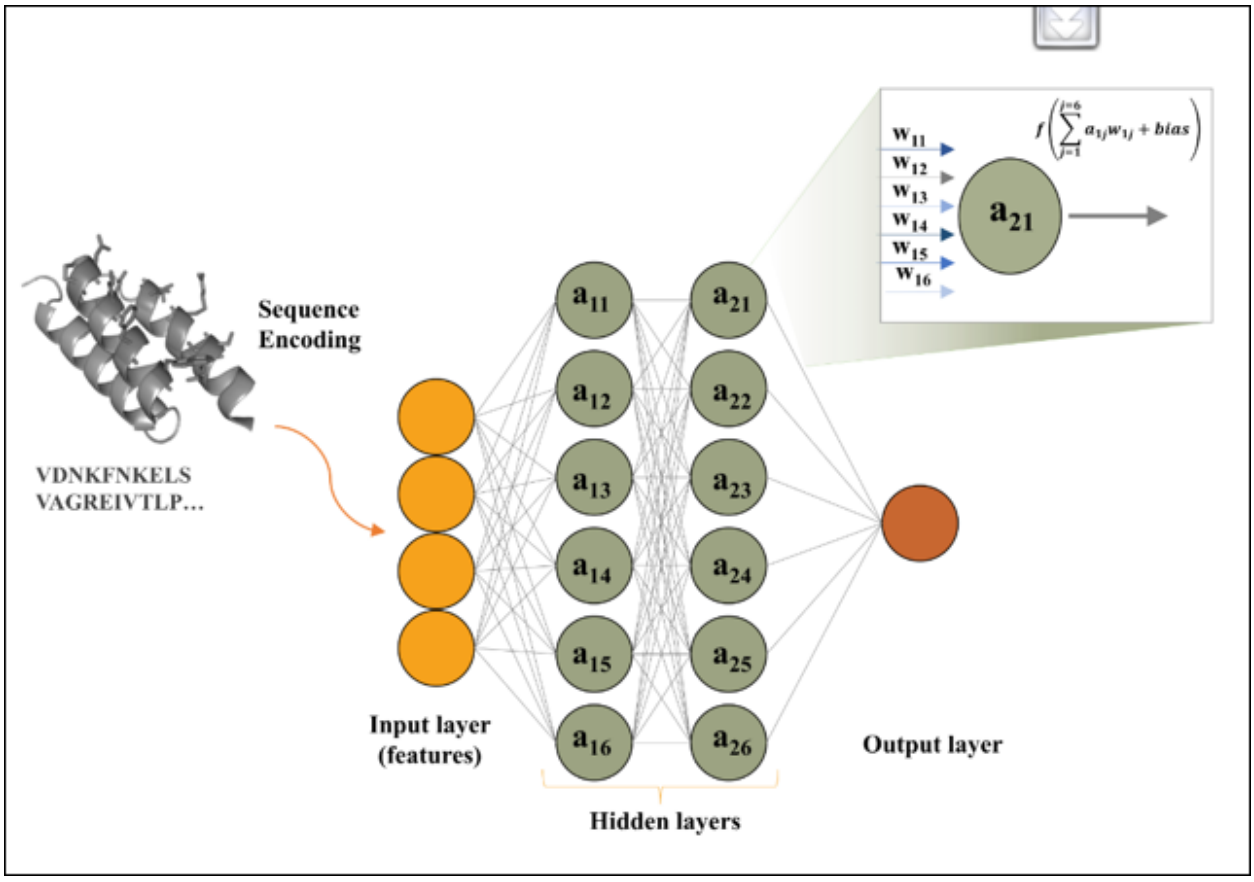


Figure 2.3 This figure represents one example of a feed-forward neural network. It consists of three main layers: the input, hidden layer(s), and output layers. It is a symbolic representation of the path from sequence to function. The magnifying glass focuses on one neuron (a21), illustrating how information passes through every neuron. Moreover, the information from six neurons in the previously hidden layer is multiplied by their weights (different colors represent the activation level in each corresponding neuron). Afterward, the result will be summed with bias and pass through the activation function to determine the output of that neuron.

Convolutional neural networks are an additional deep learning algorithm inspired by the visual cortex in animals. CNNs capture hidden relationships and spatial dependencies in provided input via various filters, pooling, and convolutional kernels. Therefore, local properties will be obtained by using sliding filters and non-linear functions [42]. In recurrent neural networks, there is a cyclic connection in the algorithm structure, whereby the current state of the algorithm will be updated based upon the past state and the current input data [43]. This characteristic makes RNN useful in time series data and capturing temporal relationships in sequences. In addition, variations of RNN such as LSTM [44] and GRU [45] have been developed to resolve the potential problems

in its architecture such as capturing long-distance relationships and resolving vanishing gradient problems. The best performance in the mentioned algorithms depends on the type of data and purpose for using deep learning. However, more advanced architectures like RNN/CNN can learn properties more efficiently than a traditional FNN.

2.3.2.2 Guidance for building a deep learning structure

Here we build a simple FNN in Keras to demonstrate a potential structure for constructing a neural network (https://github.com/WoldringLabMSU/DeepLearning.git).

```
# Importing the required libraries from Keras
from tensorflow import keras
import keras
from keras.models import Sequential
from keras.layers.core import Dense
from keras.optimizers import Adam
from keras.activations import relu, sigmoid
from sklearn.model_selection import train_test_split

# Splitting the data set into train and test
X_train, X_test, Y_train, Y_test = train_test_split(features, labels,
test_size=0.2, random_state=42)

# Defining the model
My_Model = Sequential()

# Defining the input layer, the number of neurons in that layer

My_Model.add(Dense(20, activation='relu', input_shape=(100,)))
```

```
# Defining the hidden layers , number of neurons in each hidden layer,
and corresponding activation function
My_Model.add(Dense(10, activation='relu'))
My_Model.add(Dense(5, activation='relu'))

# Defining the output layer
My_Model.add(Dense(1, activation='relu'))  # The last activation function
depends on the inherent nature of the task (see note 9)

# Compiling the model and determining the optimizer, learning rate,
loss function, and evaluation metrics
My_Model.compile(Adam(lr=0.01, decay=0.003), loss='mean_squared_error'
, metrics=['mse'])

# Fitting the defined model
My_Model.fit(X_train, Y_train, batch_size=100, epochs=40, verbose=2,
validation_split=0.2, shuffle=True)

# Evaluating the fitting performance
Metric = My_Model.evaluate(X_test, Y_test, verbose=2)

# Predicting the labels for the test data set
y_test_predicted = My_Model.predict(X_test, verbose=2)
```

2.3.3 Visualization guidance

Evaluation metrics provide useful insights into algorithm performance. Visualizations of these results help to further interpret and communicate the findings. For example, the seaborn library

allows for showing the correlation between predicted output values versus actual values. Below is a simple code implemented for this purpose.

```
import seaborn as sns
sns.joint plot(Prediciton, Actual_value, kind='scatter', s=150, color='m',
edgecolor="skyblue", linewidth=2)
```

Understanding and building such a structure is an important first step to take for utilizing deep learning in different projects. However, it should be emphasized that, even with a powerful algorithm and appropriate data processing step, the prediction might be drastically off which leads us to decision-making and further fine-tuning steps.

2.3.4 Decision Making & evaluating parameters

After processing the data and choosing the appropriate algorithm, one should be able to accurately evaluate the performance of the algorithm. Evaluation metrics depend on whether the problem is regression or classification. In classification, metrics such as accuracy, confusion matrix, AUC, and Recall are useful. While in regression, metrics such as RMSE, MSE, and MAPE are better suited (see Note 8). Obtaining poor metrics for a model may arise from various stages in the algorithm such as inadequate input data, deficient preprocessing step, overfitting, and untuned hyperparameters. Choosing the right hyperparameters and preventing the system from overfitting are two necessary tasks in training any deep learning algorithm. Two generally used methods for resolving some of these issues are elaborated in the following.

2.3.4.1 Hyperparameter Optimization [46]

Hyperparameters are a set of variables that dictate various characteristics of the algorithm's structure and influence the process used for training models. It can be considered as a meta optimization technique whereby parameter value fitness is monitored via the loss function during the training process [47]. Multiple methods can be employed for searching through hyperparameter space (e.g. manual search, grid search, random search, and Bayesian optimization). The manual search involves tuning the hyperparameters by a user based on guess and check. In grid search,

for each parameter of interest, the user defines a list of values to implement. The algorithm then calculates the loss for all possible combinations of parameter values. In random search, some but not all combinations of parameters will be selected randomly, and the best loss will be chosen based on the selected parameters. Bayesian optimization offers high efficiency by using information from the past as an experience to choose the next set of hyperparameters. HYPEROPT[6] and OPTUNA[7] are among the Python libraries designed for hyperparameter optimization based on the objective function. (Refer to https://github.com/WoldringLabMSU/DeepLearning.git for more information).

2.3.4.2 K-fold cross-validation

This resampling approach allows for more accurate prediction in model performance using even a limited data set. It splits the data into k complementary groups and uses k-1 groups for training and one group for evaluating performance. The performance of the cross-validation is calculated by taking the mean and variance over all k performances. This enables a less biased estimate of model performance [47]. (Refer to https://github.com/WoldringLabMSU/DeepLearning.git, for example, K-fold cross-validation).

2.3.5 Library Construction

Machine learning techniques enable the design of smart libraries by identifying the protein positions that are highly amenable to mutation and determining the most suitable degenerate codon candidate(s) for the protein of interest. Based on these designs, full-length genes can then be constructed using overlap extension PCR of degenerate oligos to incorporate the intended site-wise amino acid diversity. Finally, the newly constructed full-length gene library, combined with a linearized yeast surface display vector (e.g. pCT-CON2), can be electroporated into yeast (EBY100) [48] and evaluated by high-throughput techniques [49, 50, 51, 52, 53].

[6] http://hyperopt.github.io/hyperopt/
[7] https://optuna.org/

Notes

2.3.6 Learning paradigms in ANN

The learning paradigms in ANN are supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. In supervised learning, the data are provided with assigned labels that then guide the algorithm to learn the input-output mapping through a backpropagation process. Unsupervised learning algorithms (e.g. K-nearest neighbors and K-means) are used when there are no values assigned to the input features. This allows for the detection of patterns even when additional information is not provided with the input data. Semi-supervised learning is a method well suited where only partially labeled data exists, but the algorithm aims to fully benefit from provided information either labeled or unlabeled [54]. Reinforcement learning is agent-based learning interacting with the environment. Therefore, the algorithm learns based on rewards from correct prediction and penalties from incorrect guesses (i.e. trial and error methodology) [55].

2.3.7 Deep learning packages analysis

TensorFlow is one of the most popular and fastest evolving open source deep learning tools [37]. It is compatible with both GPU/CPU computation and is well-suited for working with multi-dimensional arrays. One downside could be the low-level API which makes it not the ideal choice for the direct creation of deep learning algorithms. Keras (open source) can support backends such as CNTK, TensorFlow, and Theano and its simple API makes it straightforward to implement. PyTorch is among the more flexible programming packages in Python and supports tensor computation and GPU acceleration. Its dynamic graph and easy debugging make it a strong option to choose. At its core, it uses CPU and GPU tensor and NN backends. Therefore, PyTorch brings speed and flexibility to deep learning models despite needing third-party visualization [56]. Regardless of the choice in packages, the input data's magnitude and quality will strongly impact the predictive power of the resulting model.

2.3.8 One-hot encoding

The simplest method is to use one-hot encoding by constructing a matrix of one and zeros where one represents the existence of the element at a specific position of the sequence. One-hot encoding

is easy to implement and has been proven to be effective in many cases, yet it is highly memory intensive and struggles to capture the relationship between amino acids in protein sequences. This large and sparse encoding often leads to complications in training as a result of the inherently high dimensionality.

2.3.9 Integer encoding

Integer encoding is implemented by representing each amino acid by an integer. For integer encoding one observed drawback is the tendency of the system to assume linear relationships among the provided labels. For example, if the labels are 1, 2, and 3, the system assumes a relationship between the amino acids (such that 1 is closer to 2 than 3) that are assigned to these labels. As a result, orthogonality between the labels matters. However, integer encoding is often used with a linear embedding layer (see `tf.keras.layers.embedding`) whereby integer encoding calls an embedding column like a "lookup" table.

2.3.10 Property-based encoding

Some practical encodings are obtained based on the physiochemical properties (e.g. charge, hydrophobicity, and size) of the sequences [57]. One example is the principal components score Vectors of Hydrophobic, Steric, and Electronic properties (VHSE8) [58].

2.3.11 Desired data distribution for DL algorithms

Gaussian-like distributions tend to have better performance in deep learning. Therefore, defining scoring functions that change the distribution of labels is a viable option. Power transform functions are used to reduce the skewness of data and make more Gaussian-like distributions. For example, normal power transform functions may take the nth root or the nth order logarithm of the variable. More advanced power transformers include Box-Cox and Yeo-Johnson methods. To obtain new distributions with the mentioned methods, the Scipy [59] library or sci-kit-learn preprocessing package called as `PowerTransformer` can be used.

```
# Example implementation
import scipy
```

```
import scipy.stats
Yeo-Johnson = scipy. stats.yeojohnson(label_List)
Box_Cox = scipy. stats.boxcox(label_List)
```

2.3.12 Activation function

One suggestive method is to use ReLU in all hidden layers and choose an appropriate activation function for the output layer to match the distribution of data with the nature of the task. Generally, for the output layer, sigmoid (for binary-class), softmax, and tanh (for multi-class) are used for classification tasks, and linear activation function is used for regression.

2.3.13 Evaluation metrics

Evaluation metrics are representative of algorithm performance and should be considered within the context of the nature of the problem and origin of the input data. As an example, in biased data when 90 percent of the population is in category 1 and the remainder are in category 2, the algorithm probably predicts all the data to be in the first category. In this case, if one wants to rely on the metrics, the accuracy will be 90 percent which does not address the performance of the algorithm. In this case, the confusion matrix provides useful information about the number of false positives, false negatives, true positives, and true negatives.

BIBLIOGRAPHY

[1] B.L. Hogan. Bone morphogenetic proteins: multifunctional regulators of vertebrate development. *Genes Dev*, 10:1580–1594, 1996.

[2] J. Schlessinger. Cell signaling by receptor tyrosine kinases. *Cell*, 103:211–225, 2000.

[3] V. Syrovatkina, K.O. Alegre, R. Dey, et al. Regulation, signaling, and physiological functions of g-proteins. *J Mol Biol*, 428:3850–3868, 2016.

[4] H.W. Hellinga and J.S. Marvin. Protein engineering and the development of generic biosensors. *Trends Biotechnol*, 16:183–189, 1998.

[5] N.K. Mishra, J. Chang, and P.X. Zhao. Prediction of membrane transport proteins and their substrate specificities using primary sequence information. *PLoS One*, 9:e100278–e100278, 2014.

[6] T. Yang, J.C. Wu, C. Yan, et al. Virtual screening using molecular simulations. *Proteins*, 79:1940–1951, 2011.

[7] E.E. Wrenbeck, M.S. Faber, and T.A. Whitehead. Deep sequencing methods for protein engineering and design. *Curr Opin Struct Biol*, 45:36–44, 2017.

[8] N. Kronqvist, J. Löfblom, A. Jonsson, et al. A novel affinity protein selection system based on staphylococcal cell surface display and flow cytometry. *Protein Eng Des Sel*, 21:247–255, 2008.

[9] K.K. Yang, Z. Wu, and F.H. Arnold. Machine-learning-guided directed evolution for protein engineering. *Nat Methods*, 16:687–694, 2019.

[10] H. Bohr, J. Bohr, S. Brunak, et al. A novel approach to prediction of the 3-dimensional structures of protein backbones by neural networks. *FEBS Lett*, 261:43–46, 1990.

[11] Y. Ofran and B. Rost. Predicted protein-protein interaction sites from local sequence information. *FEBS Lett*, 544:236–239, 2003.

[12] J.J. Ward, L.J. McGuffin, B.F. Buxton, et al. Secondary structure prediction with support vector machines. *Bioinformatics*, 19:1650–1655, 2003.

[13] N.V. Petrova and C.H. Wu. Prediction of catalytic residues using support vector machine with selected protein sequence and structural properties. *BMC Bioinformatics*, 7:1–12, 2006.

[14] B.Q. Li, K.Y. Feng, L. Chen, et al. Prediction of protein-protein interaction sites by random forest algorithm with mrmr and ifs. *PLoS One*, 7:1–10, 2012.

[15] L. Quan, Q. Lv, and Y. Zhang. Strum: Structure-based prediction of protein stability changes upon single-point mutation. *Bioinformatics*, 32:2936–2946, 2016.

[16] M. Tahir, H. Tayara, and K.T. Chong. irna-pseknc(2methyl): Identify rna 2'-o-methylation sites by convolution neural network and chou's pseudo components. *J Theor Biol*, 465:1–6, 2019.

[17] J.D. Bloom, S.T. Labthavikul, C.R. Otey, et al. Protein stability promotes evolvability. *Proc Natl Acad Sci U S A*, 103:5869–5874, 2006.

[18] Y. Saito, M. Oikawa, H. Nakazawa, et al. Machine-learning-guided mutagenesis for directed evolution of fluorescent proteins. *ACS Synth Biol*, 7:2014–2022, 2018.

[19] E.C. Alley, G. Khimulya, S. Biswas, et al. Unified rational protein engineering with sequence-based deep representation learning. *Nat Methods*, 16:1315–1322, 2019.

[20] S. Biswas. Low-n protein engineering with data-efficient deep learning: A paradigm for low-n protein engineering. *Preprint*, pages 1–39, 2020.

[21] B.E. Suzek, Y. Wang, H. Huang, et al. Uniref clusters: A comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31:926–932, 2015.

[22] M. Crawshaw. Multi-task learning with deep neural networks: A survey. *Preprint*, 2020.

[23] J. Im, B. Park, and K. Han. A generative model for constructing nucleic acid sequences binding to a protein. *BMC Genomics*, 20:1–13, 2019.

[24] J.E. Ness, S. Kim, A. Gottman, et al. Synthetic shuffling expands functional protein diversity by allowing amino acids to recombine independently. *Nat Biotechnol*, 20:1251–1255, 2002.

[25] R.D. Gupta and D.S. Tawfik. Directed enzyme evolution via small and effective neutral drift libraries. *Nat Methods*, 5:939–942, 2008.

[26] M.K.M. Engqvist and J. Nielsen. Ant: Software for generating and evaluating degenerate codons for natural and expanded genetic codes. *ACS Synth Biol*, 4:935–938, 2015.

[27] T.M. Jacobs, H. Yumerefendi, B. Kuhlman, et al. Swiftlib: Rapid degenerate-codon-library optimization through dynamic programming. *Nucleic Acids Res*, 43:1–9, 2015.

[28] M.L. Menéndez, J.A. Pardo, L. Pardo, et al. The jensen-shannon divergence. *J Franklin Inst*, 334:307–318, 1997.

[29] V. Bewick, L. Cheek, and J. Ball. Statistics review 12: Survival analysis. *Crit Care*, 8:389–394, 2004.

[30] TensorFlow. Index @ www.tensorflow.org, 2017.

[31] F. Chollet et al. Keras, 2015.

[32] D. Mazza and M. Pagani. Automatic differentiation in pcf. *Proc ACM Program Lang*, 5:1–4, 2021.

[33] F. Pedregosa, G. Varoquaux, A. Gramfort, et al. Scikit-learn: Machine learning in python. *J Mach Learn Res*, 12:2825–2830, 2012.

[34] W. McKinney. Data structures for statistical computing in python. In *Proc 9th Python Sci Conf*, pages 56–61, 2010.

[35] C.R. Harris, K.J. Millman, S.J. van der Walt, et al. Array programming with numpy. *Nature*, 585:357–362, 2020.

[36] F. Pedregosa, G.G. Varoquaux, A. Gramfort, et al. Scikit-learn: Machine learning in python. *J Mach Learn Res*, 12:2825–2830, 2011.

[37] M. Abadi, P. Barham, J. Chen, et al. Tensorflow: A system for large-scale machine learning. In *Proc 12th USENIX Symp Oper Syst Des Implementation, OSDI 2016*, pages 265–283, 2016.

[38] R. Rao, N. Bhattacharya, N. Thomas, et al. Evaluating protein transfer learning with tape. *Adv Neural Inf Process Syst*, 32:9689, 2019.

[39] T.A. Whitehead, A. Chevalier, Y. Song, et al. Optimization of affinity, specificity and function of designed influenza inhibitors using deep sequencing. *Nat Biotechnol*, 30:543–548, 2012.

[40] R. Shroff, A.W. Cole, D.J. Diaz, et al. Discovery of novel gain-of-function mutations guided by structure-based deep learning. *ACS Synth Biol*, 9:2927–2935, 2020.

[41] Z. Zhao and X. Gong. Protein-protein interaction interface residue pair prediction based on deep learning architecture. *IEEE/ACM Trans Comput Biol Bioinforma*, 16:1753–1759, 2019.

[42] Q. Zhang, M. Zhang, T. Chen, et al. Recent advances in convolutional neural network acceleration. *Neurocomputing*, 323:37–51, 2019.

[43] Y. Yu, X. Si, C. Hu, et al. A review of recurrent neural networks: Lstm cells and network architectures. *Neural Comput*, 31:1235–1270, 2019.

[44] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput*, 9:1735–1780, 1997.

[45] K. Cho, B. Van Merriënboer, C. Gulcehre, et al. Learning phrase representations using rnn

encoder-decoder for statistical machine translation. In *EMNLP 2014 - 2014 Conf Empir Methods Nat Lang Process Proc Conf*, pages 1724–1734, 2014.

[46] J. Bergstra, D. Yamins, and D.D. Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *30th Int Conf Mach Learn ICML 2013*, pages 115–123, 2013.

[47] S. Raschka. Model evaluation, model selection, and algorithm selection in machine learning. *Preprint*, 2018.

[48] G. Chao, W.L. Lau, B.J. Hackel, et al. Isolating and engineering human antibodies using yeast surface display. *Nat Protoc*, 1:755–768, 2006.

[49] D.R. Woldring, P.V. Holec, H. Zhou, et al. High-throughput ligand discovery reveals a sitewise gradient of diversity in broadly evolved hydrophilic fibronectin domains. *PLoS One*, 10:e0138956, 2015.

[50] D.R. Woldring, P.V. Holec, L.A. Stern, et al. A gradient of sitewise diversity promotes evolutionary fitness for binder discovery in a three-helix bundle protein scaffold. *Biochemistry*, 56:1656–1671, 2017.

[51] M.A. Kruziki, S. Bhatnagar, D.R. Woldring, et al. A 45-amino-acid scaffold mined from the pdb for high-affinity ligand engineering. *Chem Biol*, 22:946–956, 2015.

[52] M.A. Kruziki, V. Sarma, and B.J. Hackel. Constrained combinatorial libraries of gp2 proteins enhance discovery of pd-l1 binders. *ACS Comb Sci*, 20:423–435, 2018.

[53] L.A. Stern, C.M. Csizmar, D.R. Woldring, et al. Titratable avidity reduction enhances affinity discrimination in mammalian cellular selections of yeast-displayed ligands. *ACS Comb Sci*, 19:315–323, 2017.

[54] X. Zhu and A.B. Goldberg. *Introduction to semi-supervised learning*, volume 3. Synth Lect Artif Intell Mach Learn, 2009.

[55] R.S. Sutton and A.G. Barto. *Reinforcement learning: An introduction*. Bradford Book, 1998.

[56] G. Nguyen, S. Dlugolinsky, M. Bobák, et al. Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey. *Artif Intell Rev*, 52:77–124, 2020.

[57] K.K. Yang, Z. Wu, C.N. Bedbrook, et al. Learned protein embeddings for machine learning. *Bioinformatics*, 34:2642–2648, 2018.

[58] H.U. Mei, Z.H. Liao, Y. Zhou, et al. A new set of amino acid descriptors and its application in peptide qsars. *Pept Sci Orig Res Biomol*, 80:775–786, 2005.

[59] P. Virtanen, R. Gommers, T.E. Oliphant, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nat Methods*, 17:261–272, 2020.

CHAPTER 3

PROTEIN FITNESS PREDICTION IS IMPACTED BY THE INTERPLAY OF LANGUAGE MODELS, ENSEMBLE LEARNING, AND SAMPLING METHODS

[1]

Abstract

Advances in machine learning (ML) and the availability of protein sequences via high-throughput sequencing techniques have transformed the ability to design novel diagnostic and therapeutic proteins. ML allows protein engineers to capture complex trends hidden within protein sequences that would otherwise be difficult to identify in the context of the immense and rugged protein fitness landscape. Despite this potential, there persists a need for guidance during the training and evaluation of ML methods over sequencing data. Two key challenges for training discriminative models and evaluating their performance include handling severely imbalanced datasets (e.g., few high-fitness proteins among an abundance of non-functional proteins) and selecting appropriate protein sequence representations (numerical encodings). Here, we present a framework for applying ML over assay-labeled datasets to elucidate the capacity of sampling techniques and protein-encoding methods to improve binding affinity and thermal stability prediction tasks. For protein sequence representations, we incorporate two widely used methods (One-Hot encoding and physiochemical encoding) and two language-based methods (next-token prediction, UniRep; masked-token prediction, ESM). Elaboration on performance is provided over protein fitness, protein size, and sampling techniques. In addition, an ensemble of protein representation methods is generated to discover the contribution of distinct representations and improve the final prediction score. We then implement multiple criteria decision analysis (MCDA; TOPSIS with entropy weighting), using multiple metrics well-suited for imbalanced data, to ensure statistical rigor in ranking our methods. Within the context of these datasets, the synthetic minority oversampling technique (SMOTE) outperformed undersampling while encoding sequences with One-Hot, UniRep,

[1]This chapter is adapted from content published in "Protein Fitness Prediction Is Impacted by the Interplay of Language Models, Ensemble Learning, and Sampling Methods." All rights reserved. For more information, visit https://doi.org/10.3390/pharmaceutic3A.15051337.

and ESM representations. Moreover, ensemble learning increased the predictive performance of the affinity-based dataset by 4% compared to the best single-encoding candidate (F1-score = 97%), while ESM alone was rigorous enough in stability prediction (F1-score = 92%).

Keywords

Machine learning; protein fitness prediction; embeddings; sequence representation; imbalanced assay-labeled datasets; sampling methods; ensemble learning; MCDA; TOPSIS

3.1 Introduction

Proteins are biological machines involved in almost all biological processes [1, 2, 3, 4]. These molecules are made of amino acids that fold into 3-dimensional structures and perform life-sustaining biological functions [5]. Protein engineering practices aim to modify proteins to redirect what has already evolved in nature and address the industrial and medical needs of modern society [6, 7]. This has been a challenging task due to the astronomical number of possible mutations and the complex sequence-function relationship of the proteins (i.e., fitness landscape) [8]. Protein fitness—a measure of how well a protein performs a task of interest—is influenced by a variety of factors, including its structure, stability, and interactions with other molecules. In protein engineering campaigns, where protein function is modified by experimental approaches rather than natural selection, proteins with high fitness are those that perform well within relevant experimental assays, whereas proteins with low fitness result in reduced activity, altered specificity, or decreased stability. To overcome the challenge of finding high-fitness proteins among mostly non-functional mutants, various experimental and computational techniques were developed. Recently, machine learning (ML) has shown promise as a tool to supplement already established techniques, such as rational design and directed evolution [9, 10, 11, 12]. Unlike directed evolution, ML models can learn from non-functional mutants instead of simply discarding them during enrichment for functional clones. ML-assisted protein engineering, therefore, has potential as a time-efficient and cost-effective approach to searching for desired protein functionality. This provides a unique opportunity to create smart protein libraries, elevate and accelerate directed evolution and rational design strategies, and finally, enhance the probability of finding unexplored high-fitness variants in the protein fitness landscape [13, 14, 15]. Machine learning methods have attained a high success rate in predicting essential protein properties (i.e., protein fitness) including secondary structure, solubility, binding affinity, flexibility, and specificity [16, 17, 18, 19, 20]. Despite these recent milestones, to obtain generalizability and robustness in ML models, further explorations in different protein fitness prediction tasks and training details are required.

Dealing with protein fitness landscape challenges will require us to view proteins from a new

perspective that supplements our biochemical knowledge with lessons from written languages. Recent advances in ML and artificial intelligence have applied natural language processing (NLP) methods to identify context-specific patterns from written or spoken text. NLP tasks learn how words function grammatically (syntax) and how they deliver meaning within themselves and in surrounding words (semantics) [21, 22]. This has given rise to virtual assistants with voice recognition and sentiment analysis of text from diverse languages [23, 24]. Similarly, protein engineering can leverage these NLP tools—treating a string of amino acids as if they were letters on a page—to understand the language of proteins, providing a promising route to capture nuances (e.g., epistatic relationships, functional motifs) in complex sequence–function mappings [25, 26]. The rapid expansion of publicly available protein sequence data (e.g., UniProt [27], SRA [28]) further supports the use of big data and language models in the domain of protein engineering [29]. Self-supervised language models learn the context of the provided text by reconstructing the masked tokens/linguistic units of the text string using the unmasked parts. For the context of protein engineering, pre-trained protein language models—carrying valuable information about the epistasis/interaction of amino acids—can be applied to downstream tasks by extracting the optimized weight functions as a fixed-size vector (embedding) [25, 30, 31]. Among early embedding developments, Alley et al. introduced UniRep [32], a deep learning model that was trained on 24 million unique protein sequences to perform the next amino acid prediction tasks for extracting information about the global fitness landscape of proteins. Rives et al. trained ESM, a language model for masked amino acid prediction tasks, on over 250 million protein sequences [33]. The learned representations—including UniRep [32], ESM [33], TAPE [34], and ProteinBERT [35] have generated promising results in diverse areas such as predicting protein fitness, protein localization, protein-protein interaction, and disease risk of mutations in terms of improved prediction scores, increased generalizability, and mediated data requirements [36, 37, 38, 39, 40]. Using embeddings for sequence representations (transfer learning) enables knowledge transfer between protein domains and future prediction tasks by further optimizing the already-learned weights. For example, Min et al. obtained a 20% increase in the F1-score (the

harmonic mean of precision and recall) for a heat shock protein identification task when training their NLP-based model, DeepHSP [41], on top of pre-trained representations.

In this study, we perform protein sequence fitness prediction with ML techniques to demonstrate how model performance varies given the choice of protein representation, protein size, and the biological attribute (e.g., binding affinity and thermal stability) to be predicted. This work provides actionable insights for effectively building discriminative models and improving their prediction scores via sampling techniques and ensemble learning. As efficient use of embedding methods on experimental datasets is in its infancy, rigorous studies are needed to gain new insights into the performance of the pre-trained models given various training conditions and distinct biological function predictions. Importantly, embedding methods have been trained over millions of protein sequences in public databases and have produced high performance in certain fitness tasks (e.g., stability prediction), while they may not do as well in all fitness prediction tasks. To this end, we used two large datasets that were representative of common protein engineering tasks. First, we leveraged a highly imbalanced dataset (93% non-functional; Table 3.1), consisting of our previously described affinity-evolved affibody sequences [42] to explore NLP-driven practices. We then expanded our analysis to include thousands of protein sequences labeled with their experimentally measured stabilities (melting temperatures, Tm) obtained from the Novozymes Enzyme Stability Prediction (NESP) dataset [43]. Thus, with our two datasets having unique attributes, we were well positioned to address multiple questions: (i) How do different representation methods perform in predicting distinct fitness attributes such as stability or affinity? (ii) How do sampling methods perform in imbalanced protein datasets? (iii) Is ensemble learning over different protein representations helpful in boosting the performance of discriminative models? (iv) How do we rank model performances while using multiple conflicting metrics in ML prediction tasks? By addressing these challenges, we also gain direct insights for model interpretation and reveal the features that are most important for discriminating between fit and non-fit sequences (Figures 3A.1, 3A.6, and 3A.8). We discovered that oversampling (especially SMOTE) generally outperformed the undersampling techniques. In addition, ensemble over representations greatly improved the predictive performance in the

affibody data both using single and multiple performance metrics via multiple criteria decision analysis (MCDA) [44]. For protein representations (e.g., single encoders), UniRep and One-Hot outperformed other methods in the affibody (affinity) dataset while ESM achieved the best score in stability prediction in NESP. Finally, it was observed that the performance of various protein representation methods is strongly impacted by protein sequence length.

Table 3.1 Dataset attributes and prediction tasks.

Dataset	**Task**	**Fitness**	**Model**	**Attributes**
Affibody	Classification	Binding Affinity	Logistic Regression	82,663 non-binders, 6077 binders
NESP	Classification	Stability	Logistic Regression	3743 high-stability, 1311 low-stability
NESP	Regression	Stability	Random Forest Regressor	18,190 total

3.2 Materials and Methods

3.2.1 Obtaining Experimentally Labeled Sequence Data

Two different datasets with varying data characteristics were explored. The first is our experimental data of affibody sequences that previously were iteratively evolved for binding affinity and specificity against a panel of diverse targets [45]. The second collection of labeled protein sequences was obtained from the recently released Kaggle dataset wherein numerous proteins (n= 18,190) of various lengths are labeled according to their thermal stability (Tm). This dataset, NESP, was filtered to only include sequences characterized at pH = 7. For the affibody dataset, raw sequence data were cleaned by removing any sequences that contained stop codons or invalid characters. Afterwards, the frequency of each unique sequence in the experimental steps was tabulated. Infrequent sequences appearing fewer than ten and four times (within magnetic activated cell sorting (MACS) and fluorescent activated cell sorting (FACS), respectively) were treated as background and removed from the analysis. Note that the more stringent frequency removal for MACS was mainly due to the experiment type and higher probability to introduce noise in the dataset. After removing the background, sequences from MACS and FACS were combined to form

the final high-fitness population of binders. The non-binding population included the initial affibody sequence pool, which did not appear in the enriched population of the binder sequences. The initial affibody sequences that were within one hamming distance (i.e., a single amino acid mutation) of any enriched sequence were removed as well to account for potential errors encountered during deep sequencing. All affibody sequences were exactly 58 amino acids in length with mutations present at up to 17 of these positions.

3.2.2 Obtaining the Sequence Representations

We obtained four different numerical representations for our sequence data: One-Hot and physiochemical encoding, UniRep, and ESM embeddings. One-Hot encoding refers to building a matrix (amino acids × protein length) and filling it with one when there is a specific amino acid in the given position, filling the rest with zeros. For physiochemical encoding, we used the modlamp [46] package in python, which is used for extracting the physical features from protein sequences. There were two types of physical features represented in the modlamp package (global and peptide descriptors). All the global (e.g., sequence length, molecular weight, aliphatic index, etc.) and local physiochemical features based on the Eisenberg scale were extracted for this analysis (twenty in total).

Embedding refers to the continuous representation of the protein sequence in a fixed-size vector, and it should contain meaningful information about proteins [47]. For example, in the embedding visualization of amino acids in low dimensions for both UniRep and ESM, similar amino acids (in terms of size, charge, hydrophobicity, etc.) were close to each other. For UniRep representation, we used the 1900 dimension and mean representation over layers. We used Jax_UniRep for obtaining the UniRep embeddings, https://github.com/ElArkk/jax-unirep (accessed on 20 August 2022). UniRep uses the mLSTM structure for performing next-token prediction, and it was trained on 24 million sequences in the Uniref50 dataset with 18 M parameters. For ESM, we chose ESM2 [48] with 1280 vector dimensions and 650 M parameters and means over layer representations. GitHub for ESM is https://github.com/facebookresearch/esm (accessed on 21 January 2023).

3.2.3 Sampling and Splitting

Sampling refers to choosing a random subset of data to represent the underlying population. Three different sampling methods were tested for our severely imbalanced affibody dataset: undersampling, random oversampling, and the synthetic minority oversampling technique (SMOTE) [49]. Due to the sparse and rugged nature of the protein fitness landscape, it is common for experimental data obtained in the protein domain to be highly imbalanced. One practical approach for resolving the imbalanced dataset issue is using sampling techniques when training the dataset. Oversampling is randomly repeating the minority class examples; thus, it could be prone to overfitting in comparison to undersampling. However, undersampling may discard useful information, especially in severely imbalanced datasets, as it is removing many samples from the majority class. SMOTE is a more recent addition to sampling methods, and it is oversampling the minor population by synthetically generating more instances that are highly similar to the minority class. While SMOTE has shown promising results in increasing the prediction performance for various imbalanced datasets [50, 51, 52], there are also studies indicating undersampling superior performance compared to oversampling methods [53, 54]. As a result, we examined the performance of all three sampling techniques to validate which sampling method performs well within our wet-lab protein dataset over different encoding methods.

For splitting the data points within the test set in an imbalanced dataset, sampling equally from each class may lead to an overestimation of the model performance [55]. As a result, we made sure that the test set distribution follows the initial data distribution (93% naïve vs. 7% enriched).

3.2.4 Algorithm Selection and Training Details

For classification, logistic regression (LR) was chosen and L2 penalization (Ridge) was used to reduce the likelihood of overfitting. We reasoned that a simple logistic regression enables a fair comparison between cases. One regression task was also implemented over the NESP dataset with random forest regressor (RFR). We used regression to observe how models perform with increasing the prediction challenge, from binary prediction to actual label prediction. The rationale for using RFR was that the linear regression model was not viable to meet the prediction task complexity. For

a fair comparison between protein-encoding performances in regression, the RFR hyperparameters, max number of estimators and max_depth, were optimized with OPTUNA [56].

3.2.5 Ensemble Learning

To enhance the predictive performance of protein-encoding predictions, we designed a framework that integrates multiple encoding methods. Our experiments included two approaches: concatenation and a voting mechanism. In concatenation, the encodings were combined by adding them together, and we used the resulting representation as input for our predictive model. In voting, separate predictive models for each encoding method were trained. The final prediction was then calculated with the majority-voted label over a fixed test set.

3.2.6 Metrics and Statistical Analysis

One key metric we used for analyzing classification performance is the F1 score. By considering both precision and recall (Figure 1E), the F1 score is particularly well suited for evaluating the highly imbalanced data within our study (Table 3.1). Therefore, the model is trained to identify the positive instances among all positive predictions and minimize missing out the positive instances while predicting classes. Note that other classification metrics, such as confusion matrix values (TP, TN, FP, FN), are reported in the supplement figures. For regression analysis among NESP data, we used mean squared error (MSE) and R^2 to indicate how the models perform. MSE is the mean of the square of differences between the actual labels and the predicted values in the test set while R^2 represents the variation explained by the independent variables.

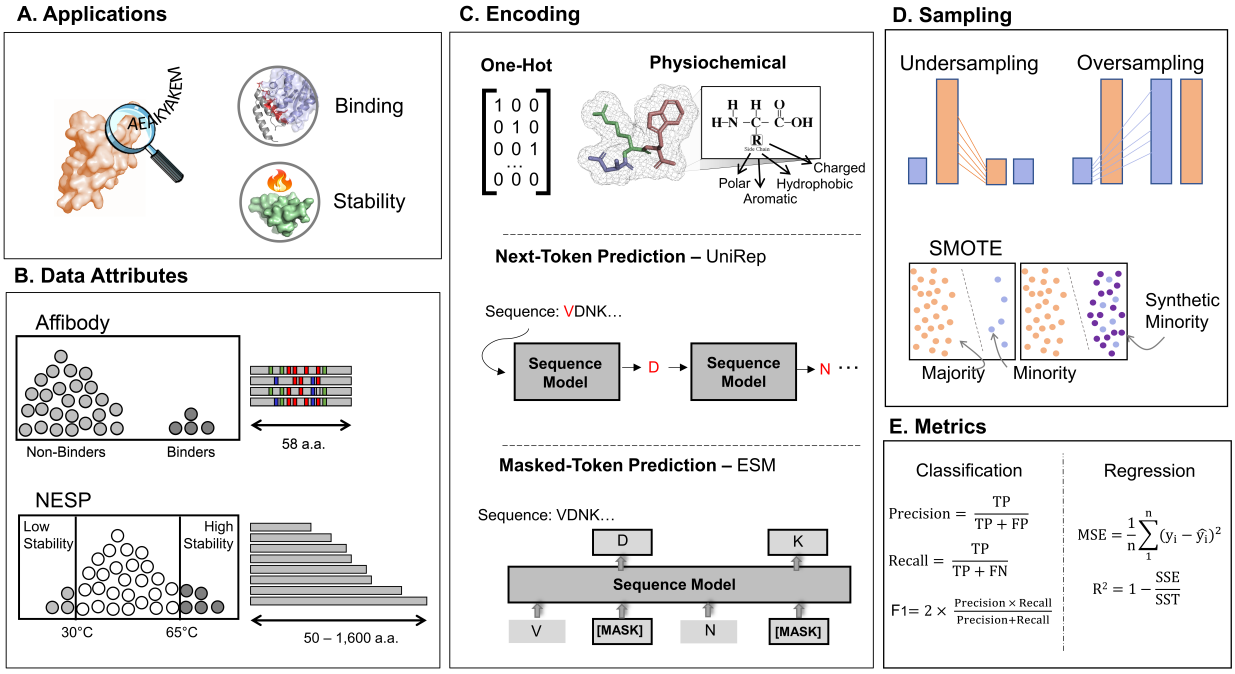


Figure 3.1 Overview of the Implemented Techniques, Data Attributes, and Evaluation Metrics. (A) Illustrates the use of sequence-function mapping to identify protein sequence functionality (e.g., therapeutics, diagnostics, enzymatic function). (B) Data attributes for the two datasets used in this study. The first dataset includes high-fitness protein binders among a pool of non-binder affibody sequences with up to 17 mutation sites. The other dataset includes a wide array of proteins with their associated melting point. (C) One-hot encoding, physicochemical encoding, and pre-trained models were used to encode the protein sequences present in our datasets. All present protein amino acid information is in a machine-readable format but in different ways. One-hot encoding converts each amino acid to a binary vector of all 0s but 1 where it belongs to its position in the matrix. In physicochemical encoding, each amino acid is represented by its physiochemical characteristics, such as polarity, charge, size, etc. Pre-trained models are trained over a large corpus of unlabeled data capturing the syntax and semantics of protein language via NLP-driven models, such as next-token prediction (e.g., UniRep) and masked token prediction (e.g., ESM). (D) The sampling methods used in this study are undersampling, oversampling, and synthetic minority oversampling techniques (SMOTE). (E) The main metrics used for evaluating the performance of prediction tasks (classification and regression) are defined (a complete list of performance metrics is listed in Figure 3A.11).

Experiments were implemented with multiple random seeds (20 in affibody and 30 in NESP dataset) to obtain a distribution of performances for each pair of encoding and sampling methods in each fitness prediction task. Then we implemented multiple statistical tests to confirm if the obtained differences were significant. Analysis of variance among the groups was performed with ANOVA [57]. After obtaining significant results in ANOVA, post hoc methods were implemented

to account for family-wise error rates. Here, we implemented two post hoc methods, Bonferroni [58] and Tukey [59], for adjusting the p-values and reducing the risk of type-1 error. The null hypothesis assumes that the performances of the methods are similar and when rejected, we consider the methods to be statistically significant in their obtained output. The results for multiple seeds are shown with violin plots where the white dots represent the mean values. A complete collection of statistical analyses for comparing the significance among the means are located in the supplementary information.

3.2.7 Multiple Criteria Decision Analysis (MCDA)

We used the F1-score as our primary classification metric in classification to optimize the algorithms based on finding the rare positive sequences. Depending on specific applications, the user may need to choose different criteria for analyzing the ML predictive performance. Note that it is also generally advised to use multiple metrics to establish more rigorous analyses, specifically in imbalanced datasets [60, 61]. Therefore, we incorporated five more classification metrics in addition to F1-score and implemented MCDA, which is a robust approach for decision-making (i.e., ranking alternatives based on multiple, often conflicting, criteria). In our study, the alternatives are the choice of protein representation within different sampling methods. The criteria (classification metrics) used for this MCDA include F1-score, false positive rate (FPR), true positive rate (TPR), precision, negative predictive value (NPV), and false discovery rate (FDR). FPR and TPR measure the model's ability to identify the positive and negative classes. Precision quantifies the number of correctly positive classes among all being predicted as positive, while NPV measures this for the negative class. FDR measures the number of false positives over all instances that are predicted as positives.

The performance of each encoding and sampling technique was recorded based on all six mentioned criteria. For implementing the decision-making, we chose a well-established and widely used MCDA method: the technique for order of preference by similarity to ideal solution (TOPSIS) [62]. TOPSIS finds the optimal solution rooted in the idea that the best alternative should have the minimum Euclidean distance from the positive ideal solution and maximum distance from the

negative ideal solution.

For the implementation of TOPSIS, the PyTopsis Python package was utilized (https://github.com/shivambehl/PyTopsis, accessed on 4 April 2023). This method requires three primary inputs: a decision matrix representing alternative scores across various criteria, a list of weights reflecting the importance of each criterion, and a list of signs where -1 indicates a criterion to be minimized, and 1 indicates maximization. Utilizing this framework, we constructed our decision matrix and defined the optimization direction for each criterion in classification tasks.

Weights for the criteria were determined through two approaches: subjective weighting, based on the decision-maker's preferences, and objective weighting, derived from a numerical analysis of the decision matrix. Both weighting methods were employed to assess and compare their efficacy in ranking alternatives. In the subjective weighting approach, higher weights were assigned to precision and FPR metrics to emphasize the identification of positive instances.

For objective weighting, we applied Shannon's entropy method [63] to calculate the entropy values from the decision matrix, which then informed the weighting process. The formula for calculating weights based on entropy, where x_{ij} represents each entry in the matrix, n the number of alternatives, and m the number of criteria, is as follows:

$$\text{Normalizing the decision matrix value:} \quad r_{ij} = \frac{x_{ij}}{\sum_{i=1}^{n} x_{ij}} \tag{3.1}$$

$$\text{Calculating Entropy for each criterion:} \quad E_j = -k \sum_{i=1}^{n} (r_{ij} \ln r_{ij}), \quad k = \frac{1}{\ln(n)} \tag{3.2}$$

$$\text{Calculating weight for each criterion:} \quad w_j = \frac{1 - E_j}{\sum_{j=1}^{m} (1 - E_j)} \tag{3.3}$$

The results from TOPSIS need to be validated via statistical methods to ensure the correct ranking among alternatives (i.e., difference between performances is not random but significant). Therefore, we applied multivariate analysis of variance (MANOVA) [64] followed by a post-hoc method, Tukey [59], to analyze the result significance overall and between a pair of alternatives, respectively.

Figure 3.1 provides an overview of the data attributes (e.g., protein size, protein fitness) that

will be predicted and alternatives (e.g., protein encodings within different sampling methods) that will be compared.

3.3 Results

3.3.1 Sequence-Function Mapping Obtained from High-Throughput Selection Methods and Deep Sequencing Affibody Dataset

To investigate the impact of feature representation, ensemble learning, and sampling methods on predictive modeling, we engaged in several prediction tasks using deep sequencing data. Specifically, we focused on a classification task involving the affibody dataset. The goal was to predict the scarce high-affinity binder class from a pool of non-binders. Details of the sequences obtained after data cleaning and the specific prediction tasks are presented in Table 3.1. For the NESP dataset, we categorized the data into two stability classes based on their thermal stability (Tm): low-stability (Tm $\leq 35°C$) and high-stability (Tm $\geq 60°C$). Additionally, we implemented a regression task to predict the actual Tm values, increasing the prediction difficulty and providing insights into the performance of different protein encodings. Only sequences measured at pH 7 were included in this analysis.

Detailed results and additional analyses of the NESP dataset are discussed in Section 3.5 and provided in the supplementary materials.

3.3.2 Physiochemical Feature Encoding, Interpretable yet Lower Predictive Capacity

The classification results in physiochemical encodings are shown in Figures 2 and 3. We ranked the leading features in discriminating non-binder and binder classes and listed the encoding method's F1 score in different sampling methods. The physiochemical encoding performance was not among the lead encoding methods, yet it achieved a high F1 score with only 20 features. It also provided insights into how physical features correlate with each other in the given data (Figure 3A.1).

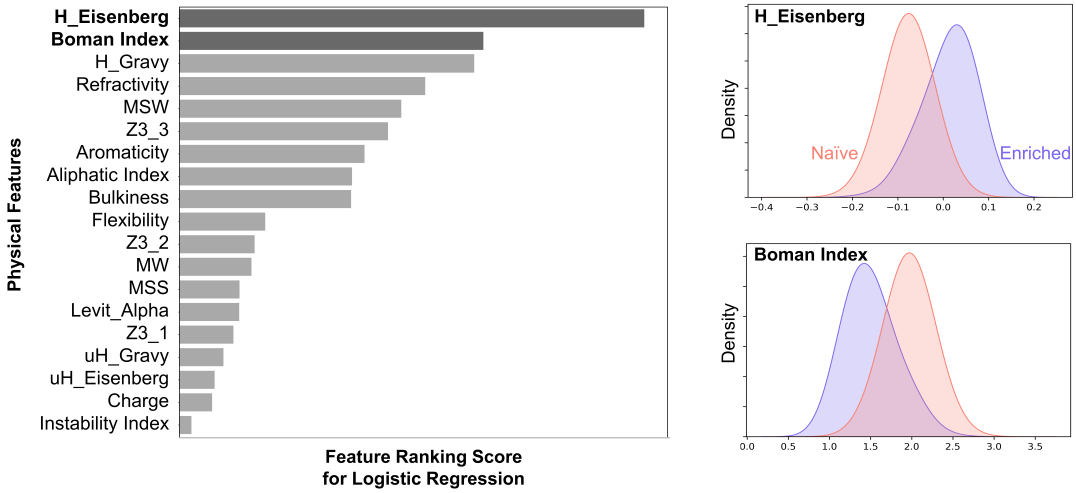


Figure 3.2 The lead physical features in naïve and enriched class discriminations in affinity-based data were H-Eisenberg, Boman Index, and H- Gravy. Gravy and Eisenberg capture hydrophobicity scales. The Boman Index is a measure of the protein's ability to interact with its environment based on the solubility of individual residues. The enriched proteins in our library have gone through negative screening and are specific to their target. Therefore, there is a shift to a lower Boman index for this population. Note that the plot is the result of oversampling, SMOTE, in the logistic regression task.

3.3.3 Comparison of Encoding and Sampling Methods

Once the primary physical features of high-affinity binders were established, we evaluated the performance of various protein representation techniques alongside our chosen sampling methods. This analysis aimed to assess how each encoding method influenced the predictive capability regarding the fitness of proteins.

Our findings revealed distinct performance variations among the encoding methods. Specifically, One-Hot and UniRep emerged as the most effective techniques, showing superior performance in several metrics. Concerning sampling strategies, the Synthetic Minority Over-sampling Technique (SMOTE) consistently enhanced the F1 score across almost all scenarios. Figure 3.4 displays the distribution of F1-scores, generated using 20 different random seeds, as violin plots, which visually represent the density and distribution of the scores.

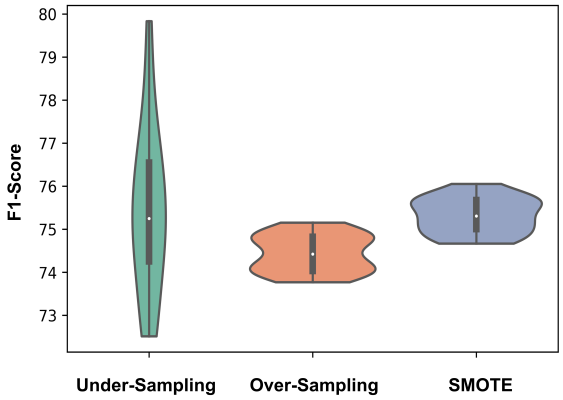


Figure 3.3 When physical features were used to encode the affibody sequences, the mean F1-score was 75.5% with SMOTE. Both SMOTE and undersampling methods were similarly effective, with no significant difference in performance (i.e., did not reject the null hypothesis). The violin plots are created over 20 random seeds for each sampling method.

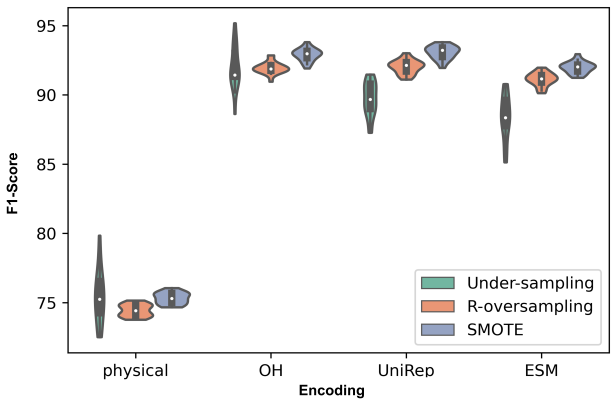


Figure 3.4 Performance analysis of encoding methods highlighting the comparative weaknesses of physical feature-based encodings and the effectiveness of the SMOTE sampling method.

The analysis incorporated protein sequences encoded using physical features, One-Hot, UniRep, and ESM to perform classification tasks within the affibody dataset. For each encoding strategy, various sampling methods including undersampling, random oversampling, and SMOTE were evaluated. The resultant F1 scores observed over 20 random seeds, provided a robust basis for statistical analysis.

A one-way ANOVA was conducted to quantify the differences in performance across the encoding and sampling methods, yielding a p-value of 9.52×10^{-190}. This result signifies a statistically significant difference among the groups. Detailed post-hoc results for method ranking are presented in Figure 3A.2, further supporting the conclusions drawn from the comparative analysis.

3.3.4 Increased Generalizability and Predictive Performance via Ensemble Learning

Due to the varying performances of the protein encodings, we postulated that ensemble learning increases the models' predictive performance. As oversampling performed better than undersampling in three out of four encoding methods, we exclusively analyzed the ensemble learning for the two oversampling types (i.e., random oversampling and SMOTE). The physical encoding for this analysis was discarded since its performance was not as potent as the other encodings. Figure 3.5 shows the ensemble technique, voting, which remarkably enhanced the performance with respect to all methods with a mean F1-score of 97% over 20 random seeds.

As shown in Figure 3.5 and Figure 3A.3-4, voting notably enhanced the prediction score among the candidates, and SMOTE improved the performance in single encoders compared to random oversampling (R-oversampling). In pursuit of a more informed and transparent decision-making process among the discussed methods, we also integrated a Multi-Criteria Decision Analysis (MCDA) using the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) across five additional classification metrics beyond the F1-score. These classification criteria were evaluated over single encoders, the concat_all encoder, and the upvoting technique. Figure 3.6 presents a summary of our MCDA design along with the ranking results obtained from TOPSIS.

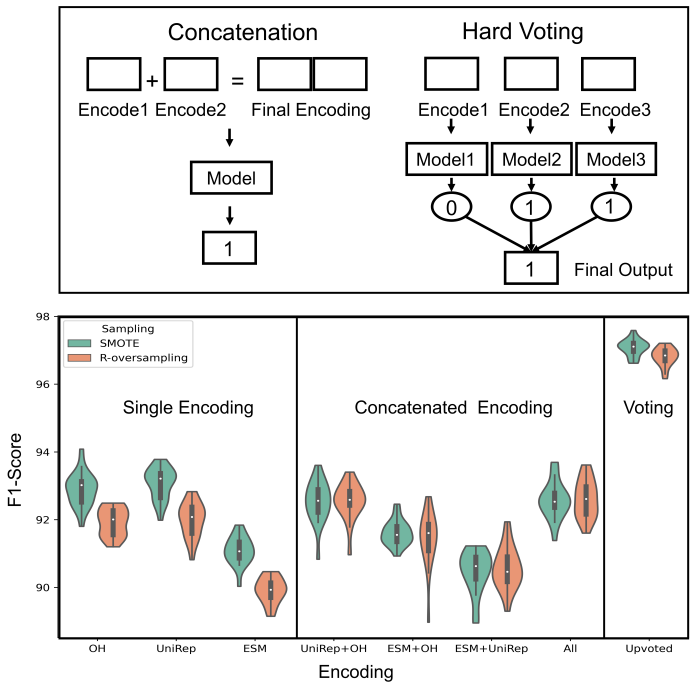


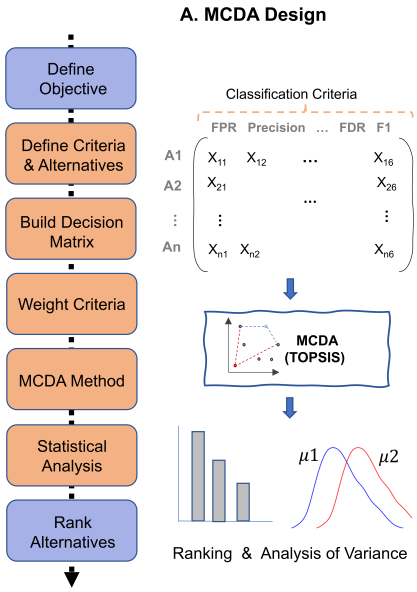
Figure 3.5 Voting substantially improved the predictive performance across all random initializations over different encoding methods. The illustrated plot is divided into three regions from left to right: single encoding methods, concatenation of encodings, and voting of predictions. The voting was executed such that each encoding was processed through a predictive model over the same dataset, with the final prediction obtained by majority voting. This figure highlights how voting enhances the models' robustness and generalizability. The concatenation of encodings performed similarly or worse than the best model among single encodings. The best model across all predictions was "Upvote" with oversampling methods, achieving a Mean-F1-score of 97% and a Mean-F1-score of 96.80% (no statistical significance detected among oversampling performances in upvoting). For a comprehensive summary of the statistical analysis and confusion matrix plots, refer to the supplementary materials (Figure 3A.3-4, Figure 3A.6).

3.3.5 MCDA Design and Statistical Validation

The MCDA design in the affibody dataset, with only 7% high-fitness population, enabled the comparison of encoding and sampling methods over multiple conflicting criteria. Additionally, selective weighting in MCDA allows users to bias the results towards more favorable outcomes, based on data attributes and specific applications. It is important to note that these rankings need to be validated by statistical analysis. Our MANOVA analysis showed significant results between the candidates. The Tukey method for pairwise comparison and family-wise error correction indicated that while upvoting methods achieved significant results over other candidates, there was no statistical difference between upvoting methods using either SMOTE or R-oversampling.

Table 3.2 Tukey results over all classification metrics between selected representation methods.

Comparison	**Mean GP1**	**Mean GP2**	**Metrics**	**Reject Null**
Upvote_SM vs. Upvote_RO	0.9712	0.9682	F1	FALSE
Upvote_SM vs. Upvote_RO	0.0187	0.0258	FDR	FALSE
Upvote_SM vs. Upvote_RO	0.9614	0.9622	TPR	FALSE
Upvote_SM vs. Upvote_RO	0.9813	0.9742	Precision	FALSE
Upvote_SM vs. Upvote_RO	0.9972	0.9972	NPV	FALSE
Upvote_SM vs. Upvote_RO	0.0019	0.0013	FPR	FALSE
Upvote_SM vs. UniRep_SM	0.9712	0.9307	F1	TRUE
Upvote_SM vs. UniRep_SM	0.0187	0.0930	FDR	TRUE
Upvote_SM vs. UniRep_SM	0.9614	0.9557	TPR	TRUE
Upvote_SM vs. UniRep_SM	0.9813	0.9070	Precision	TRUE
Upvote_SM vs. UniRep_SM	0.9972	0.9967	NPV	TRUE
Upvote_SM vs. UniRep_SM	0.0019	0.0072	FPR	TRUE
Upvote_RO vs. Concat_RO	0.9682	0.9261	F1	TRUE
Upvote_RO vs. Concat_RO	0.0258	0.1061	FDR	TRUE
Upvote_RO vs. Concat_RO	0.9622	0.9607	TPR	FALSE
Upvote_RO vs. Concat_RO	0.9742	0.8939	Precision	TRUE
Upvote_RO vs. Concat_RO	0.9972	0.9971	NPV	FALSE
Upvote_RO vs. Concat_RO	0.0013	0.0084	FPR	TRUE
Concat_RO vs. UniRep_SM	0.9261	0.9307	F1	TRUE
Concat_RO vs. UniRep_SM	0.1061	0.0930	FDR	TRUE
Concat_RO vs. UniRep_SM	0.9607	0.9557	TPR	TRUE
Concat_RO vs. UniRep_SM	0.8939	0.9070	Precision	TRUE
Concat_RO vs. UniRep_SM	0.9971	0.9967	NPV	TRUE
Concat_RO vs. UniRep_SM	0.0084	0.0072	FPR	TRUE



B. Ranking Results

Subjective Weighting (Positive-Class Priority)

Alternatives		Score	Ranking
R-Oversampling	One-Hot	0.2878	8
	UniRep	0.3112	7
	ESM	0.0227	10
	Concat	0.3730	5
	Upvote	0.9524	**2**
SMOTE	One-Hot	0.4342	4
	UniRep	0.4725	**3**
	ESM	0.1931	9
	Concat	0.3688	6
	Upvote	0.9926	**1**

Objective Weighting (Entropy)

Alternatives		Score	Ranking
R-Oversampling	One-Hot	0.3496	7
	UniRep	0.3142	8
	ESM	0.1398	10
	Concat	0.4506	**3**
	Upvote	0.9563	**1**
SMOTE	One-Hot	0.4214	5
	UniRep	0.4044	6
	ESM	0.1638	9
	Concat	0.4498	4
	Upvote	0.9523	**2**

Figure 3.6 Upvoting achieved the best ranking both in subjective and objective weighting in MCDA design. A. The main steps for performing MCDA are elaborated. This includes detailing the selected methods for implementing MCDA such as classification criteria, model selection, and statistical analysis. B. TOPSIS scores, i.e., closeness coefficients, and their associated rankings are shown for subjective and objective weighting, demonstrating the effectiveness of different encoding methods.

A list of pairwise comparisons over all alternatives can be found in the supplementary information.

The voting method enhanced the prediction score using both a single metric and multiple metrics in MCDA by combining the predictions of multiple models based on single encodings. We concluded that as different encodings might capture the distance and relationships of the data points differently, combining their predictions boosted the final model performance. The encoding methods used for the voting technique in the dataset are visualized in Figure 3.7 in a uniform manifold approximation and projection (UMAP) plot.

3.3.6 How Protein Encodings Perform Considering Different Data Attributes

The hypotheses were tested over affibody datasets that had notable attributes such as severe imbalance, multiple mutation sites, affinity and specificity enrichment, and small molecular protein length. The obtained results indicated voting and oversampling were highly effective methods to boost fitness prediction performance. However, individual protein-encoding performance comparisons need a more convincing explanation and thorough exploration. Specifically, we

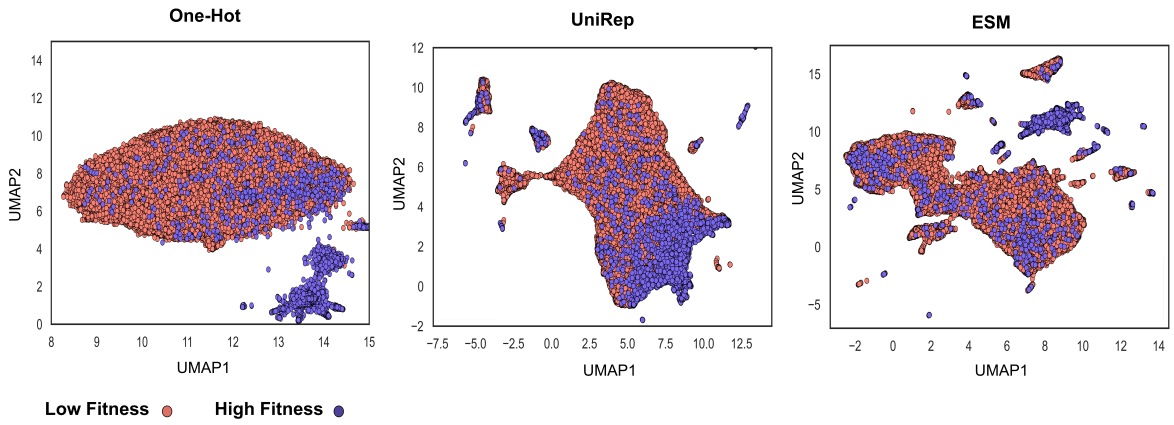


Figure 3.7 Different protein encodings potentially capture distinct functional aspects of the proteins. A 2D visualization of the encoding techniques that resulted in improved prediction in the voting method in UMAP. This method is a dimensionality reduction technique such as principal component analysis (PCA) [65] with unique advantages such as preserving the local structure of the data and capturing non-linear relationships between data points. In observing the sequence–function relationship in proteins, one can conclude that each protein sequence representation/encoding has the potential to capture different aspects of fitness.

wondered why ESM underperformed One-Hot and UniRep despite the more powerful setup in pretraining and being showcased in studies for high prediction potential [66]. While the performance could be due to the datatype (e.g., small protein, complex fitness, etc.), we decided to further analyze the encoding prediction scores in a completely different dataset and bring insights on embedding performances in various conditions (e.g., data size in training, protein length, prediction task difficulty). The curated data contains 18,190 sequences with varying amino acid (aa) lengths and provides melting points that indicate protein stability. Figure 8 is the performance comparison in the stability prediction of embeddings, their concatenation, and voting using different data sizes. Despite underperforming in the affibody affinity data, ESM performed best for stability prediction when including proteins with max aa length = 500.

We further evaluated the performance of embedding methods in large proteins $400 \leq$ aa length $\leq$

1500 and small proteins aa length $\leq$ 120 to check if ESM still outperforms the other representations in stability prediction. A complete list of statistical analysis is attached in the supplementary materials. The analysis of physical feature encoding is provided in Figure 3A.9.

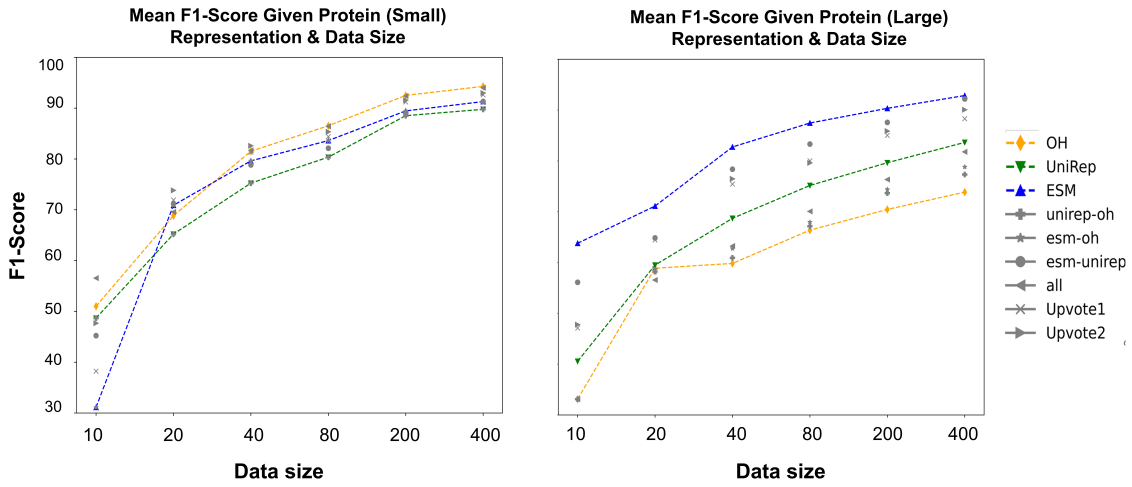


Figure 3.8 The effect of protein size on the performance of encoding methods in stability prediction while data sizes vary. The obtained results are largely different with respect to the protein size—small proteins (aa length $\leq$ 120) vs. large (400 $\leq$ aa length $\leq$ 1500). Highlights: For small proteins, upon comparing the violin plots and statistical test results, protein sequence encoding methods performed distinctively with respect to the initial dataset (protein max length = 500). One-Hot encoding had a more significant contribution in boosting the classification metrics for small proteins. As an example, when n = 400, both One-Hot and All-Encoding concatenation with a mean F1-score of 94% outperformed the other encoding methods. One-Hot tends to be problematic for large proteins as it results in a highly sparse encoding vector. This was shown in this plot when One-Hot encoding performance was not satisfactory in comparison with ESM and UniRep. When n = 400, based on both the violin plots and the post-hoc analysis after ANOVA (both Bonferroni and Tukey), either ESM or ESM_UniRep with 92% mean F1-score achieved the highest performance. One-Hot with 73% mean F1-score was the lowest score among all the encodings. Refer to the supplementary information for all one-by-one comparisons of the statistics and classification.

3.3.7 Regression Analysis for Melting Point Prediction

The last analysis is a regression task for predicting the melting point value. We wondered how different encoding methods performed if we used all the data and increased the prediction challenge

(Tm prediction rather than stability class prediction). MSE and R^2 are shown predicting the Tm values of a dataset of 18,190 sequences with 0.3 test size. There was a significant difference in the performance of encoding methods, which was not the case in the classification task. ESM was the best encoding method in predicting stability (R^2 score= 0.65). Note that we used all the data (i.e., did not use sampling) for training our regression model. The regression metrics are reported in Table 3.

Table 3.3 Regression metrics for encoding methods in validation and test.

Encoding	**Validation**		**Test**	
	R^2	**MSE**	**R^2**	**MSE**
One-Hot	0.21	141	0.24	130
UniRep	0.49	108	0.40	102
ESM	0.65	63	0.65	60

3.4 Discussion

In this study, we shed light on two key challenges of applying discriminative models over amino acid sequence data for protein engineering applications: (1) handling imbalanced data and (2) choosing an appropriate protein representation (i.e., encoding). Assay-labeled sequence data in this domain is often severely imbalanced (due to the rugged and sparse nature of the protein fitness landscape) and requires careful consideration in data sampling, splitting, and choice in data representation for model training. To capture this common occurrence of imbalanced data, we trained discriminative ML models over our cytometry-sorted deep-sequenced small protein (affibody) data to distinguish between functional sequences (n = 6077) among a large collection of non-functional protein sequences (n = 82,663). We then explored the impact of encoding protein sequences using two simplistic approaches (One-Hot encoding, physiochemical encoding) and two language-based methods (UniRep, ESM). We hypothesized that as each protein representation may capture distinct information, combining representations via embedding concatenation and ensemble learning increases overall performance and generalizability.

To address the issue of imbalanced data, we implemented various sampling techniques, including undersampling, random oversampling, and SMOTE, and evaluated their performance using multiple

classification metrics. Our results indicate that implementing oversampling techniques over imbalanced datasets improves predictive performance relative to undersampling or the exclusion of sampling methods. Among the sequence representation methods, embeddings are the answer to improved fitness prediction and data requirements. However, it is essential to consider the choice of protein representation, its benefits, and its drawbacks. For example, the choice of fitness to be predicted (e.g., thermal stability, binding affinity, target specificity) and the language model pretraining procedure affect the model's predictive performance and need further discussion. Therefore, we analyzed an additional dataset (i.e., the NESP dataset, which included a variety of protein sequences with their Tm) to discuss the effect of protein representations over variables such as protein length, protein fitness, and prediction type (i.e., classification vs. regression). For ensemble learning, we used majority voting to combine the prediction of each representation over the same ML model, which significantly improved the prediction score, and its obtained results were statistically significant using MANOVA and the post-hoc method (Figure 3.6, Table 3.2).

3.5 Conclusions

This study intends to inform protein engineers that: (i) embeddings derived from self-supervised representation techniques are not always the optimal route to take, depending on the protein size and protein fitness to be predicted; (ii) oversampling techniques, especially SMOTE, have the ability to overcome the notorious challenge of highly imbalanced data in the protein fitness landscape; (iii) different aspects learned in each protein encoding can be combined by voting techniques and result in better predictive scores. These conclusions were revealed in the context of integrating machine learning and protein engineering knowledge to identify high-fitness protein sequences. Specifically, we quantified model performance while varying the choice of feature representation, ensemble learning, and sampling methods. Analysis across a broad range of protein chain lengths revealed the ESM language model to be most beneficial for encoding large protein sequences (Figure 3.8). However, in the context of small protein sequences, a comparable performance was observed between One-Hot encoding and the language models (ESM and UniRep). In our analysis, oversampling proved to be an effective technique to improve performance when dealing with

severely imbalanced datasets (Figure 3.4). Finally, ensemble learning was a promising method for boosting the binding prediction scores when using unique, competitive encoding methods (Figure 3.5, Figure 3.6).

BIBLIOGRAPHY

[1] W. Liebermeister, E. Noor, A. Flamholz, D. Davidi, J. Bernhardt, and R. Milo. Visual account of protein investment in cellular functions. *Proceedings of the National Academy of Sciences of the United States of America*, 111:8488–8493, 2014.

[2] J. Schlessinger. Cell signaling by receptor tyrosine kinases. *Cell*, 103:211–225, 2000.

[3] B.L. Hogan. Bone morphogenetic proteins: Multifunctional regulators of vertebrate development. *Genes & Development*, 10:1580–1594, 1996.

[4] E. Andrianantoandro, S. Basu, D.K. Karig, and R. Weiss. Synthetic biology: New engineering rules for an emerging discipline. *Molecular Systems Biology*, 2:2006.0028–2006.0028, 2006.

[5] M. Heim, L. Römer, and T. Scheibel. Hierarchical structures made of proteins. the complex architecture of spider webs and their constituent silk proteins. *Chemical Society Reviews*, 39:156–164, 2010.

[6] H. Kolmar. Biological diversity and therapeutic potential of natural and engineered cystine knot miniproteins. *Current Opinion in Pharmacology*, 9:608–614, 2009.

[7] A. Krasniqi, M. D'Huyvetter, N. Devoogdt, F.Y. Frejd, J. Sörensen, A. Orlova, M. Keyaerts, and V. Tolmachev. Same-day imaging using small proteins: Clinical experience and translational prospects in oncology. *Journal of Nuclear Medicine*, 59:885–891, 2018.

[8] P.A. Romero and F.H. Arnold. Exploring protein fitness landscapes by directed evolution. *Nature Reviews Molecular Cell Biology*, 10:866–876, 2009.

[9] H.W. Hellinga. Rational protein design: Combining theory and experiment. *Proceedings of the National Academy of Sciences of the United States of America*, 94:10015–10017, 1997.

[10] C. Jäckel, P. Kast, and D. Hilvert. Protein design by directed evolution. *Annual Review of Biophysics*, 37:153–173, 2008.

[11] G. Li, Y. Dong, and M.T. Reetz. Can machine learning revolutionize directed evolution of selective enzymes? *Advanced Synthesis & Catalysis*, 361:2377–2386, 2019.

[12] N. Anand, R. Eguchi, I.I. Mathews, C.P. Perez, A. Derry, R.B. Altman, and P.S. Huang. Protein sequence design with a learned potential. *Nature Communications*, 13:716, 2022.

[13] Z. Wu, S.B.J. Kan, R.D. Lewis, B.J. Wittmann, and F.H. Arnold. Machine learning-assisted directed protein evolution with combinatorial libraries. *Proceedings of the National Academy of Sciences of the United States of America*, 116:8852–8858, 2019.

[14] Y. Saito, M. Oikawa, T. Sato, H. Nakazawa, T. Ito, T. Kameda, K. Tsuda, and M. Umetsu.

Machine-learning-guided library design cycle for directed evolution of enzymes: The effects of training data composition on sequence space exploration. *ACS Catalysis*, 11:14615–14624, 2021.

[15] A.W. Golinski, K.M. Mischler, S. Laxminarayan, N.L. Neurock, M. Fossing, H. Pichman, S. Martiniani, and B.J. Hackel. High-throughput developability assays enable library-scale identification of producible protein scaffold variants. *Proceedings of the National Academy of Sciences of the United States of America*, 118:e2026658118, 2021.

[16] J. Chen, S. Zheng, H. Zhao, and Y. Yang. Structure-aware protein solubility prediction from sequence through graph convolutional network and predicted contact map. *Journal of Cheminformatics*, 13:1–10, 2021.

[17] S. Wang, D. Liu, M. Ding, Z. Du, Y. Zhong, T. Song, J. Zhu, and R. Zhao. Se-onionnet: A convolution neural network for protein–ligand binding affinity prediction. *Frontiers in Genetics*, 11:607824, 2021.

[18] K. Kuzmin, A.E. Adeniyi, A.K. DaSouza, D. Lim, H. Nguyen, N.R. Molina, L. Xiong, I.T. Weber, and R.W. Harrison. Machine learning methods accurately predict host specificity of coronaviruses based on spike sequences alone. *Biochemical and Biophysical Research Communications*, 533:553–558, 2020.

[19] S. Das and S. Chakrabarti. Classification and prediction of protein–protein interaction interface using machine learning algorithm. *Scientific Reports*, 11:1–12, 2021.

[20] Y. Vander Meersche, G. Cretin, A.G. de Brevern, J.C. Gelly, and T. Galochkina. Medusa: Prediction of protein flexibility from sequence. *Journal of Molecular Biology*, 433:166882, 2021.

[21] T. Young, D. Hazarika, S. Poria, and E. Cambria. Recent trends in deep learning based natural language processing [review article]. *IEEE Computational Intelligence Magazine*, 13:55–75, 2018.

[22] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

[23] M. Mnasri. Recent advances in conversational nlp: Towards the standardization of chatbot building. *arXiv preprint*, 2019.

[24] G. Campagna, S. Xu, M. Moradshahi, R. Socher, and M.S. Lam. Genie: A generator of natural language semantic parsers for virtual assistant commands. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 394–410, 2019.

[25] M. Heinzinger, A. Elnaggar, Y. Wang, C. Dallago, D. Nechaev, F. Matthes, and B. Rost. Modeling aspects of the language of life through transfer-learning protein sequences. *BMC Bioinformatics*, 20:1–17, 2019.

[26] D. Ofer, N. Brandes, and M. Linial. The language of proteins: Nlp, machine learning & protein sequences. *Computational and Structural Biotechnology Journal*, 19:1750–1758, 2021.

[27] A. Bateman, M.J. Martin, C. O'Donovan, M. Magrane, R. Apweiler, E. Alpi, R. Antunes, J. Arganiska, B. Bely, and M. Bingley. Uniprot: A hub for protein information. *Nucleic Acids Research*, 43:D204–D212, 2015.

[28] K. Katz, O. Shutov, R. Lapoint, M. Kimelman, J.R. Brister, and C. O'Sullivan. The sequence read archive: A decade more of explosive growth. *Nucleic Acids Research*, 50:D387–D390, 2022.

[29] M. Torrisi, G. Pollastri, and Q. Le. Deep learning methods in protein structure prediction. *Computational and Structural Biotechnology Journal*, 18:1301–1310, 2020.

[30] A. Elnaggar, M. Heinzinger, C. Dallago, G. Rehawi, W. Yu, L. Jones, T. Gibbs, T. Feher, C. Angerer, and M. Steinegger. Prottrans: Towards cracking the language of life's code through self-supervised deep learning and high performance computing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:7112–7127, 2021.

[31] N. Ferruz, S. Schmidt, and B. Höcker. Protgpt2 is a deep unsupervised language model for protein design. *Nature Communications*, 13:4348, 2022.

[32] E.C. Alley, G. Khimulya, S. Biswas, M. AlQuraishi, and G.M. Church. Unified rational protein engineering with sequence-based deep representation learning. *Nature Methods*, 16:1315–1322, 2019.

[33] A. Rives, J. Meier, T. Sercu, S. Goyal, Z. Lin, J. Liu, D. Guo, M. Ott, C.L. Zitnick, and J. Ma. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences of the United States of America*, 118:e2016239118, 2021.

[34] R. Rao, N. Bhattacharya, N. Thomas, Y. Duan, X. Chen, J. Canny, P. Abbeel, and Y.S. Song. Evaluating protein transfer learning with tape. In *Advances in Neural Information Processing Systems*, volume 32, pages 9689–9699, 2019.

[35] N. Brandes, D. Ofer, Y. Peleg, N. Rappoport, and M. Linial. Proteinbert: A universal deep-learning model of protein sequence and function. *Bioinformatics*, 38:2102–2110, 2022.

[36] C. Hsu, H. Nisonoff, C. Fannjiang, and J. Listgarten. Combining evolutionary and assay-labelled data for protein fitness prediction. *bioRxiv*, 2021.

[37] J. Meier, R. Rao, R. Verkuil, J. Liu, T. Sercu, and A. Rives. Language models enable zero-shot prediction of the effects of mutations on protein function. *Advances in Neural Information Processing Systems*, 35:29287–29303, 2021.

[38] S.K.S. Chu and J. Siegel. Predicting single-point mutational effect on protein stability. *Growth*, 16:35–45, 2021.

[39] Z. Lv, P. Wang, Q. Zou, and Q. Jiang. Identification of sub-golgi protein localization by use of deep representation learning features. *Bioinformatics*, 36:5600–5609, 2020.

[40] K. Li, Y. Zhong, X. Lin, and Z. Quan. Predicting the disease risk of protein mutation sequences with pre-training model. *Frontiers in Genetics*, 11:1–10, 2020.

[41] S. Min, H.G. Kim, B. Lee, and S. Yoon. Protein transfer learning improves identification of heat shock protein families. *PLOS ONE*, 16:e0251865, 2021.

[42] D.R. Woldring, P.V. Holec, L.A. Stern, Y. Du, and B.J. Hackel. A gradient of sitewise diversity promotes evolutionary fitness for binder discovery in a three-helix bundle protein scaffold. *Biochemistry*, 56:1656–1671, 2017.

[43] D. Pultz, E. Friis, Inversion, J. Salomon, Maggie, P. Fischer Hallin, and S. Baagøe Jørgensen. Novozymes enzyme stability prediction, 2022. Kaggle.

[44] R.L. Keeney, H. Raiffa, and D.W. Rajala. *Decisions with Multiple Objectives: Preferences and Value Trade-Offs*, volume 9. IEEE Transactions on Systems, Man, and Cybernetics, 1977.

[45] D.R. Woldring, P.V. Holec, L.A. Stern, Y. Du, and B.J. Hackel. A gradient of sitewise diversity promotes evolutionary fitness for binder discovery in a three-helix bundle protein scaffold. *Biochemistry*, 56:1656–1671, 2017.

[46] A.T. Müller, G. Gabernet, J.A. Hiss, and G. Schneider. Modlamp: Python for antimicrobial peptides. *Bioinformatics*, 33:2753–2755, 2017.

[47] K.K. Yang, Z. Wu, C.N. Bedbrook, and F.H. Arnold. Learned protein embeddings for machine learning. *Bioinformatics*, 34:2642–2648, 2018.

[48] Z. Lin, H. Akin, R. Rao, B. Hie, Z. Zhu, W. Lu, N. Smetanin, R. Verkuil, O. Kabeli, and Y. Shmueli. Evolutionary-scale prediction of atomic level protein structure with a language model. *Science*, 379:1123–1130, 2023.

[49] B. Kovács, F. Tinya, C. Németh, and P. Ódor. Smote: Synthetic minority over-sampling technique nitesh. *Ecological Applications*, 30:321–357, 2020.

[50] A. Fernández, S. García, F. Herrera, and N.V. Chawla. Smote for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary. *Journal of Artificial*

Intelligence Research, 61:863–905, 2018.

[51] A.J. Mohammed. Improving classification performance for a novel imbalanced medical dataset using smote method. *International Journal of Advanced Trends in Computer Science and Engineering*, 9:3161–3172, 2020.

[52] V. Rupapara, F. Rustam, H.F. Shahzad, A. Mehmood, I. Ashraf, and G.S. Choi. Impact of smote on imbalanced text features for toxic comments classification using rvvc model. *IEEE Access*, 9:78621–78634, 2021.

[53] T. Hasanin, T.M. Khoshgoftaar, J.L. Leevy, and R.A. Bauder. Severely imbalanced big data challenges: Investigating data sampling approaches. *Journal of Big Data*, 6:1–25, 2019.

[54] R. Blagus and L. Lusa. Evaluation of smote for high-dimensional class-imbalanced microarray data. In *Proceedings of the 11th International Conference on Machine Learning and Applications (ICMLA 2012)*, volume 2, pages 89–94, 2012.

[55] R. van den Goorbergh, M. van Smeden, D. Timmerman, and B. Van Calster. The harm of class imbalance corrections for risk prediction models: Illustration and simulation using logistic regression. *Journal of the American Medical Informatics Association*, 29:1525–1534, 2022.

[56] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2623–2631, 2019.

[57] M.L. McHugh. Multiple comparison analysis testing in anova. *Biochemia Medica*, 21:203–209, 2011.

[58] R.A. Armstrong. When to use the bonferroni correction. *Ophthalmic & Physiological Optics*, 34:502–508, 2014.

[59] J. Tukey. The problem of multiple comparisons. Unpublished manuscript, Department of Statistics, Princeton University, Princeton, NJ.

[60] P. Branco, L. Torgo, and R. Ribeiro. A survey of predictive modelling under imbalanced distributions. *ACM Comput. Surv. (CSUR)*, 49:1–50, 2015.

[61] K. Borowska and J. Stepaniuk. Imbalanced data classification: A novel re-sampling approach combining versatile improved smote and rough sets. In *Proceedings of the Computer Information Systems and Industrial Management: 15th IFIP TC8 International Conference, CISIM 2016*, volume 9842, pages 31–42, Vilnius, Lithuania, 2016. Springer International Publishing.

[62] C.-L. Hwang and K. Yoon. *Multiple Attribute Decision Making: Methods and Applications A State-of-the-Art Survey*. Springer, Berlin, Heidelberg, Germany, 1981.

[63] C.E. Shannon. A mathematical theory of communication. *Bell Syst. Tech. J.*, 27:379–423, 1948.

[64] R.A. Johnson and D.W. Wichern. *Applied Multivariate Statistical Analysis*. Pearson Prentice Hall, NJ USA, 2007.

[65] I.T. Jolliffe. Principal component analysis: A beginner's guide—i. introduction and application. *Weather*, 45:375–382, 1990.

[66] Z. Lin, H. Akin, R. Rao, B. Hie, Z. Zhu, W. Lu, A.d.S. Costa, M. Fazel-Zarandi, T. Sercu, S. Candido, et al. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *bioRxiv*, 2022.

APPENDIX 3A

SUPPLEMENTARY INFORMATION

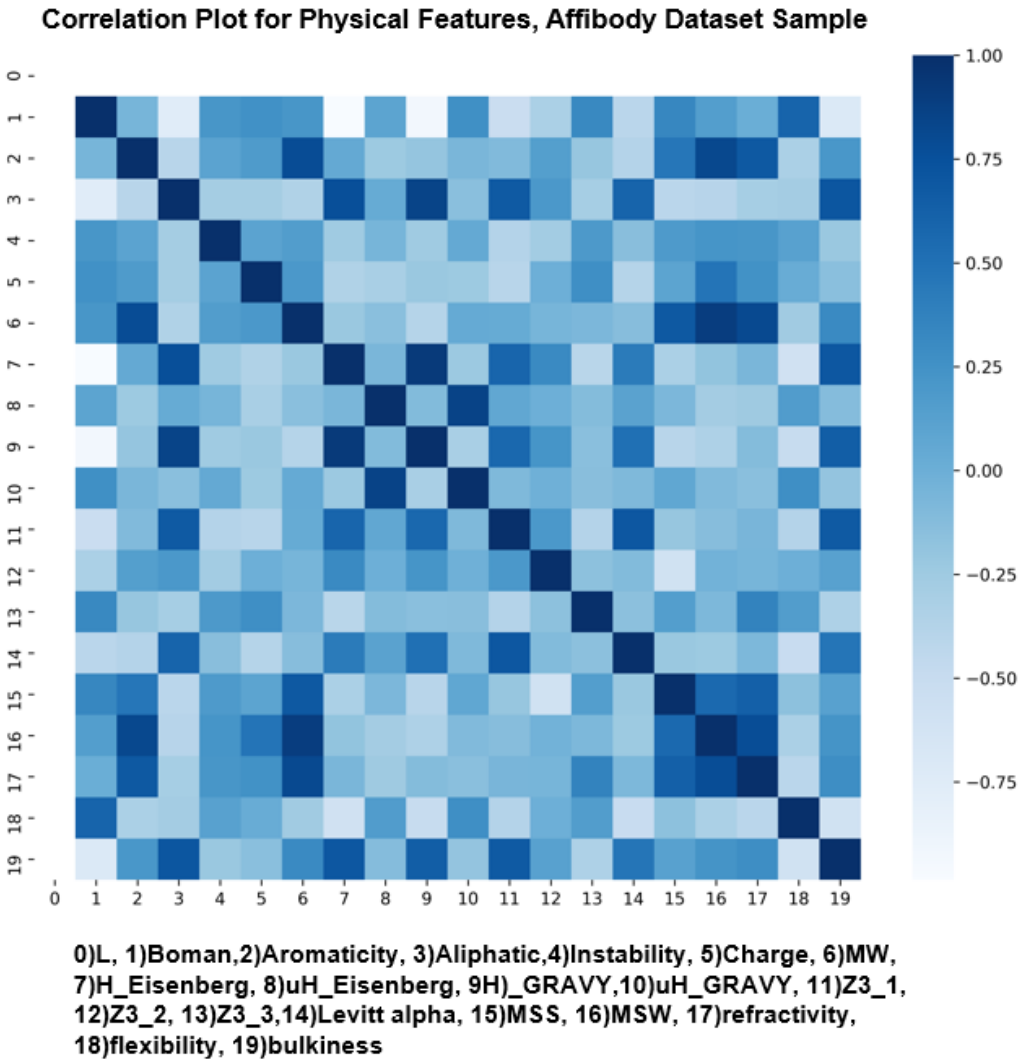


Figure 3A.1 The majority of physical features are at least correlated with one other physical feature; leaving us insights into attribute dependencies in the affibody dataset. The plot is drawn upon sampling from both naïve and enriched populations. Bold colors represent a higher correlation of features. As the affibody length in our dataset was fixed, the first row and column of the matrix are empty. Insights on the given correlation plot include i) a high correlation of Boman index and protein flexibility index, ii) a high correlation of aromaticity with molecular weight, MSW, and refractivity, iii) Aliphatic index correlating with multiple features such as bulkiness, H-Eisenberg, and H-Gravy.

	Comparison	P-Value
Under-sampling	Physical vs. **OH**	**2.37E-27**
	Physical vs. **UniRep**	**2.60E-24**
	Physical vs. **ESM**	**2.22E-20**
	OH vs. **UniRep**	**6.86E-06**
	OH vs. ESM	**9.31E-13**
	UniRep vs. ESM	**6.44E-08**
R-oversampling	Physical vs. **OH**	**6.12E-52**
	Physical vs**. UniRep**	**3.20E-50**
	Physical vs. **ESM**	**5.11E-50**
	OH vs. UniRep	0.619
	OH vs. ESM	**1.06E-18**
	UniRep vs. ESM	**9.01E-17**
SMOTE	Physical vs. **OH**	**2.06E-51**
	Physical vs. **UniRep**	**2.87E-50**
	Physical vs. **ESM**	**1.15E-50**
	OH vs. UniRep	0.223
	OH vs. ESM	**9.29E-15**
	UniRep vs. ESM	**9.01E-17**

Figure 3A.2 The initial ANOVA test was done for multiple comparisons, and it resulted in p-value $= 9.53 \times 10^{-190}$. The following tables in the supplement show the t-test results when Bonferroni correction is considered. The statistically significant results are **bolded**.

Comparisons R-Oversampling for i vs. j	Mean i	Mean j	P-value
OH-vs-UniRep	91.92	92	0.619375
OH-vs-ESM	91.92	89.91	**1.06E-18**
OH-vs-UniRep+OH	91.92	92.54	**0.000201**
OH-vs-ESM+OH	91.92	91.45	0.026281
OH-vs-ESM+UniRep	91.92	90.56	**1.77E-09**
OH-vs-All	91.92	92.61	**8.38E-05**
OH-vs-Upvoted	91.92	96.81	**2.97E-31**
UniRep-vs-ESM	92	89.91	**2.04E-16**
UniRep-vs-UniRep+OH	92	92.54	0.002124
UniRep-vs-ESM+OH	92	91.45	0.015463
UniRep-vs-ESM+UniRep	92	90.56	**1.57E-09**
UniRep-vs-All	92	92.61	**0.000896**
UniRep-vs-Upvoted	92	96.81	**8.12E-26**
ESM-vs-UniRep+OH	89.91	92.54	**4.29E-19**
ESM-vs-ESM+OH	89.91	91.45	**2.15E-08**
ESM-vs-ESM+UniRep	89.91	90.56	**0.000344**
ESM-vs-All	89.91	92.61	**9.44E-19**
ESM-vs-Upvoted	89.91	96.81	**2.52E-38**
UniRep+OH-vs-ESM+OH	92.54	91.45	**1.47E-05**
UniRep+OH-vs-ESM+UniRep	92.54	90.56	**4.37E-13**
UniRep+OH-vs-All	92.54	92.61	0.706828
UniRep+OH-vs-Upvoted	92.54	96.81	**8.42E-24**
ESM+OH-vs-ESM+UniRep	91.45	90.56	**0.000355**
ESM+OH-vs-All	91.45	92.61	**7.00E-06**
ESM+OH-vs-Upvoted	91.45	96.81	**1.35E-19**
ESM+UniRep-vs-All	90.56	92.61	**2.59E-13**
ESM+UniRep-vs-Upvoted	90.56	96.81	**2.84E-25**
All-vs-Upvoted	92.61	96.81	**8.43E-23**

Figure 3A.3 Figure 5 T-test Results-part1.

Comparisons for SMOTE for i vs. j	Mean i	Mean j	P-value
OH-vs-UniRep	92.88	93.07	0.222968
OH-vs-ESM	92.88	91.09	**9.29E-15**
OH-vs-UniRep+OH	92.88	92.53	0.050908
OH-vs-ESM+OH	92.88	91.61	**5.80E-11**
OH-vs-ESM+UniRep	92.88	90.47	**1.85E-15**
OH-vs-All	92.88	92.58	0.074292
OH-vs-Upvoted	92.88	97.08	**4.02E-24**
UniRep-vs-ESM	93.07	91.09	**9.01E-17**
UniRep-vs-UniRep+OH	93.07	92.53	0.003022
UniRep-vs-ESM+OH	93.07	91.61	**3.23E-13**
UniRep-vs-ESM+UniRep	93.07	90.47	**1.72E-16**
UniRep-vs-All	93.07	92.58	0.004057
UniRep-vs-Upvoted	93.07	97.08	**6.46E-25**
ESM-vs-UniRep+OH	91.09	92.53	**2.43E-10**
ESM-vs-ESM+OH	91.09	91.61	**0.000153**
ESM-vs-ESM+UniRep	91.09	90.47	0.000995
ESM-vs-All	91.09	92.58	**1.50E-11**
ESM-vs-Upvoted	91.09	97.08	**1.88E-32**
UniRep+OH-vs-ESM+OH	92.53	91.61	**1.56E-06**
UniRep+OH-vs-ESM+UniRep	92.53	90.47	**8.55E-13**
UniRep+OH-vs-All	92.53	92.58	0.792126
UniRep+OH-vs-Upvoted	92.53	97.08	**1.11E-21**
ESM+OH-vs-ESM+UniRep	91.61	90.47	**1.11E-07**
ESM+OH-vs-All	91.61	92.58	**1.50E-07**
ESM+OH-vs-Upvoted	91.61	97.08	**2.07E-34**
ESM+UniRep-vs-All	90.47	92.58	**1.74E-13**
ESM+UniRep-vs-Upvoted	90.47	97.08	**1.37E-24**
All-vs-Upvoted	92.58	97.08	**4.06E-23**

Figure 3A.4 Figure 5 T-test Results-part2.

Comparison for Samplings R-Oversampling vs. SMOTE	Mean R-Oversampling	Mean SMOTE	P-Value
OH	91.92	92.88	**7.30e-08**
UniRep	92	93.07	**3.08e-08**
ESM	89.91	91.09	**1.90e-11**
UniRep+OH	92.54	92.53	0.96
ESM+OH	91.45	91.61	0.44
ESM+UniRep	90.56	90.47	0.65
All	92.61	92.58	0.88
Upvoted	96.81	97.08	0.002

Figure 3A.5 SMOTE either Improved the performance or had no hampering effect with respect to R-Oversampling.

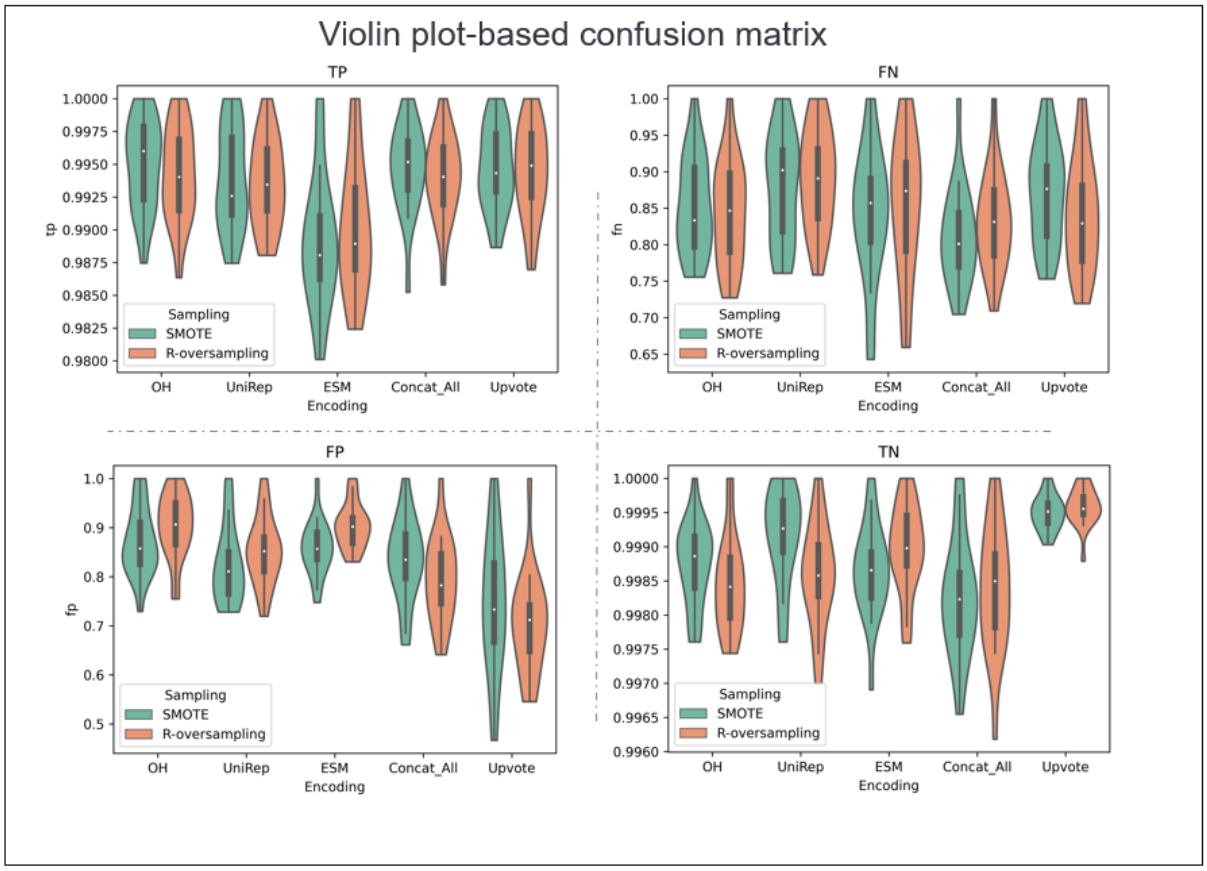


Figure 3A.6 Violin Plot-Based Confusion Matrix.Individual Encodings Perform uniquely in the confusion matrix entities. While the overall predictive performance is the main goal and it is represented via F1-Score throughout the literature, inspecting how each model performs for maximizing true positives(TP) and true negatives(TN) while minimizing false positives(FP) and false negatives(FN), provides insights about each model performance.

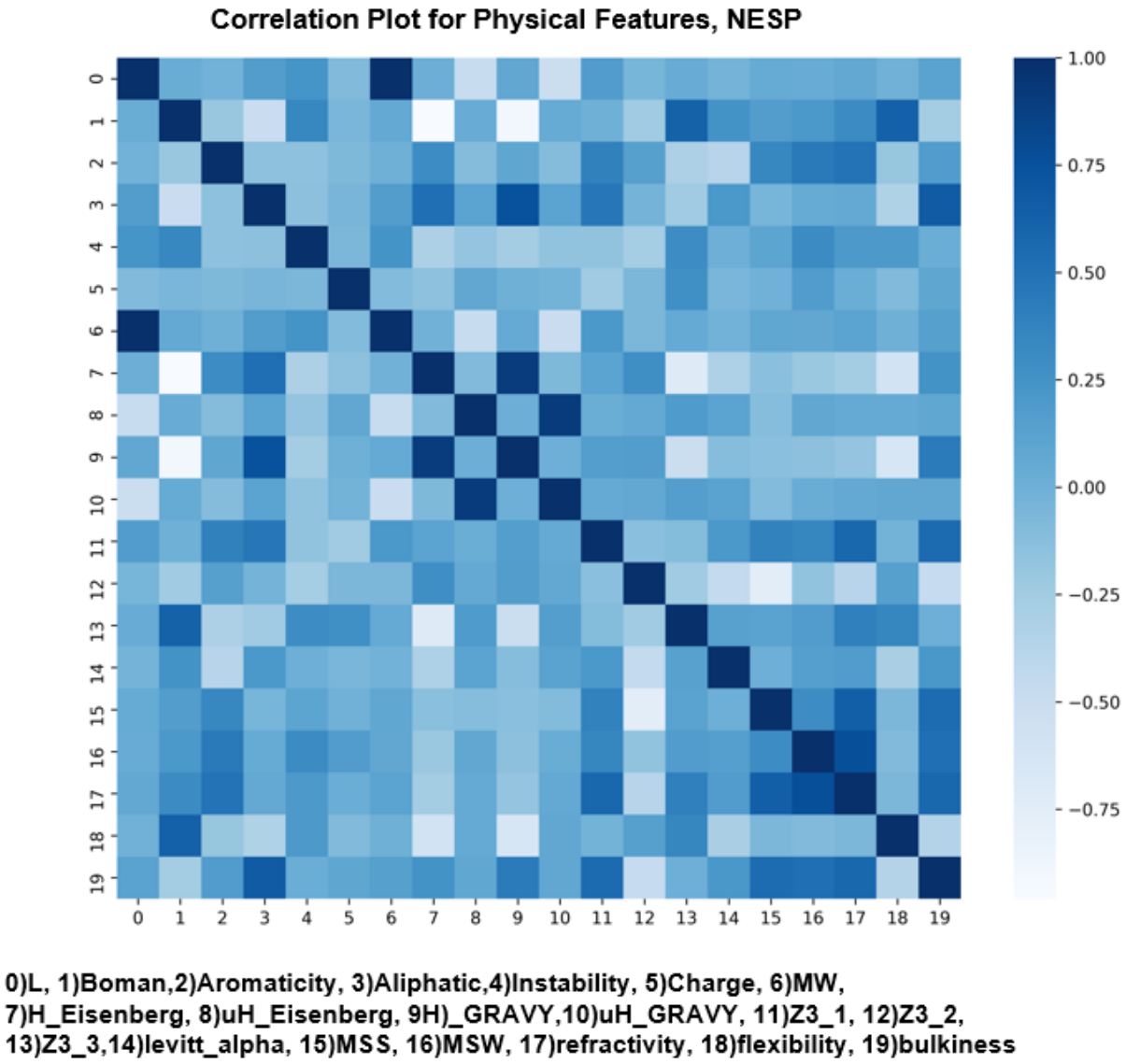


Figure 3A.7 Physical feature correlation plot for NESP dataset.

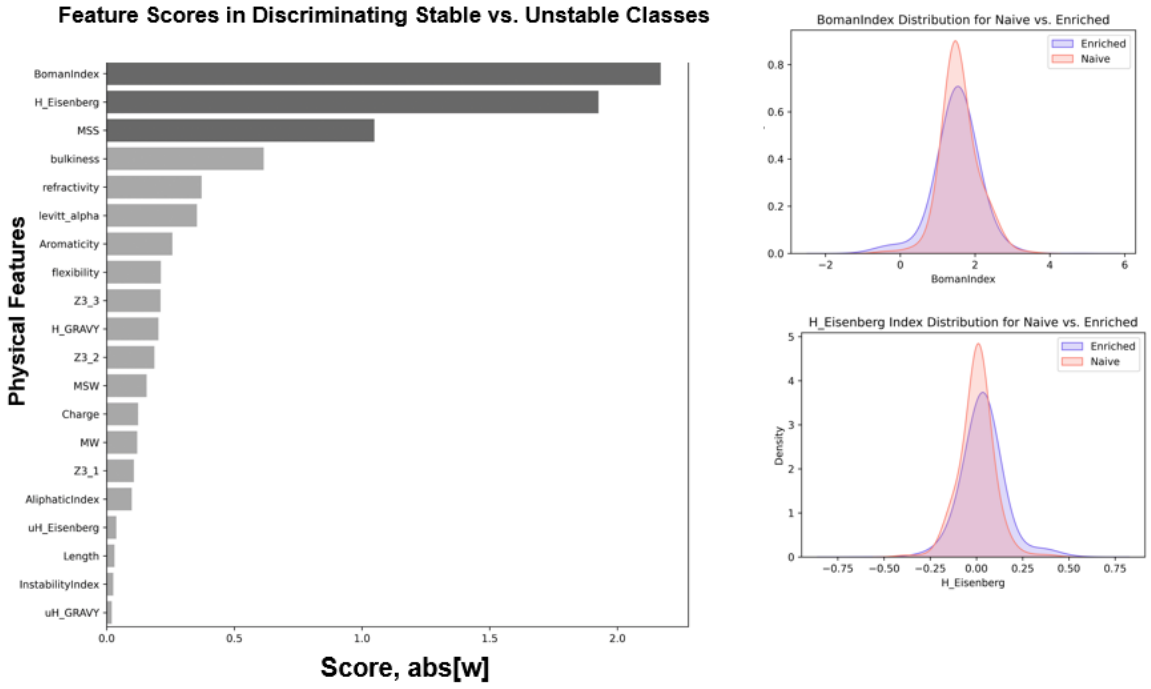


Figure 3A.8 Physical feature ranking for NESP dataset. The feature ranking is presented after using all the data for the classification of stable vs. unstable sequences. Boman Index, H-Eisenberg, and MSS are the lead features. However, their scores are not significantly higher than the other physical attributes, indicating that more features are incorporated in the final F1 Score=0.86.

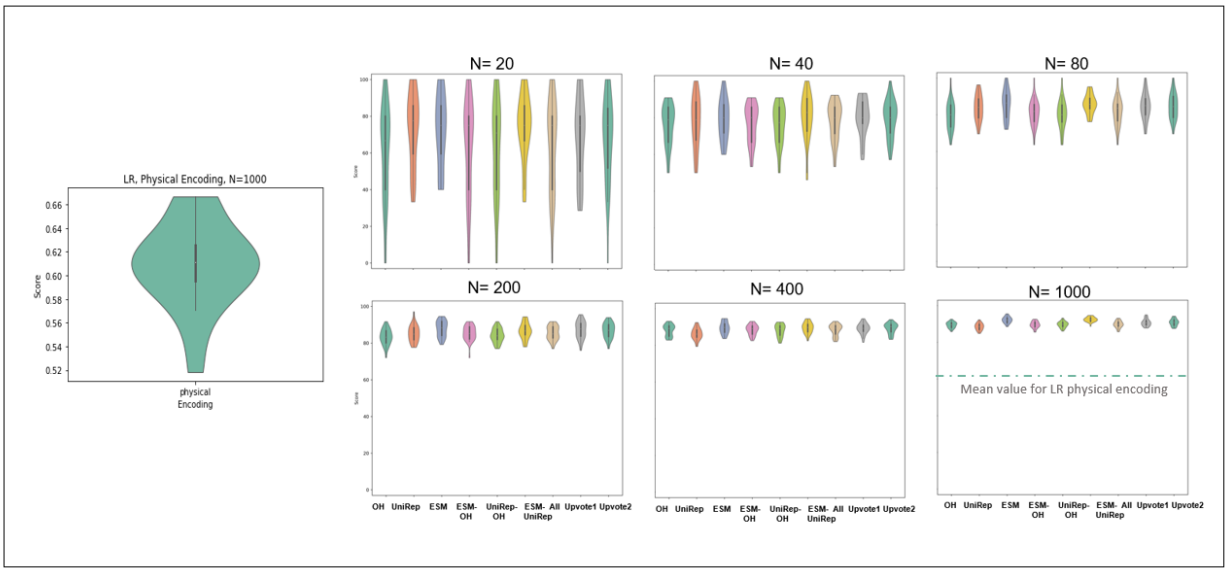


Figure 3A.9 Physical Feature representation while using maximum N=1000, performed poorly and have not got selected for the main figure.

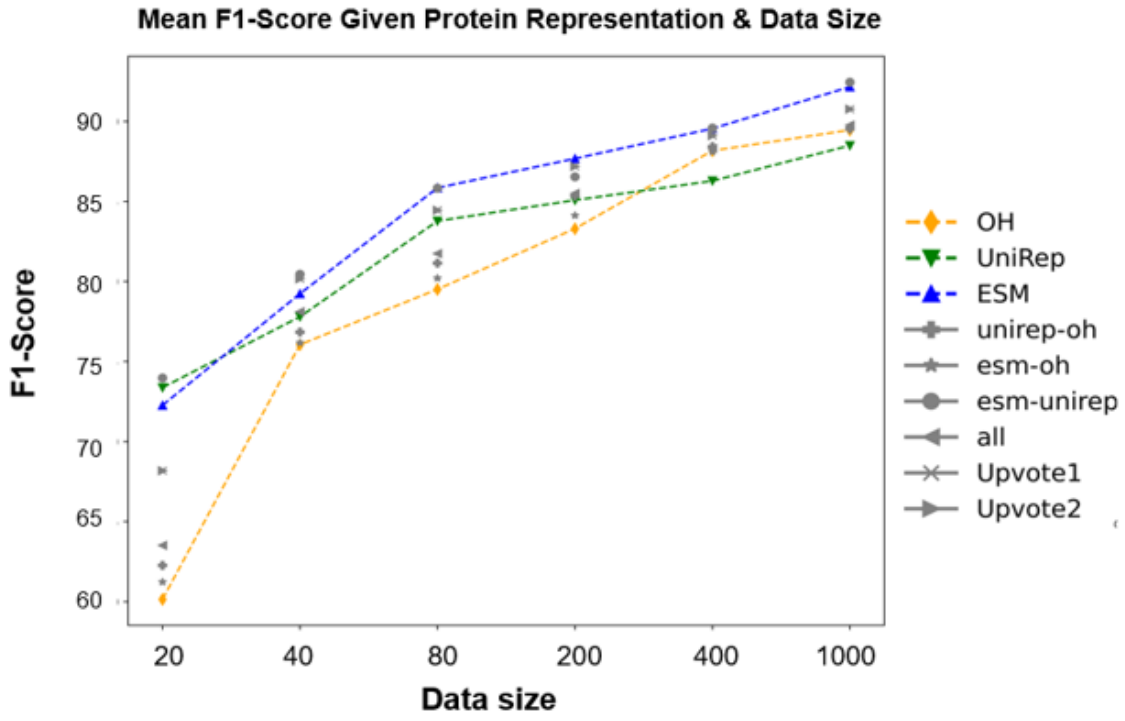


Figure 3A.10 The predictive performances (F1-score) of multiple protein representations (One-Hot, UniRep, and ESM) were evaluated across stable ($T_m \geq 60°C$; $n = 3140$) vs. unstable ($T_m \leq 35°C$; $n = 1116$) proteins. The performance of individual representations is compared against the effects of concatenating each embedding as well as ensemble methods (upvote1: hard voting, upvote2: soft voting). The violin plots were generated by repeating the analysis over 30 random seeds for sensitivity analysis. N represents the total number of data used with 0.3 as a test-size ratio. Welch t-test with unequal variances has been implemented over the obtained results to showcase the statistical significance in comparisons (refer to supplementary information for p-values). **Highlight:** In low data size ($N = 20$), One-Hot performs poorly with the mean F1-score of 0.60, and the embeddings that included One-Hot were outperformed by both ESM and UniRep. While increasing the data size resulted in increased performance for all the methods, concatenating ESM with UniRep representations obtained the best score, with a mean F1 of 0.92.

Metric	Equation
F_1	$\frac{2 * Precision * Recall}{Precision + Recall}$
Precision	$\frac{TP}{TP + FP}$
Recall, TPR	$\frac{TP}{TP + FN}$
FPR	$\frac{FP}{FP + TN}$
FDR	$\frac{FP}{FP + TP}$
NPV	$\frac{TN}{TN + FN}$

Figure 3A.11 A complete list of used criteria for MCDA their derivations from confusion matrix values.

CHAPTER 4

PREDICTING ORGANIC ANION TRANSPORTER PROTEIN-DRUG INHIBITION BY INTEGRATING STRUCTURAL INSIGHTS WITH GRAPH NEURAL NETWORKS

Abstract

Organic Anion Transporting Polypeptides (OATPs) are crucial for hepatic drug uptake, impacting drug efficacy and toxicity. However, predicting OATP-mediated drug-drug interactions (DDIs) remains challenging due to the scarcity of structural data, the dynamic behavior of these proteins, and inconsistencies in experimental protocols. This chapter introduces the Heterogeneous Graph Neural Network for Inhibition Perdition of OATPs (HIPO-GNN). This novel computational approach combines molecular modeling and graph neural networks to improve the prediction of OATP-drug inhibition. By combining ligand molecular features with protein-ligand interaction data, HIPO-GNN significantly outperforms traditional ligand-based methods, achieving median F1 and AUC scores of 0.82 and 0.81, respectively, compared to ECFP (F1: 0.68, AUC: 0.70) and RDKit (F1: 0.78, AUC: 0.75) built upon XGBoost. The analysis further evaluates the utility of interaction features in GNNs, with HIPO-GNN showing marked improvement in F1 score over a ligand-only GNN model, and no significant compromise in AUC. Beyond improving inhibition prediction, this study explores the impact of computational noise from molecular modeling, and identifies optimal thresholds in molecular docking for model improved modeling performance. Additionally, protein residues involved in inhibitory versus non-inhibitory drug interactions are characterized, specifically highlighting residues F38, K41, N213, L378, G552, and V556, which show notable differences between inhibitors and non-inhibitors. The findings enhance the predictive performance and interpretability of OATP-mediated DDIs, advocating for the integration of advanced computational techniques with standardized experimental protocols. This work contributes to the field of drug discovery and development by providing a novel approach to predict and understand OATP inhibition.

Keywords

Molecular Modeling, Drug-Drug Interaction, Machine Learning, OATP, GNN

4.1 Introduction

Drug-drug interactions (DDIs) pose a significant challenge in modern pharmacotherapy, as they can lead to the impairment or induction of critical metabolic enzymes and transporter proteins responsible for processing multiple drugs simultaneously. These interactions can be further exacerbated by genetic variation, potentially resulting in lethal outcomes [1]. Recognizing the importance of predictive DDI models in the drug development process, the US Food and Drug Administration (FDA) has recently issued guidelines advocating for their consideration in regulatory pathways for new drugs [2]. While such models have been well-established for metabolic DDIs involving cytochrome P450s, there is a notable lack of predictive models for transporter-mediated DDIs (tDDIs). This deficiency is particularly concerning in the context of organic anion transporting polypeptides (OATPs), a family of proteins responsible for transporting a wide range of drugs into the liver for bile-mediated elimination. Numerous alarming DDIs involving OATPs have been reported, highlighting the urgent need for the development of reliable predictive tDDI models [3]. Addressing this knowledge gap is crucial for ensuring the safety and efficacy of medications and ultimately improving patient outcomes.

OATPs are not only expressed in the liver for drug transport but also in other pharmacologically relevant tissues, such as the kidneys and intestines. These proteins facilitate the uptake of a diverse array of endogenous and exogenous small molecules, including hormones and drugs. To accommodate the vast number and variety of substrates processed by these tissues, OATPs exhibit a high degree of promiscuity. This promiscuity gives rise to intricate multimolecular interactions and elaborate regulatory networks that can profoundly influence drug efficacy and toxicity. Understanding the complexities of these interactions is essential for predicting and mitigating the potential risks associated with OATP-mediated drug transport. By elucidating the mechanisms underlying OATP promiscuity and the resulting regulatory networks, researchers can develop more comprehensive and accurate models for predicting tDDIs. This knowledge will ultimately contribute to the design of safer and more effective medications, as well as the optimization of drug dosing and administration strategies to minimize the risk of adverse drug

reactions.

Currently, there is no generalizable method to consistently predict how a molecule will interact with OATPs, despite valiant efforts to develop machine learning (ML) algorithms to do so [4, 5, 6]. This is due to two major challenges: (1) there is significant variability in the available in vitro data, resulting in inconsistent data labeling [7, 8], and (2) until very recently, no OATP structures had been solved, substantially limiting options for accurate structure-based analyses [4].

Challenge one—the lack of consistent in vitro data—severely constrains model applicability to the data on which it was trained. These data inconsistencies arise from different in vitro experimental conditions being used to evaluate the same ligand (e.g., with or without pre-incubation, various cell types), leading to drugs being labeled inconsistently as inhibitors or substrates depending on the context of experimental conditions [9]. Overlooking the context (i.e., chemical environment) associated with OATP-inhibition assays is deeply problematic as it distorts our understanding of how OATPs function and results in inaccurately labeled inhibitor datasets. For example, studies have shown significant variability in OATP1B1/1B3 inhibition data due to different experimental conditions, leading to inconsistencies in drug classifications [9, 10]. This variability can be attributed to various factors, including genetic polymorphisms in transporters like OATP1B1, which are major determinants of drug pharmacokinetics and can lead to variability in drug disposition [11, 12]. We identify key physicochemical properties that correlate with inhibition measurement variability, as these properties play a significant role in compound disposition and safety, influencing formulation, absorption, and toxicity during drug development [13].

Challenge two—unsolved OATP structures (and, prior to AlphaFold2 [14])—limited past ML models to only physiochemical and/or topological ligand properties. Structure elucidation precluded incorporating protein-ligand interaction analysis into past computational workflows. Adding to the complexity of this challenge is the dynamic nature of OATPs, whose transport mechanisms involve at least three conformers (outward-facing, occluded, and inward-facing). Saidi jam et al. [15] recently published cryo-EM structures of OATP1B1 (inward- and outward-facing) and OATP1B3 [16] (inward-facing) have enabled us to explore how protein-ligand interaction

features (PLIFs) may be integrated into these models. The availability of both inward- and outward-facing structures of OATP1B1 [ref] allowed us to expand our analyses to determine if purely computational workflows could capture the dynamic nature of the PLIFs to be used in predictive tDDI models.

Collectively, in this study, we (1) identify physiochemical ligand features (LFs) that highly correlate with in vitro data variability, and (2) explore the integration of computationally produced PLIFs into tDDI models. Here in, we present a workflow to extract PLIFs from OATP1B1 conformers docked against a library of experimentally characterized ligands. Adapting to the constraints of the available data, we limited evaluation of our models to data obtained in a single in vitro inhibition study from [7].

4.2 Methods

This section details the steps taken to curate data, select and label ligands, prepare ligand and protein structures, perform molecular docking, and develop machine learning models for predicting OATP inhibition.

4.2.1 Data Curation

Several studies have been conducted over the past decade that characterize and evaluate OATP interactions with ligands. Due to largely unexplained inter-study variability among the available in vitro OATP inhibition data [9], we reasoned that sourcing data from a single set of experiments would provide the most consistent framework for our study [7].

4.2.1.1 Ligand Selection & Labeling

Experimental inhibition data was obtained from Karlgren, et al. for 222 ligands with each OATP1B1, OATP1B3, and OATP2B1 Of these 225 compounds, 222 were compatible with our molecular docking workflow; we excluded the remaining three ligands. As was done in the original Karlgren study, compounds with reported inhibition percentages greater than 50% were labeled as inhibitors (Class = 1), while those with inhibition percentages less than or equal to 50% were labeled as noninhibitors (Class = 0).

4.2.1.2 Training & Test Sets

Under constraints that each test set must have an equal class balance, twenty test sets were constructed by randomly selecting 10% of ligands from the total dataset. For each of the twenty test sets, the corresponding training set consisted of the remaining 90% of compounds not in the hold-out test set.

4.2.2 Structural Modeling

4.2.2.1 Ligand Structure Preparation

Canonical SMILES were compiled from PubChem [17] for all 222 compounds to be used for structural modeling. Ligand representations were converted from SMILES format to three dimensional structures (SDF format), energetically minimized, and protonated (to pH 7.5) using Open Babel [18].

4.2.2.2 Protein Structure Preparation

Protein sequences of human OATP1B1, OATP1B3, and OATP2B1 were obtained from the UniProt database in FASTA format [19]. The cryo-EM structures of the inward- and outward-facing OATP1B1 conformers (PDB IDs 8HND, 8HNB) [20], along with the inward-facing OATP1B3 conformer (PDB ID 8PG0) [16], were retrieved from the Protein Data Bank [21]. These structures were prepared for docking by removing water molecules, ions, and small molecule ligands using PyMOL [22] and then relaxed using the Rosetta FastRelax protocol [23] to minimize any structural artifacts from the cryo-EM experiments. Due to the lack of experimental structures for the outward-facing OATP1B3 and both conformers of OATP2B1, we utilized AlphaFold2 [24, 25], a state-of-the-art protein structure prediction algorithm, to generate these missing structures. The AlphaFold2 predicted structures were validated by comparing them to the available cryo-EM structures of OATP1B1 and OATP1B3 using root-mean-square deviation (RMSD) calculations, as described by Sala et al. [26]. The low RMSD values observed between the predicted and experimental structures provided confidence in the quality of the AlphaFold2 predictions for the missing conformers.

4.2.2.3 Ligand-Protein Docking

Molecular docking was performed between all 222 ligands and both the outward and inward conformers of human OATP1B1, OATP1B3, and OATP2B1 using the Rosetta modeling suite (version 3.12) [27]. The RosettaLigand score function (pre-talaris2013) was chosen for its proven performance in small molecule docking compared to other Rosetta score functions and external docking algorithms [28]. The docking protocol involved the following steps: (1) concatenating the ligand conformer files with the prepared protein structures, (2) defining the starting coordinates for the ligand within the known transport site of the OATPs [20] using an XML file, (3) specifying the dimensions of the grid in which the ligand is allowed to move during the docking simulation, and (4) running the docking simulation for all ligands. To ensure a thorough sampling of the binding space, 1000 docked poses were generated for each ligand-protein pair. To further validate the docking results, the top-ranked poses for each ligand were visually inspected for any unrealistic conformations or interactions. Additionally, the binding energies of the top poses were compared to available experimental binding data [7] to assess the accuracy of the RosettaLigand score function in ranking the poses. These validation steps provided additional confidence in the quality of the docking results for downstream analysis.

4.2.2.4 Ranking Docked Poses for Each Ligand

For each docked pose, Rosetta reports the ligand-protein interface energy (`interface_delta_X`). This was used as a relative metric to rank the 1000 docked poses for a single ligand-OATP conformer pair. Poses with the lowest `interface_delta_X` scores were interpreted as higher-quality docked poses (i.e., more stable binding). For all poses, `interface_delta_X` scores were recorded in a CSV file for use in subsequent analyses (e.g., determining the optimal number of poses to capture transporter dynamics).

4.2.3 Feature Selection, Encoding, and Preprocessing

Two distinct types of feature sets were generated for each OATP conformer: traditional ligand-only molecular descriptors (ligand features, LFs) and protein-ligand interaction features (PLIFs).

4.2.3.1 Ligand Features (LFs)

Two types of vectorized molecular representations, extended-connectivity fingerprints (ECFP6) and RDKit physicochemical descriptors, were generated for each of the 222 ligands using the pinky [1] and RDKit [29] python libraries, respectively. As both types of LFs consist of numeric, noncategorical data, no encoding was required. Variance (<0.001) and correlation coefficient (>0.99) thresholding was performed to remove any obvious or redundant features.

4.2.3.2 Protein-Ligand Interaction Features (PLIFs)

The Protein-Ligand Interaction Profiler (PLIP) [30] was used to generate interaction profiles for each docked pose. PLIP produces intricate features to describe each interaction present in a docked pose, including interaction type, distance, angle, donor and acceptor atom IDs, etc. These comprehensive PLIFs were parsed into a CSV file, with each row containing vectorized PLIFs for a single docked pose. Raw PLIF csv files were processed with an initial variance filtering (<0.001) to remove any of the non-categorical, numerical features (e.g., interaction distance, angle) with variance below this threshold. One hot encoding was then performed for all categorical features. Encoded data was then filtered by correlation coefficient (>0.99) to remove highly redundant features.

4.2.3.3 Protein-Ligand Interaction Feature Reduction: Minimal PLIFs (mPLIFs)

To capture a more interpretable, less noisy PLIP representation, a simplified set of PLIFs (minimal PLIFs, mPLIFs) was parsed from the PLIF data. These more minimal features define whether each OATP residue is involved in a specific interaction type in a given pose (0=not involved, 1=involved). The mPLIF naming system follows the pattern [interaction type].[OATP residue] (examples in supplementary Figure 4D.1).

4.2.4 Machine Learning Classification Models

This section elaborates on data processing techniques to improve ML classification tasks, model selection, from machine learning classifiers to neural network for mutlilabel classification task.

[1] https://github.com/ubccr/pinky/

4.2.4.1 Data Preprocessing

Prior to training the machine learning models, several preprocessing steps were applied to the dataset. One-hot encoding was performed on categorical features to convert them into a binary representation suitable for the models. Variance and multicollinearity correlation thresholding were employed to remove features with low variance (<0.001) and high correlation (>0.99), respectively, reducing the dimensionality of the feature space and mitigating the risk of overfitting.

A drug-wise splitting approach was adopted to create the holdout test sets to ensure a robust evaluation of the models' performance. This approach, as opposed to entirely random pose-wise splitting, prevents leakage of information between the training and test sets. Five balanced test sets were created, each containing 10% of the total dataset, with an equal representation of inhibitors and non-inhibitors. The remaining 90% of the data was used for training and cross-validation. Additionally, feature normalization using StandardScaler was applied to ensure that all features have zero mean and unit variance, improving the convergence and stability of the machine learning algorithms.

4.2.4.2 Classifier Training Evaluation

Several machine learning algorithms, including logistic regression, support vector machines (SVM), random forests, and XGBoost, were trained and evaluated using the preprocessed data. The training process involved k-fold cross-validation, where the training data was split into k subsets, and the models were trained and validated k times, using a different subset for validation each time. This approach helps to assess the models' performance and stability across different subsets of the data. The results for classification are reported for XGBoost due to consistent out-performance.

The trained models were then evaluated on the holdout test sets to assess their generalization performance. Metrics such as accuracy, precision, recall, F1-score, and area under the receiver operating characteristic curve (AUC-ROC) were calculated to provide a comprehensive assessment of the model's performance in predicting OATP inhibition.

4.2.4.3 Multi-label Neural Network

In our approach to predicting OATP inhibition, we developed a multi-label neural network classification model. This methodology facilitates simultaneous prediction of inhibition across multiple OATP subtypes (1B1, 1B3, and 2B1), providing a more holistic perspective on potential drug interactions compared to traditional single-family classification models. The multi-label strategy excels in revealing the intricate interplay between drugs and various OATP subtypes, enabling the discovery of cross-subtype inhibition patterns that might be overlooked in isolated analyses.

Our implementation featured a custom-designed InhibitorClassifier neural network architecture, consisting of multiple hidden layers with ReLU activation functions. For multi-label classification, we employed the BCEWithLogitsLoss criterion, which is well-suited for handling multiple binary labels. To enhance generalization and prevent overfitting, we incorporated an early stopping mechanism. Additionally, hyperparameter tuning was conducted using the Optuna optimization package [31], ensuring robust performance across different OATP subtypes. This comprehensive setup highlights the efficacy of our approach in tackling the challenges of OATP-mediated drug interaction prediction.

4.2.5 Heterogeneous Graph Neural Network for Inhibition Perdition of OATPs (HIPO-GNN)

A Heterogeneous Graph Neural Network (HeteroGNN) is constructed for each protein-ligand complex to capture the intricate interactions between the protein and the ligand. The graph contains two types of nodes: amino acid residues and ligand atoms. For amino acids in the protein sequence, 28 features are extracted (Table 4.1), including one-hot encoded residue type, molecular weight, aromaticity, isoelectric point, hydrophobicity, flexibility, and secondary structure propensities. These features capture the essential properties involved in protein-ligand interactions. For ligand atoms, 79 atomic features are derived (Table 4.2), including one-hot encoded atom type, number of heavy neighbors, formal charge, hybridization state, presence in rings, aromaticity, atomic mass, van der Waals radius, covalent radius, chirality, and number of implicit hydrogens. This set of atomic features provides a detailed representation of the ligand's chemical structure and

properties, ensuring that the HeteroGNN effectively models the complex dynamics of protein-ligand interactions.

Edges in the constructed graph represent the inter-interactions between amino acids and atoms, as well as the intra-interactions among atoms within the ligand. For protein-ligand interactions, 332 edge features are extracted (Table 4.3), including interaction type, distance, angle, and other geometric properties. For ligand bonds, 10 features are derived (Table 4.4), such as bond type, conjugation, presence in rings, and stereochemistry. This graph construction process captures the protein-ligand complex's local and global structural information, providing a comprehensive representation for subsequent analysis. Lastly, Table 4.5 presents the edge formulation indices and their corresponding sizes, detailing the interactions between amino acids (aa) and atoms, as well as the bonds between atoms.

4.2.5.1 Model Architecture

The HeteroGNN architecture is designed to model the protein-drug interaction and perform a graph classification task. It uses separate convolution layers for amino acid-to-atom, atom-to-amino acid, and atom-to-atom interactions. Edge features are first processed through Multi-Layer Perceptrons (MLPs) to capture interaction characteristics. Convolutional layers (i.e., NNConv) then update node features based on neighboring nodes and processed edge features, propagating information across the protein-ligand interface. A HeteroConv layer aggregates outputs from these convolutions, combining information from different interaction types (i.e., inter and intra). Linear projection layers then map atom and amino acid features to a common space. Global mean pooling is applied to create fixed-size representations for the ligand and protein. These pooled features are concatenated and passed through a classification network consisting of linear layers with ReLU activation and dropout. The network outputs a single value predicting the compound's inhibitor status. This architecture integrates heterogeneous information from protein-ligand complexes, considering both local interactions and global structure to predict inhibitory activity.

4.2.5.2 Data Splitting and Cross-Validation

The data splitting strategy aims to ensure robust evaluation and minimize data leakage. 10% of unique ligands are randomly selected as the test set, including all related docked poses. The remaining data is split into training (90%) and validation (10%) sets based on unique ligands. This process is repeated 20 times, creating 20 different train/validation/test splits for a more comprehensive assessment of model performance.

4.2.5.3 Training and Evaluation

The model is trained using the AdamW optimizer with weight decay. The OneCycleLR learning rate scheduler is used for adaptive learning rates. Mixed precision training improves computational efficiency. Binary Cross-Entropy with Logits serves as the loss function. Early stopping, based on validation loss with a patience of 5 epochs, prevents overfitting. Each model trains for up to 100 epochs or until early stopping occurs.

For each of the 20 data splits, a separate model is trained and evaluated on the corresponding test set. Multiple metrics are calculated: accuracy, precision, recall, F1 score, balanced accuracy, and AUC-ROC. This thorough evaluation provides a robust understanding of the model's predictive capabilities across various classification aspects.

Table 4.1 Node Features for Amino Acids.

Feature	**Type**	**Size**
One-Hot Encoded Residue	Binary Vector	20
Molecular Weight	Numerical	1
Aromaticity	Numerical	1
Isoelectric Point	Numerical	1
Hydrophobicity	Numerical	1
Flexibility	Numerical	1
Secondary Structure	Numerical Vector	3

4.3 Results

To address the challenge of predicting OATP-mediated drug-drug interactions (DDIs), we developed a comprehensive computational approach that integrates molecular modeling, machine learning classifiers, and Graph Neural Networks (GNNs). Our novel structure-based OATP-ligand

Table 4.2 Node Features for Atoms.

Feature	Type	Size
One-Hot Encoded Atom Type	Binary Vector	43
Number of Heavy Neighbors	Binary Vector	6
Formal Charge	Binary Vector	8
Hybridization Type	Binary Vector	7
In a Ring	Binary	1
Aromaticity	Binary	1
Atomic Mass	Numerical	1
Van der Waals Radius	Numerical	1
Covalent Radius	Numerical	1
Chirality	Binary Vector	4
Number of Implicit Hydrogens	Binary Vector	6

Table 4.3 Edge Feature for Interaction (Amino Acid Interaction with Atom).

Feature	Description
Interaction Type	hydrophobic, halogen, hydrogen bond, salt bridge, pi-cation, pi-stacking
Residue Type	One-hot encoded vector for protein residues involved in the interaction
Distance	Normalized distance between the interacting protein residue and ligand atom
Angle	Normalized angle measurements (if applicable, e.g., in hydrogen bonds)
Donor/Acceptor Type	One-hot encoded vector for donor/acceptor roles (in halogen/hydrogen bonds)

Table 4.4 Edge Features for Intra-Interactions (Within Ligand Atoms).

Feature	Description
Bond Type	One-hot encoded vector for bond types (e.g., single, double, aromatic)
Conjugation	Binary for whether the bond is conjugated
Ring Status	Binary for whether the bond is part of a ring
Stereochemistry	One-hot encoded vector for stereochemistry (e.g., cis, trans)

Table 4.5 Edge Formulation.

Edge Index	Size
'aa', 'interacts with', 'atom'	332
'atom', 'interacts with', 'aa'	332
'atom', 'bonded to', 'atom'	10

interaction prediction platform, HIPO-GNN, combines ligand molecular features with protein-ligand interaction data to significantly enhance inhibition prediction accuracy. The study unfolds in several key stages. Initially, we employed molecular docking followed by a Protein-Ligand Interaction Profiler (PLIP) [30] to extract interaction features, generating Protein-Ligand Interaction Fingerprints (PLIFs). These PLIFs enabled comparisons between key residues interacting with inhibitors versus non-inhibitors, highlighting specific residues (F38, K41, N213, L378, G552, and V556) that show notable differences in their interactions. We then utilized these interaction features to build and optimize machine learning classifiers, exploring the effect of docked pose sampling on predictive performance. The culmination of this process was the development of HIPO-GNN, designed to capture intricate intra- and inter-molecular dynamics between drugs and proteins. HIPO-GNN significantly outperformed traditional ligand-based methods, achieving median F1 and AUC scores of 0.82 and 0.81, respectively, compared to ECFP (F1: 0.68, AUC: 0.70) and RDKit (F1: 0.78, AUC: 0.75) models built upon XGBoost. Additionally, we conducted a meta-analysis to underscore the critical importance of data quality in studying complex transporter proteins like OATPs. This analysis revealed challenges such as discrepancies in experimental dataset labeling under different conditions and identified physicochemical features of drugs highly correlated with these discrepancies. We also demonstrated how the availability of crystal structures influences ML prediction outcomes, showing significant variations in results between proteins with and without these structures. By addressing these aspects, our integrated approach aims to enhance the prediction accuracy for OATP inhibition, provide more comprehensive insights into OATP-ligand interactions, and contribute to the design of safer and more effective medications.

4.3.1 Structure-Based Protein-Drug Interaction Prediction

This section explores the effectiveness of structure-based and drug-based features in predicting protein-drug interactions, specifically for the OATP1B1 subfamily which its crystal structure has been determined. We present key findings from our studies on molecular docking simulations and inhibition classification, emphasizing the performance of various feature representations (e.g., ECFP, PLIP, RDKit) in classification tasks. Additionally, our analysis identifies critical residues

and interaction types that differentiate inhibitory drugs from non-inhibitors. The results indicate that structure-based features are most effective when high-quality crystal structures for both inward and outward conformers are available. The analysis of key residues and interaction types also shows promising results and agreement with recent findings about OATP transport. For instance, non-inhibitors exhibited approximately twice as many pi-stacking interactions with residues Y352 and F356 in the inward conformer compared to known inhibitors. This observation aligns with the findings of Shan et al.[20], highlighting the significance of Y352 and F356 in the binding and transport of known OATP1B1 substrates.

4.3.1.1 Molecular Docking Simulation Generated Novel OATP-Ligand Interaction Interface.

Molecular docking simulations have identified novel interaction interfaces between OATP1B1 and various ligands. Utilizing high-quality cryo-EM structures of OATP1B1, we performed extensive docking simulations to model the binding poses of 222 experimentally characterized ligands with the inward-facing conformer of OATP1B1 [20]. From each docking pose, we extracted protein-ligand interaction features (PLIFs), highlighted in orange in Figure 1A. These features encompass critical interaction details such as distance, angle, and the specific atoms involved, capturing the intricacies of potential interaction hotspots. The distribution of Rosetta interface binding energies for OATP1B1, OATP1B3, and OATP2B1 transporters can be found in the supplement appendix (Figure 4C.1).

4.3.1.2 Interacting Residues in Docked Poses are Generally Consistent with Experimental Findings.

To identify key residues that characteristically interact more often with inhibitors, the distribution of interactions per OATP1B1 residue was analyzed. This involved parsing the comprehensive Protein-Ligand Interaction Fingerprints (PLIFs) into a subset of interaction features, termed minimal PLIFs (mPLIFs). Specifically, mPLIFs encompass only the interaction types present at each OATP1B1 residue. Figure 6.1B (left) highlights the frequency of any interaction type occurring at each orthosteric site residue (Y352, F356, and F386) identified by Shan et al [20]. Inhibitory ligands were observed to interact significantly more often with the phenylalanine residues

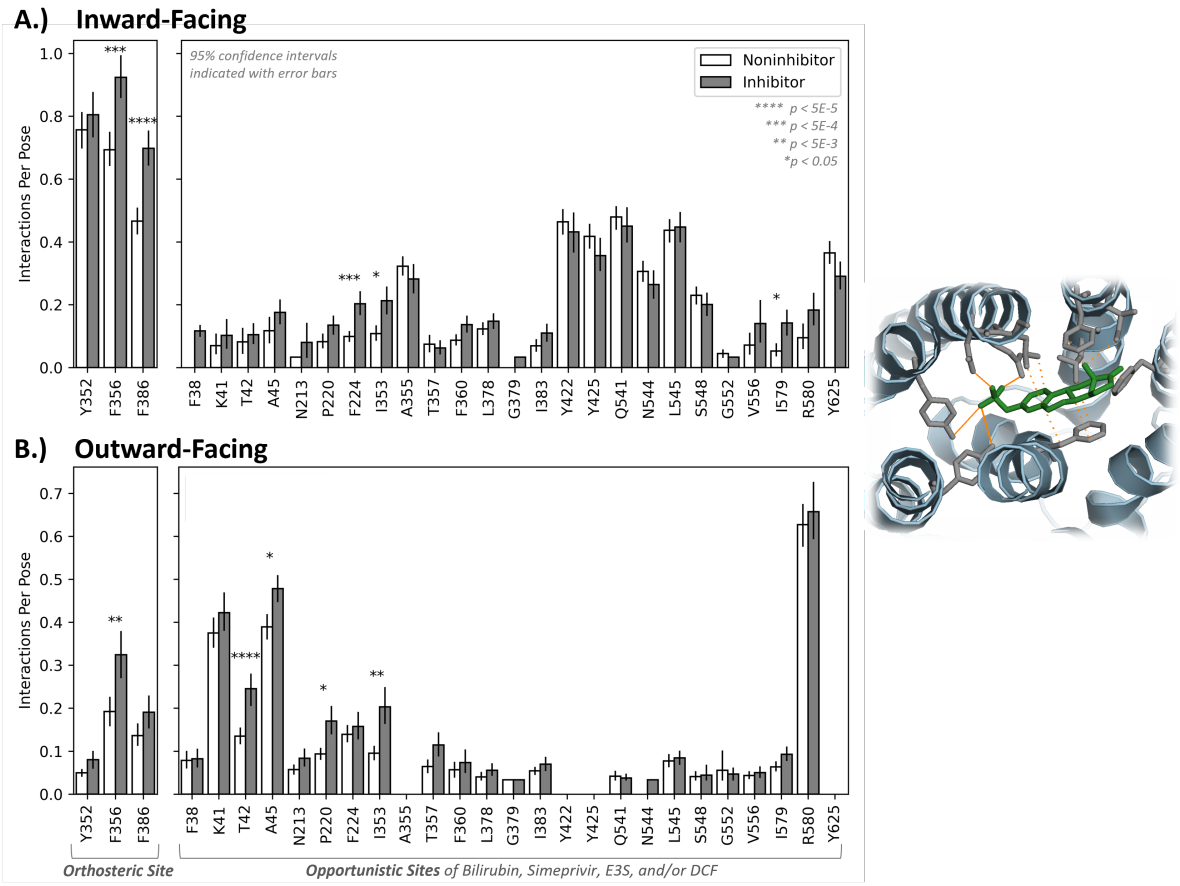


Figure 4.1 Interaction probabilities of OATP1B1 residues with inhibitors and non-inhibitors. A. Displays the inward-facing conformers of OATP1B1, and panel (B) Exhibits the outward-facing conformers for OATP1B1. Each panel is divided to show interaction probabilities at the orthosteric site (left) and opportunistic sites (right). White bars represent non-inhibitors and gray bars denote inhibitors. Error bars indicate 95% confidence intervals. Statistical significance is marked by asterisks where *p<0.05, **p<0.01, ***p<0.001, and ****p<0.0001, as determined by Mann-Whitney U-tests with Bonferroni correction. The inset structural model (right) illustrates E3S (green) docked to OATP1B1, with protein-ligand interaction fingerprints (PLIFs) highlighted in orange.

in the orthosteric site than non-inhibitory ligands ($p < 5 \times 10^{-4}$ for F356, and $p < 5 \times 10^{-5}$ for F386).

4.3.1.3 Structure-Based vs. Ligand-Based ML Prediction Perform Distinctly in OATP Families

To obtain a holistic view of structure-based and drug-based features' performances, a multilabel classification study uses PLIP (structure-based), RDKit, and ECFP (both drug-based) features. This approach allows us to evaluate the overall effectiveness of each feature set in distinguishing the

protein-drug interaction (i.e. whether an inhibitor for x or not) simultaneously for all three OATP families.

Figure 3.2 compares the distribution of F1 Scores and AUC values for these feature sets. The results show that PLIP-based features achieved a relatively high median F1 Score, suggesting they are effective in predicting true positive rates across multiple labels. RDKit features exhibited the highest median AUC, indicating a strong ability to distinguish between classes. The ECFP features displayed more variability in both F1 Scores and AUC values, suggesting some inconsistency in performance.

To further elucidate the performance differences across OATP subfamilies, we conducted a comparative analysis of structure-based and ligand-based prediction methods for each transporter type. As shown in Figure 4B.1, RDKit-based machine learning models consistently outperformed other approaches for OATP1B3 and OATP2B1, indicating that ligand-based features are particularly crucial for predicting inhibition in these subtypes. In contrast, OATP1B1 demonstrated superior performance with structure-based modeling approaches, suggesting that the availability of high quality structural data for this transporter provides valuable insights into protein-ligand interaction dynamics while maintaining alignment within the margins. Notably, OATP2B1 posed the greatest challenge for prediction among the tested families in the Karlgren dataset, highlighting the complex nature of its interactions. These findings underscore the importance of tailoring computational approaches to the specific characteristics of each OATP subfamily and informed our decision to focus subsequent analyses on OATP1B1, leveraging its amenability to structure-based prediction methods for further enhancement of predictive performance.

4.3.2 GNN Model Captures Nuanced Protein-Ligand Interactions and Improves Inhibition Prediction Performance.

We present the performance of our novel HIPO-GNN model compared to other state-of-the-art methods for predicting OATP inhibition. Our results demonstrate that HIPO-GNN, which integrates both ligand features and protein-ligand interaction data, achieves superior performance in terms of F1 score, indicating a better balance between precision and recall in inhibition prediction.

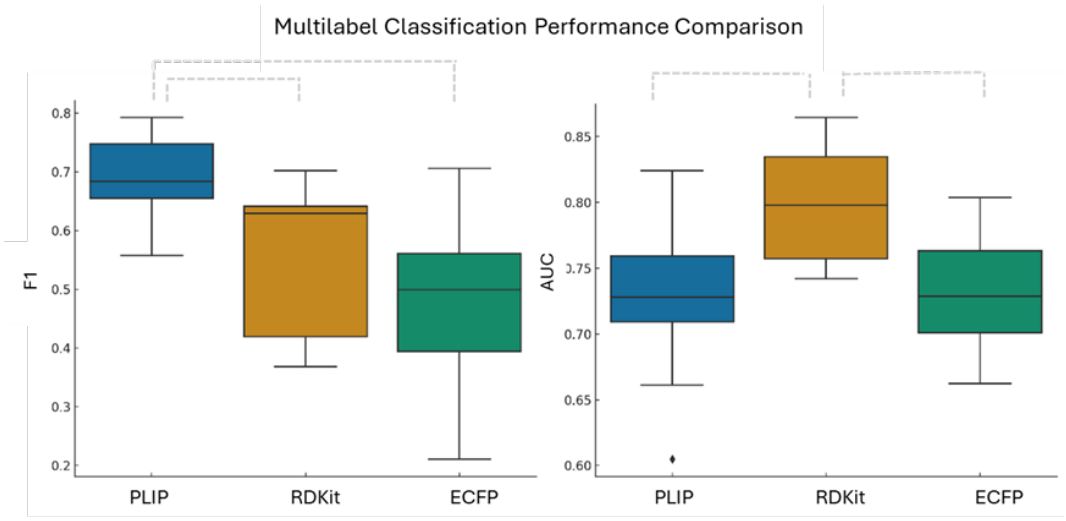


Figure 4.2 PLIP-based features demonstrated the highest median F1 Scores, while RDKit features exhibited the highest median AUC value. The dotted lines showed statistical significance among the performances.

This improvement is particularly noteworthy given the complex nature of OATP-mediated drug interactions and the challenges associated with their prediction.

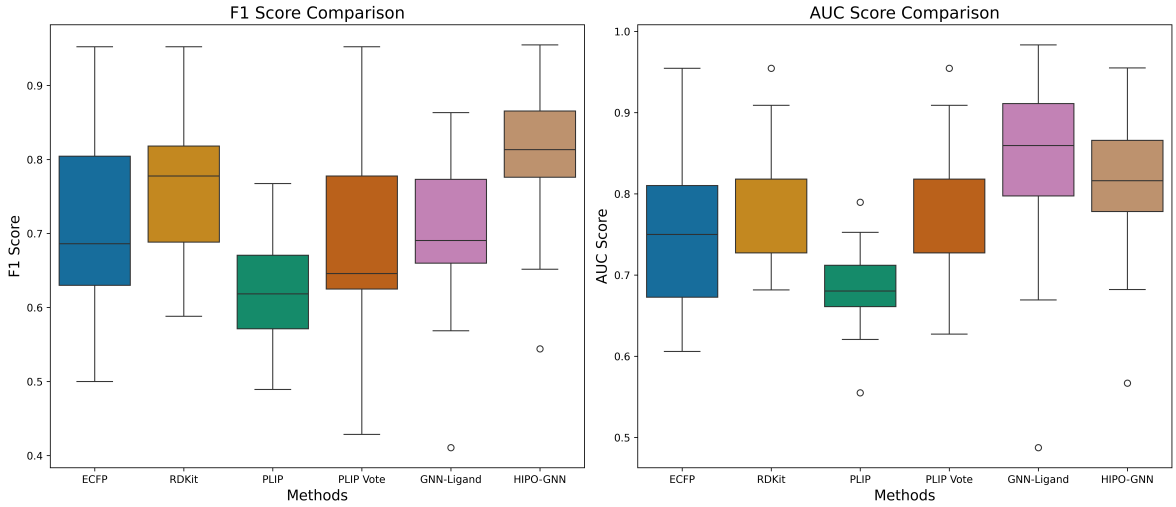


Figure 4.3 The box plots display the distribution of scores for ECFP, RDKit, PLIP, PLIP Vote, HIPO-GNN, and GNN-Ligand methods. HIPO-GNN shows superior F1 scores, indicating better overall prediction accuracy while maintaining competitive AUC scores.

Post-analysis of these results reveals interesting insights. While HIPO-GNN demonstrates the highest median F1 score, its AUC score, though competitive, is slightly lower than that of the GNN-Ligand model. This discrepancy might be attributed to the additional noise introduced by the

protein-ligand interaction features in HIPO-GNN, which are absent in the ligand-only GNN model. The interaction features, while providing valuable information about the binding dynamics, may also introduce some level of uncertainty due to the inherent variability in protein-ligand interactions and potential limitations in the molecular docking process. Nevertheless, the higher F1 weighted score suggests that HIPO-GNN's predictions are more reliable in predicting OATP inhibition across the dataset, taking into account the class distribution. This indicates that HIPO-GNN performs well on both common and rare cases of inhibition, providing a more comprehensive and practically useful model for drug-drug interaction predictions in real-world scenarios where the class imbalance is typical.

4.3.3 From Noise to Knowledge

In this section, we aim to enhance the interpretability and knowledge gained from our OATP inhibition prediction model. We explored the physicochemical properties most closely associated with OATP inhibition, providing a mechanistic understanding of the inhibition process. Additionally, we introduce simple machine learning engineering techniques to maximize the signal obtained from our computational modeling, effectively separating meaningful patterns from noise. This comprehensive approach not only enhances our model's performance but also contributes to a deeper understanding of OATP inhibition mechanisms, addressing challenges in data consistency, and improving the overall reliability of drug-drug interaction predictions for OATP-mediated transport.

4.3.3.1 Consideration of Multiple Docked Poses Provides Dynamic Insight at the Expense of Signal.

For each ligand, Rosetta docking simulations generated 1,000 poses of varying quality. The *Rosetta interface_delta_X* energy score was employed as a quality metric to represent the binding interface energy. Figure 4.4. A illustrates the distribution of interface energy scores for the most stable 100 poses for both inhibitors and noninhibitors docked to inward-facing OATP1B1. These distributions indicated that the docked interfaces of noninhibitors to the inward-facing conformer are significantly more stable compared to those of inhibitors.To investigate the impact of including multiple poses on model performance, datasets were created containing one, two,

three, five, ten, 20, 30, 50, 75, and 100 of the 'best' poses per ligand. The performance of the XGBoost model was observed to peak when approximately 30 poses per ligand were included in the PLIF dataset. Interestingly, mPLIF models displayed maximum performance with only two poses per ligand (Figure 4.4B). We hypothesized that removing nuanced, quantitative PLIFs might reduce noise and improve model performance, although this was not observed in our results. Conversely, regardless of the number of poses included per ligand, XGBoost models trained on the comprehensive PLIF dataset generally outperformed those trained on the simplified mPLIFs (Figure 4.4B). The best-performing conditions (30 poses, full PLIFs) were selected for constructing the HIPO-GNN model. In addition to implications on predictive performance, expanding datasets to include multiple poses per ligand may broaden the dynamic perspective of transporter interactions. The residue-interaction distributions of our docking-derived PLIFs were evaluated for consistency with experimentally identified interaction sites. Our analyses were divided into three groups of OATP1B1 residues: 1. We investigated PLIFs involving the orthosteric site (Y352, F356, F386), which is consistently seen to be involved in OATP1B1 transport function [20]. 2. We defined a second category of residues from known opportunistic sites, groups of residues important for the binding and/or transport of only certain ligands. Our analyses considered a broadly encompassing set of residues present in opportunistic sites of simeprivir, estrone-3-sulfate, bilirubin, and 2',7'-dichlorofluorescein (DCF). We refer to this grouping as the 'opportunistic sites' for brevity. 3. Lastly, we investigated PLIFs involving residues previously unrecognized in either the orthosteric or opportunistic sites.

As expected, increasing the number of poses included in our datasets resulted in a greater diversity of OATP1B1 residues captured in the PLIFs of each ligand. Figures 4.4 C–E depict the [coverage = for each ligand, out of all n-poses, how many of the residues from the (a) orthosteric, (b) opportunistic, (c) neither site were present.

4.3.3.2 Physiochemical Feature Importance Analysis for OATP Inhibition

Building upon our previous identification of key protein residues involved in OATP inhibition 6.1, we also focuses on physicochemical properties of drugs that contribute to this inhibition.

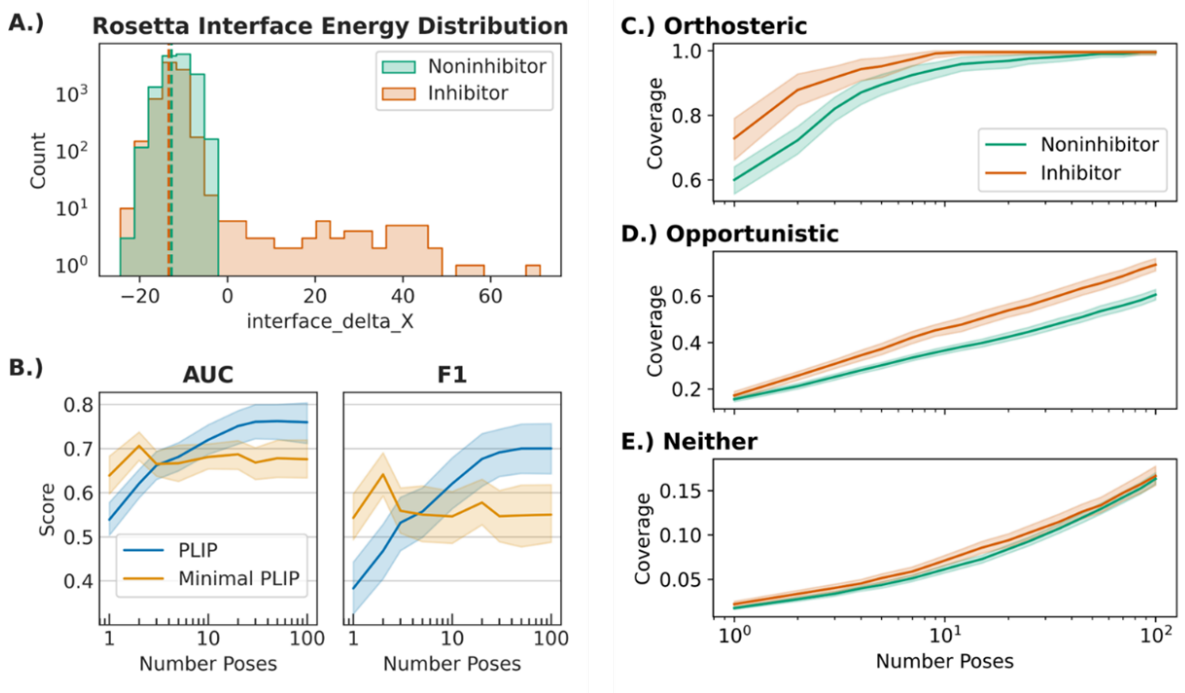


Figure 4.4 Multiple Docking Pose Effect in ML Performance and OATP Inhibition Mechanism Insight. A. Distribution of Rosetta binding interface energy scores (*interface_delta_X*, in Rosetta energy units) for 100 poses of each ligand docked to the inward-facing OATP1B1. Experimentally characterized noninhibitors are shown in green, and inhibitors in red. Dashed lines indicate the 30th percentile – the 'best' 30 poses. (B) All panels with 95% confidence intervals. B. Area under the curve (AUC) and F1 scores plotted for PLIP and mPLIP feature sets for a variable number of poses per ligand. Scores shown are for models incorporating majority voting. (C - E) Fraction of unique residues from the orthosteric site, opportunistic sites, and remaining protein sequence captured in PLIFs as a function of the number of poses. Shaded regions indicate 95% confidence intervals in panels B - E.

Understanding both protein-ligand interactions and drug characteristics is crucial for a detailed view of OATP-mediated drug-drug interactions. While achieving strong predictive performance is crucial, the interpretability of our models offers critical insights for advancing drug development and ensuring safety assessments.. Our analysis in 4.5 reveals the key physicochemical properties that influence OATP inhibition across multiple subtypes and specifically for OATP1B1. These findings not only enhance our mechanistic understanding but also offer practical guidance for rational drug design.

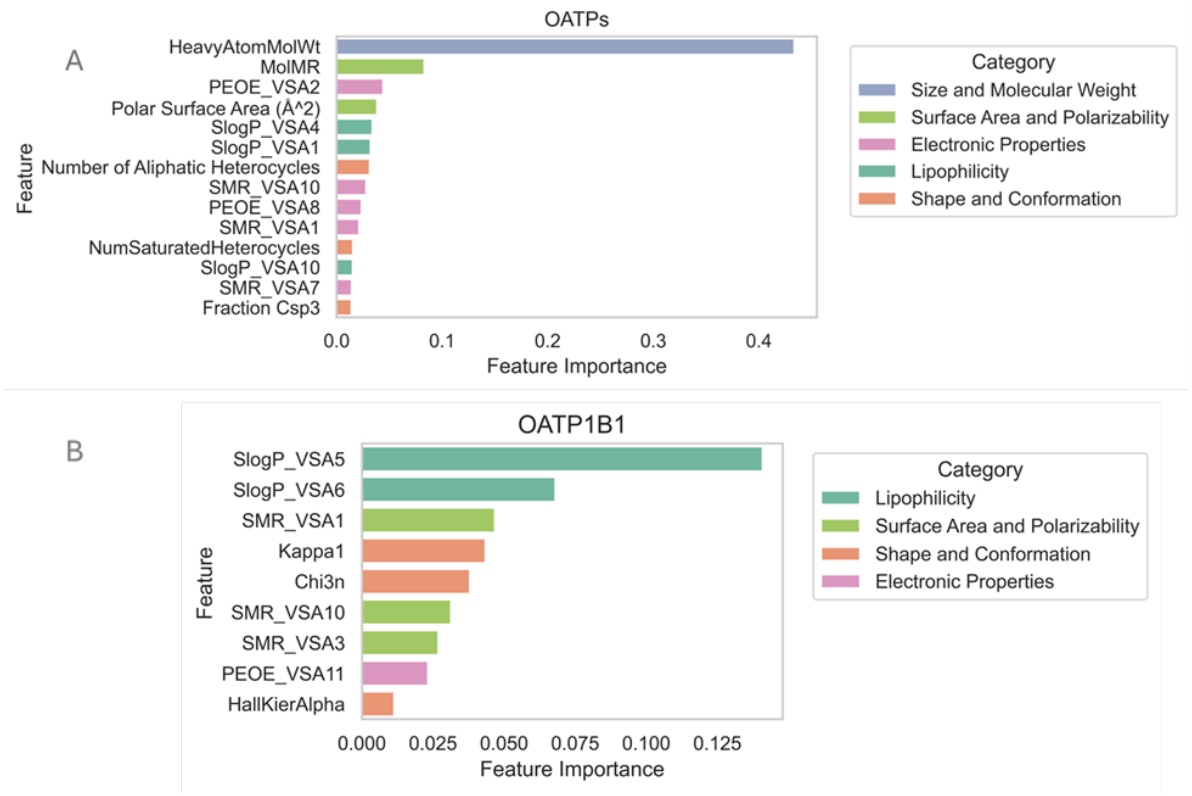


Figure 4.5 Feature importance analysis for OATP inhibition prediction. A. Top features contributing to the inhibition prediction across OATP1B1, OATP2B1, and OATP1B3, categorized by physicochemical properties. The model classifies compounds as inhibitors only if they inhibit all three OATP subtypes. B. Specific features important for OATP1B1 inhibition prediction. Feature importance is quantified based on the model's reliance on each feature for predictions. Color-coding represents different categories of physicochemical properties. Key features include molecular weight (*HeavyAtomMolWt*), molar refractivity (*MolMR*), partial charge distributions (*PEOE_VSA2*), lipophilicity (*SlogP_VSA1/5/6*), and molecular shape descriptors (*SMR_VSA1/3/7/10, Kappa1, Chi3n*).

4.3.4 Discussion

Understanding the transport mechanisms of OATPs is particularly challenging due to their membrane-bound nature and dynamic conformational changes. Unlike soluble proteins, OATPs are embedded in cellular membranes, complicating their study. Their polyspecificity, (i.e. ability to interact with a wide range of substrates) exacerbates this challenge as it demands detailed knowledge of diverse and often overlapping binding interactions. The hydrophobic nature of OATPs makes them difficult to crystallize, resulting in a scarcity of high-resolution structural data, which impedes the development of accurate models for predicting drug interactions. This lack of structural

information, combined with their significant flexibility and propensity for instability, presents challenges throughout various stages of research, including expression, solubilization, purification, and structural elucidation. Consequently, the principles governing OATP drug transport and inhibition mechanisms are not fully understood. This necessitates resource-intensive in vitro and in vivo assays and highlights the need for advanced computational methods to fill these gaps.

Machine learning (ML) techniques have been extensively applied to predict interactions between drugs and OATPs and provide transport insights. Researchers have made progress in identifying and predicting OATP-mediated drug transport and inhibition mechanisms by distinct ML algorithms and encoding various features of drug-protein interactions into ML platforms. Khuri et al. [32] employed a Random Forest classifier to screen a virtual library of over 5000 compounds, identifying potential OATP2B1 inhibitors. The descriptors for the ligands encompassed physicochemical and topological properties of the drugs, capturing the essence of their interaction capabilities with the transporter. To enhance the precision of their initial findings, they proceeded to evaluate the shortlisted compounds via docking against several comparative protein structure models of OATP2B1. This docking process served as a subsequent step to further refine and filter the list of candidates. Among the ten selected drugs for experimental validation, three were confirmed as potent inhibitors of OATP2B1-mediated E3S transport. In another study, De Bruyn et al.[33] incorporated protein sequence physiochemical features and ligand circular fingerprint (ECFP6) into a random forest classifier to predict OATP1B1/3 inhibitors. The trained model, prospectively validated on 54 compounds not previously included in the original library, predicted their inhibitory potential against OATP1B1 and OATP1B3, with 80% and 74% of the compounds correctly identified for their inhibition effects on OATP1B1 and OATP1B3, respectively. Subsequent studies employed multiple strategies such as logistic regression, SVC, XGBoost, and ensemble models. These approaches leveraged varied representations of features, incorporating extensive protein sequence data (such as amino acid compositions and sequence order features) along with ligand attributes (e.g. ECFP, and RDK fingerprints). This expanded feature set facilitated the early detection and pharmacokinetic profiling of OATP-drug interactions [34]. Recently, Hao Duan et al.

[35] presented a more robust approach for predicting transporter inhibitors using both conventional machine learning and multi-task deep learning models. The study leverages a comprehensive dataset curated from multiple sources (i.e. Karlgren, ChembL, De Bruyn, [12 papers, 6 databases]), achieving high predictive performance, particularly with the MLT-GAT model. However, while the dataset curation is extensive, the study faces challenges with data imbalance and the need for higher quality data to enhance model accuracy and reliability further. This work highlights the high potential and current limitations of computational methods in transporter-drug interaction prediction. While influential, these studies face several limitations. Notably, the inconsistency in public datasets regarding the classification of drugs as inhibitors or substrates leads to discrepancies in definitions resulting in misguided labeling across ML implementations. Additionally, most studies concentrate solely on drug attributes, while crucial, may not fully capture the nuances necessary for accurate predictions of interactions with complex and dynamic proteins. The lack of protein structural information in ML models limits the ability to comprehensively incorporate features from both drugs and proteins. Given the complexity of the system, a more sophisticated modeling approach is required to address the challenges posed by the structural variability of proteins and the intricate nature of drug-protein interactions.

This study demonstrates the value of integrating structural information and advanced machine learning techniques in predicting OATP-mediated drug interactions. OATPs' crucial role in hepatic drug uptake, combined with the challenges of their membrane-bound nature and limited structural data, necessitates innovative approaches. Our structure-based approach leverages protein-ligand interaction features and heterogeneous graph neural networks to address these challenges by effectively capturing the complex dynamics of OATP-ligand interactions. The HIPO-GNN model's improved predictive performance, particularly in F1 score, highlights the benefit of incorporating both ligand and protein structural information. Our analysis of feature importance and dataset quality provides key insights for drug development strategies. This work enhances our ability to predict and understand OATP inhibition mechanisms, contributing to more efficient drug design and improved patient safety through better prediction of drug-drug interactions. Looking ahead,

the synergy between computational models and experimental techniques promises to unlock new frontiers in OATP research. By iteratively refining our models with emerging structural and functional data, we can anticipate more accurate predictions, leading to tailored drug designs that minimize unwanted interactions and optimize therapeutic outcomes.

BIBLIOGRAPHY

[1] Innocent G Asiimwe and Munir Pirmohamed. Drug–drug–gene interactions in cardiovascular medicine. *Pharmacogenomics and Personalized Medicine*, pages 879–911, 2022.

[2] A Kalliokoski and M Niemi. Impact of oatp transporters on pharmacokinetics. *British Journal of Pharmacology*, 158(3):693–705, 2009.

[3] Kenta Yoshida, Cen Guo, and Rucha Sane. Quantitative prediction of oatp-mediated drug-drug interactions with model-based analysis of endogenous biomarker kinetics. *CPT: Pharmacometrics & Systems Pharmacology*, 7(8):517–524, 2018.

[4] Alžběta Türková and Barbara Zdrazil. Current advances in studying clinically relevant transporters of the solute carrier (slc) family by connecting computational modeling and data science. *Computational and structural biotechnology journal*, 17:390–405, 2019.

[5] Akash Khandelwal, Matthew D Krasowski, Erica J Reschly, Michael W Sinz, Peter W Swaan, and Sean Ekins. Machine learning methods and docking for predicting human pregnane x receptor activation. *Chemical research in toxicology*, 21(7):1457–1467, 2008.

[6] Lei Zhang, Yuanchao Zhang, and Shiew-Mei Huang. Scientific and regulatory perspectives on metabolizing enzyme- transporter interplay and its role in drug interactions: challenges in predicting drug interactions. *Molecular pharmaceutics*, 6(6):1766–1774, 2009.

[7] Maria Karlgren, Anna Vildhede, Ulf Norinder, Jacek R Wisniewski, Emi Kimoto, Yurong Lai, Ulf Haglund, and Per Artursson. Classification of inhibitors of hepatic organic anion transporting polypeptides (oatps): influence of protein expression on drug–drug interactions. *Journal of medicinal chemistry*, 55(10):4740–4763, 2012.

[8] Mikko Niemi, Marja K Pasanen, and Pertti J Neuvonen. Organic anion transporting polypeptide 1b1: a genetically polymorphic transporter of major importance for hepatic drug uptake. *Pharmacological reviews*, 63(1):157–181, 2011.

[9] Savannah J McFeely, Tasha K Ritchie, and Isabelle Ragueneau-Majlessi. Variability in in vitro oatp 1b1/1b3 inhibition data: Impact of incubation conditions on variability and subsequent drug interaction predictions. *Clinical and Translational Science*, 13(1):47–52, 2020.

[10] Y Tomita, K Maeda, and Y Sugiyama. Ethnic variability in the plasma exposures of oatp1b1 substrates such as hmg-coa reductase inhibitors: a kinetic consideration of its mechanism. *Clinical Pharmacology & Therapeutics*, 94(1):37–51, 2013.

[11] Mikko Niemi, Janne T Backman, Lauri I Kajosaari, Julian B Leathart, Mikko Neuvonen, Ann K Daly, Michel Eichelbaum, Kari T Kivistö, and Pertti J Neuvonen. Polymorphic organic anion transporting polypeptide 1b1 is a major determinant of repaglinide pharmacokinetics. *Clinical Pharmacology & Therapeutics*, 77(6):468–478, 2005.

[12] Anne T Nies, Mikko Niemi, Oliver Burk, Stefan Winter, Ulrich M Zanger, Bruno Stieger, Matthias Schwab, and Elke Schaeffeler. Genetics is a major determinant of expression of the human hepatic uptake transporter oatp1b1, but not of oatp1b3 and oatp2b1. *Genome medicine*, 5:1–11, 2013.

[13] Nicholas A Meanwell. Improving drug candidates by design: a focus on physicochemical properties as a means of improving compound disposition and safety. *Chemical research in toxicology*, 24(9):1420–1456, 2011.

[14] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

[15] Massoud Saidijam, Fatemeh Karimi Dermani, Sareh Sohrabi, and Simon G Patching. Efflux proteins at the blood–brain barrier: review and bioinformatics analysis. *Xenobiotica*, 48(5):506–532, 2018.

[16] Anca-Denise Ciută, Kamil Nosol, Julia Kowal, Somnath Mukherjee, Ana S Ramírez, Bruno Stieger, Anthony A Kossiakoff, and Kaspar P Locher. Structure of human drug transporters oatp1b1 and oatp1b3. *Nature Communications*, 14(1):5774, 2023.

[17] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, et al. Pubchem 2023 update. *Nucleic acids research*, 51(D1):D1373–D1380, 2023.

[18] Noel M O'Boyle, Michael Banck, Craig A James, Chris Morley, Tim Vandermeersch, and Geoffrey R Hutchison. Open babel: An open chemical toolbox. *Journal of cheminformatics*, 3:1–14, 2011.

[19] Amos Bairoch, Rolf Apweiler, Cathy H. Wu, Winona C. Barker, Brigitte Boeckmann, Serenella Ferro, Elisabeth Gasteiger, Hongzhan Huang, Rodrigo Lopez, Michele Magrane, Maria J. Martin, Darren A. Natale, Claire O'Donovan, Nicole Redaschi, and Lai-Su L. Yeh. The Universal Protein Resource (UniProt). *Nucleic Acids Research*, 33(suppl1):D154–D159, 01 2005.

[20] Ziyang Shan, Xuemei Yang, Huihui Liu, Yafei Yuan, Yuan Xiao, Jing Nan, Wei Zhang, Wenqi Song, Jufang Wang, Feiwen Wei, et al. Cryo-em structures of human organic anion transporting polypeptide oatp1b1. *Cell Research*, 33(12):940–951, 2023.

[21] Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, Talapady N Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. The protein data bank. *Nucleic acids research*, 28(1):235–242, 2000.

[22] Serena Rosignoli and Alessandro Paiardini. Boosting the full potential of pymol with structural

biology plugins. *Biomolecules*, 12(12):1764, 2022.

[23] Patrick Conway, Michael D Tyka, Frank DiMaio, David E Konerding, and David Baker. Relaxation of backbone bond geometry improves protein energy landscape modeling. *Protein science*, 23(1):47–55, 2014.

[24] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

[25] Jeffrey Skolnick, Mu Gao, Hongyi Zhou, and Suresh Singh. Alphafold 2: why it works and its implications for understanding the relationships of protein sequence, structure, and function. *Journal of chemical information and modeling*, 61(10):4827–4831, 2021.

[26] D Sala, F Engelberger, HS Mchaourab, and J Meiler. Modeling conformational states of proteins with alphafold. *Current Opinion in Structural Biology*, 81:102645, 2023.

[27] Jens Meiler and David Baker. Rosettaligand: Protein–small molecule docking with full side-chain flexibility. *Proteins: Structure, Function, and Bioinformatics*, 65(3):538–548, 2006.

[28] Shannon T Smith and Jens Meiler. Assessing multiple score functions in rosetta for drug discovery. *PLoS One*, 15(10):e0240450, 2020.

[29] Greg Landrum. Rdkit documentation. *Release*, 1(1-79):4, 2013.

[30] Melissa F Adasme, Katja L Linnemann, Sarah Naomi Bolz, Florian Kaiser, Sebastian Salentin, V Joachim Haupt, and Michael Schroeder. Plip 2021: Expanding the scope of the protein–ligand interaction profiler to dna and rna. *Nucleic acids research*, 49(W1):W530–W534, 2021.

[31] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. pages 2623–2631, 2019.

[32] Natalia Khuri, Arik A Zur, Matthias B Wittwer, Lawrence Lin, Sook Wah Yee, Andrej Sali, and Kathleen M Giacomini. Computational discovery and experimental validation of inhibitors of the human intestinal transporter oatp2b1. *Journal of chemical information and modeling*, 57(6):1402–1413, 2017.

[33] Tom De Bruyn, Gerard J P van Westen, Adriaan P Ijzerman, Bruno Stieger, Peter de Witte, Patrick F Augustijns, and Pieter P Annaert. Structure-based identification of oatp1b1/3 inhibitors. *Molecular pharmacology*, 83(6):1257—1267, June 2013.

[34] Thomas R. Lane, Fabio Urbina, Xiaohong Zhang, Margret Fye, Jacob Gerlach, Stephen H.

Wright, and Sean Ekins. Machine learning models identify new inhibitors for human oatp1b1. *Molecular Pharmaceutics*, 19(11):4320–4332, 2022. PMID: 36269563.

[35] Hao Duan, Chaofeng Lou, Yaxin Gu, Yimeng Wang, Weihua Li, Guixia Liu, and Yun Tang. In silico prediction of inhibitors for multiple transporters via machine learning methods. *Molecular Informatics*, 43(3):e202300270, 2024.

APPENDIX 4A

LIGAND-PROTEIN DOCKING WORKFLOW

In this study, we leverage protein-ligand docking simulations to inform machine learning models on accurately classifying OATP ligands as either inhibitors or non-inhibitors. The docking workflow involved selecting protein and ligand for docking. Optionally, the CASTp server was used to identify potential binding sites on the protein. The workflow required organizing files in a specific manner and utilizing Python scripts designed for this structure.The initial script prepared necessary files, including a Params file for ligand processing instructions and a concatenated file of the protein and ligand structures. The subsequent script generated the docking job and an XML file with docking parameters. Customizations in the XML file included adjustments to the docking starting coordinates, box size for ligand movement, and the scoring grid for residue analysis. The options file was edited to define file locations for Rosetta, including the concatenated files, Params file, and XML file, along with the docking iterations count. Docking jobs were executed from a designated directory, with outputs stored in a specified output folder. Post-docking, a script was used to extract the top docking results. The approach was designed for high throughput processing of multiple proteins and ligands, ensuring file paths and naming conventions were consistent with the established directory structure. This protocol enabled effective protein-ligand docking using Rosetta for multiple entities.

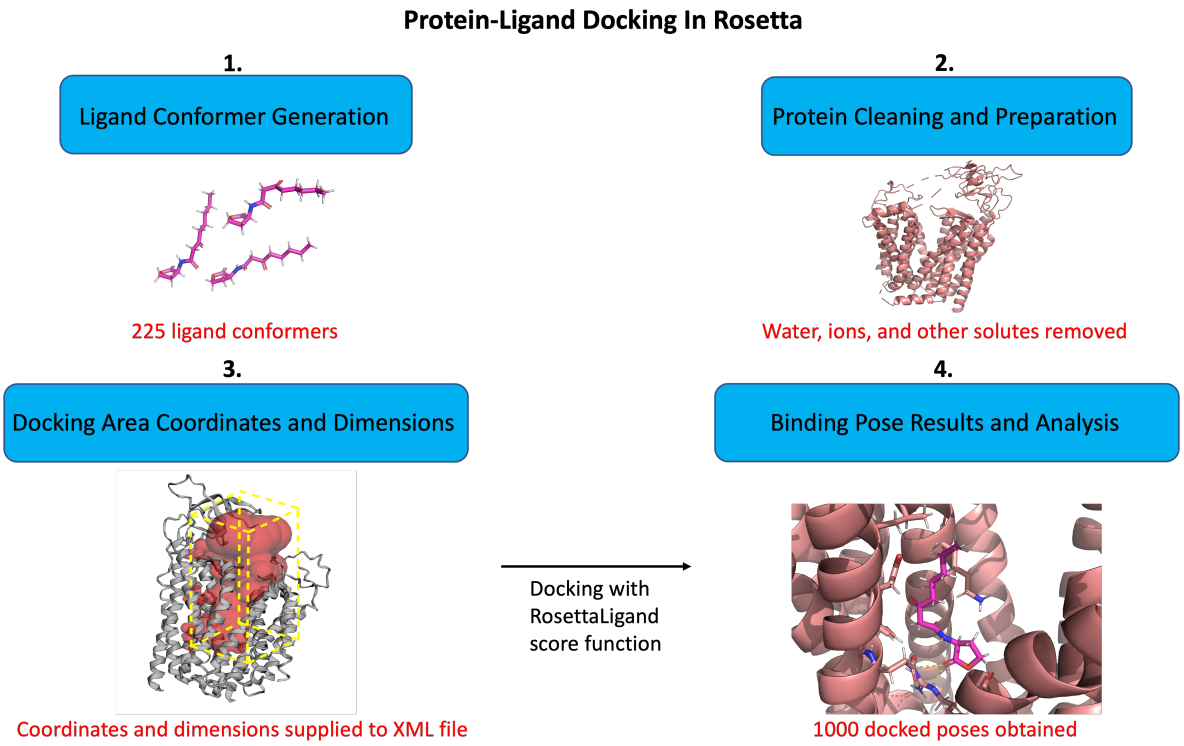


Figure 4A.1 Docking Workflow for Drug Inhibition Prediction.

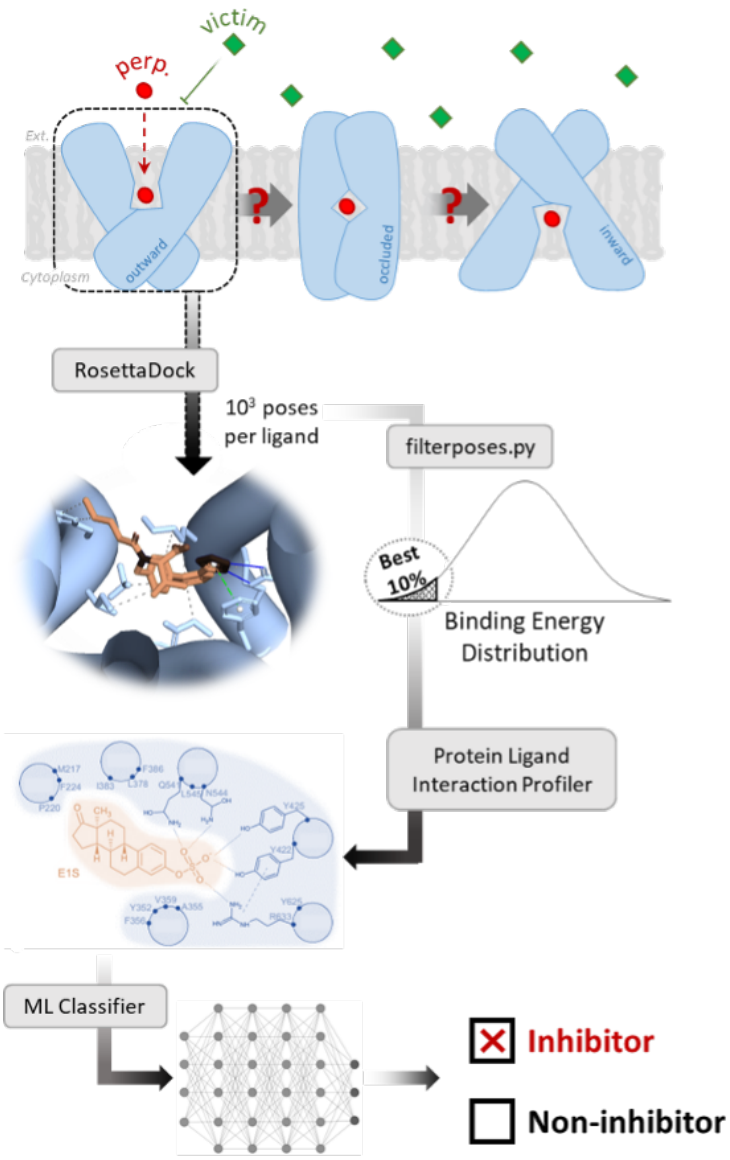


Figure 4A.2 ML Workflow for Drug Inhibition Prediction.

APPENDIX 4B

COMPARATIVE ANALYSIS BASED ON STRUCTURE AND LIGAND PREDICTIONS ACROSS OATP SUBFAMILIES

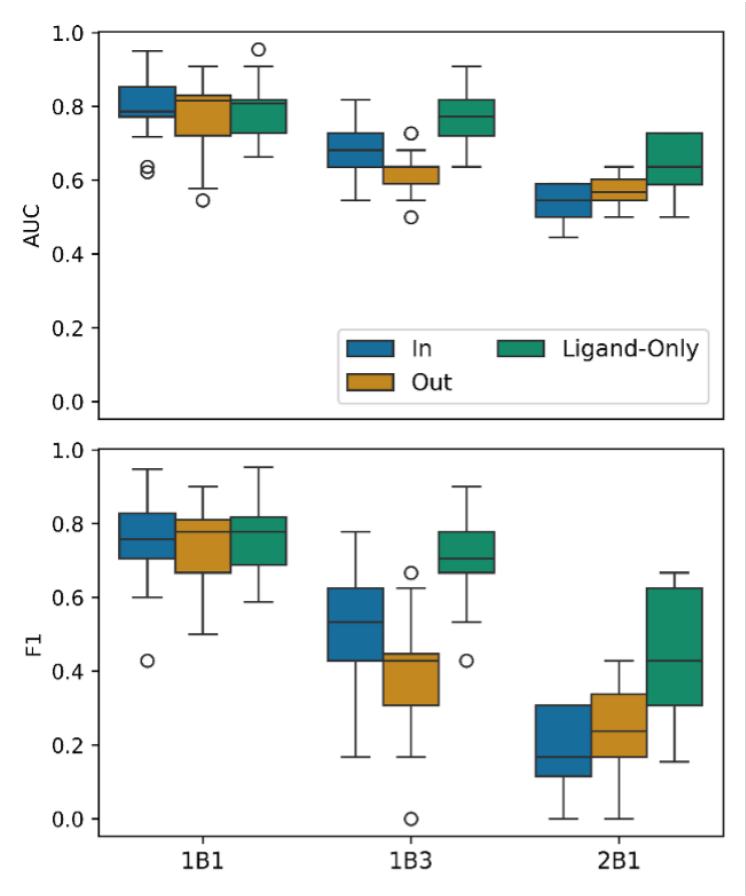


Figure 4B.1 Comparative performance of machine learning models across OATP subfamilies. RDKit-based predictions consistently outperform other models for OATP1B3 and OATP2B1, while OATP1B1 shows superior performance with structure-based modeling approaches.

These results provide valuable insights into the relative effectiveness of structure-based and ligand-based approaches for predicting inhibition across different OATP subfamilies. The analysis reveals distinct patterns in predictive performance, highlighting the importance of tailored methods for each OATP subtype.For OATP1B3 and OATP2B1, ligand information emerges as the key determinant in inhibition prediction. The consistently superior performance of RDKit-based machine learning models for these subtypes underscores the significance of ligand physicochemical properties and molecular descriptors in capturing the essence of their interactions with these

transporters. In contrast, OATP1B1 exhibits a unique behavior, demonstrating enhanced predictive performance when structure-based methods are employed. This suggests that the availability of high-quality structural data for OATP1B1 provides crucial insights into the protein-ligand interaction dynamics that are not fully captured by ligand-based approaches alone. Notably, the analysis reveals that among the tested OATP families in the Karlgren dataset, OATP2B1 posed the most significant challenge for machine learning prediction. This observation highlights the complex nature of OATP2B1 interactions and suggests that further refinement of predictive models or the incorporation of additional data types may be necessary to improve accuracy for this subtype. The comparative performance across OATP subfamilies, as illustrated in Figure 4B.1, provides a clear visualization of these trends. The consistent outperformance of RDKit-based models for OATP1B3 and OATP2B1 is evident, while the superior performance of structure-based modeling for OATP1B1 is distinctly visible. Based on these findings, the study's focus naturally gravitated towards OATP1B1, leveraging its amenability to structure-based prediction methods. This strategic shift paved the way for the development and implementation of novel approaches aimed at further enhancing the predictive performance for OATP1B1 inhibition. These results not only provide valuable insights into the differential predictive requirements of OATP subfamilies but also underscore the importance of tailoring computational approaches to the specific characteristics of each transporter. Such nuanced understanding is crucial for advancing our ability to accurately predict OATP-mediated drug interactions and ultimately contribute to more effective and safer drug development processes.

APPENDIX 4C

DISTRIBUTION OF ROSETTA INTERFACE BINDING ENERGIES FOR OATP TRANSPORTERS

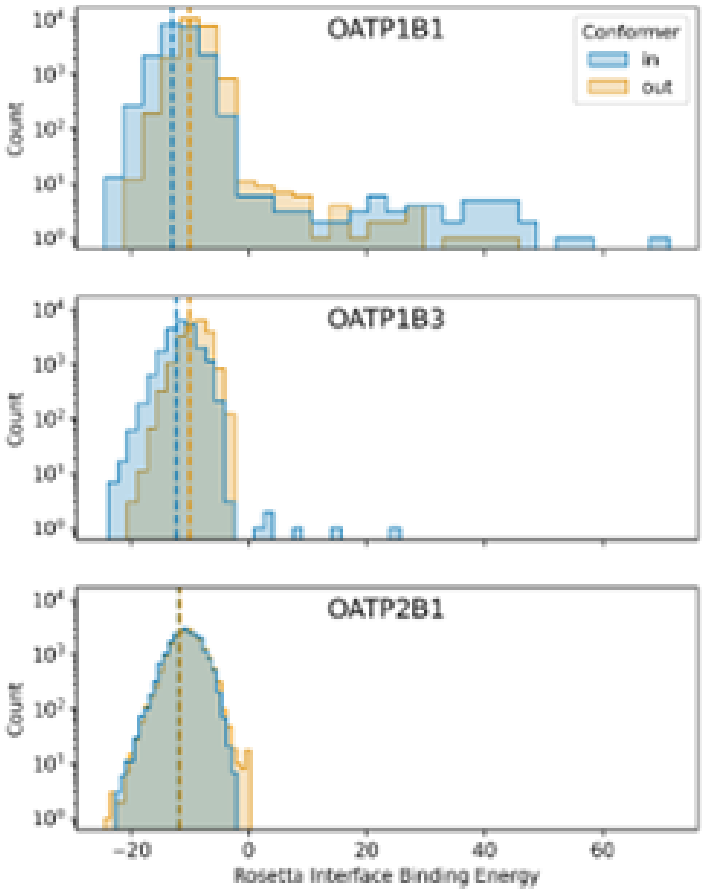


Figure 4C.1 Distribution of Rosetta interface binding energies for OATP1B1, OATP1B3, and OATP2B1 transporters. The histograms show the binding energies for the inward (blue) and outward (orange) conformers. The x-axis represents the Rosetta interface binding energy, while the y-axis represents the count of interactions. Data indicate a broader distribution for the inward conformer of OATP1B1, a noticeable difference between the conformers of OATP1B3, and a symmetric distribution for OATP2B1, suggesting diverse binding affinities and interaction patterns. Comparison with AlphaFold-generated structures reveals more diversity and wider distributions in REU with experimentally solved structures.

APPENDIX 4D

DISTRIBUTION OF OATP1B1 RESIDUE INTERACTIONS WITH INHIBITORS AND NONINHIBITORS

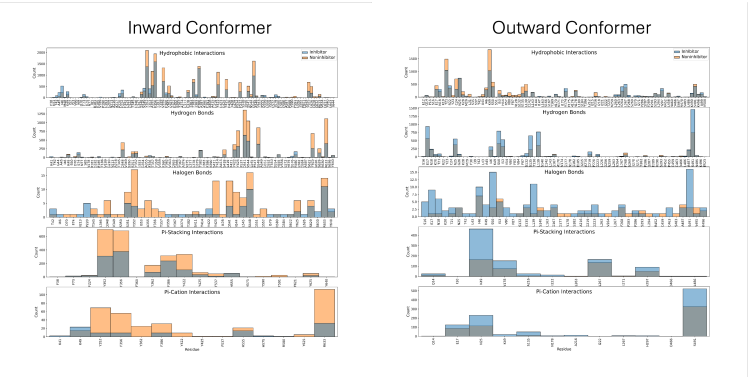


Figure 4D.1 Distribution of OATP1B1 residue involvement in interactions with inhibitors (blue) and noninhibitors (orange), as computed from PLIFs of the 'best' 30 poses for each OATP conformer-ligand pair. Distributions presented are per interaction type: hydrophobic, hydrogen bonding, halogen bonding, pi-stacking, and pi-cation interactions (A – E). Inward-facing conformer interactions (left) are notably more frequent with noninhibitors, whereas the outward-facing conformer (right) is observed to have a greater frequency of inhibitor interactions. Especially notable, known noninhibitors were observed to have roughly twice more pi-stacking interactions with each Y352 and F356 in the inward conformer when compared to the known inhibitors in the dataset. This agrees with findings of Shan et al. (2023) [20]regarding the importance of Y352 and F356 in binding and transport of known OATP1B1 substrates.

CHAPTER 5

GENERATIVE MODELS FOR PROTEIN SEQUENCE MODELING: RECENT ADVANCES AND FUTURE DIRECTIONS

[1]

Abstract

The widespread adoption of high-throughput omics technologies has exponentially increased the amount of protein sequence data involved in many salient disease pathways and their respective therapeutics and diagnostics. Despite the availability of large-scale sequence data, the lack of experimental fitness annotations underpins the need for self-supervised and unsupervised machine learning (ML) methods. These techniques leverage the meaningful features encoded in abundant unlabeled sequences to accomplish complex protein engineering tasks. Proficiency in the rapidly evolving fields of protein engineering and generative AI is required to realize the full potential of ML models as a tool for protein fitness landscape navigation. Here, we support this work by (i) providing an overview of the architecture and mathematical details of the most successful ML models applicable to sequence data (e.g. variational autoencoders, autoregressive models, generative adversarial neural networks, and diffusion models), (ii) guiding how to effectively implement these models on protein sequence data to predict fitness or generate high-fitness sequences and (iii) highlighting several successful studies that implement these techniques in protein engineering (from paratope regions and subcellular localization prediction to high-fitness sequences and protein design rules generation). By providing a comprehensive survey of model details, novel architecture developments, comparisons of model applications, and current challenges, this study intends to provide structured guidance and robust framework for delivering a prospective outlook in the ML-driven protein engineering field.

Keywords: Generative Machine Learning (ML) Models, Protein Engineering, Generative Adversarial Neural Networks (GANs), Variational Autoencoders (VAE), NLP, Diffusion Models

[1]This chapter is adapted from content published in "Briefings in Bioinformatics," Oxford Academia. All rights reserved. For more information, visit https://doi.org/10.1093/bib/bbad358.

5.1 Introduction

Proteins, as genetically encoded macromolecules, play a pivotal role in regulating biological systems. Their diverse sizes and chemical compositions empower them with a wide range of functionalities. Consequently, engineered proteins with tailored functions find applications across various fields, including cosmetics and environmental bioremediation. Engineered proteins can be optimized to target disease biomarkers for the early detection of cancer, Alzheimer's and inflammatory diseases [1, 2, 3, 4, 5]. Protein-based therapeutics is also another significant application; namely serum therapy with therapeutic antibodies which started more than a century ago and evolved with scientific advancements. In addition, protein engineering is a practical tool to address environmental issues resulting from industrialization [6, 7, 8, 9]. For example, heavy metal protein binders displayed on the bacterial surface are capable of remediating environments contaminated with heavy metals [7, 10]. Despite the promise of protein engineering to revolutionize medicine and industry, discovering proteins with desired functionalities is exceedingly challenging. Within the astronomical number of ways to build a protein (i.e. unique protein sequences), the vast majority lack function entirely. Moreover, making a random change to a functional protein is typically detrimental to its function and stability. This highlights a need for improved strategies in obtaining novel proteins with favorable properties such as high binding affinity and desired developability [11, 12]. While previous strategies such as energy-based scoring [13] and evolutionary [14] methods are still informative, they have drawbacks, including inaccurate modeling and search strategy inefficiency. Recent advancements in computational methods and the rapidly growing availability of protein sequence data facilitated the use of new, data-driven approaches for protein design and engineering [15, 16].

ML techniques may offer a promising route for navigating the high-dimensional landscape of protein design and engineering. They have shown a high success rate in various domains such as processing text, images and audio. In theory-driven approaches, the researcher obtains the domain knowledge of the problem that needs to be solved and produces mathematical models to capture the attributes and physics of the study. In contrast, ML methods are mainly centered

on modeling the observed data while previous knowledge and theory may also be infused to the model. To assess how well a dataset is modeled via ML, a loss function is defined that measures the difference between the model's prediction and real data. It then is optimized so that the prediction is close to reality (i.e. the difference between the model and actual data is minimized). Training a model involves fitting the model parameters to optimize the loss function. A trained ML model, therefore, is useful in understanding the given data and aiding in future decisions by identifying trends, predicting outcomes and recognizing anomalies. Machine learning models can be classified into two types of models—discriminative and generative. Discriminative models are ML models used to predict the conditional probability of labels based on the given data features. In contrast, generative models aim to discover how the data is constructed by estimating the joint probability distributions of features and corresponding labels. Deep learning (DL), a subset of ML that uses neural networks for the training, is particularly well-suited for complex domains since it can extract high-level features (i.e. features that cannot be interpreted by humans) from the given dataset [17].

Sparse high-fitness variants can be efficiently sampled from the vast, rugged protein fitness space landscape using DL techniques that implement statistical and probabilistic models [18]. For sequence-function mapping, protein sequences can be vectorized (e.g. with one-hot encodings or embeddings) to get proper input representations that are compatible with ML algorithms. Therefore, due to both high demand and compatibility, various ML models have been applied to predict protein binding affinity [19], thermostability [20], developability [12], solubility [21] and stability [22, 23]. ML-driven predictive models have shown remarkable success in various applications compared to wet lab methods like directed evolution and traditional computational methods like rational design. However, the incorporation of natural language processing (NLP) techniques and generative models in protein engineering has led to a revolution in the field by improving prediction accuracy, reducing data requirements and enabling the generation of novel and functional proteins. Exploiting NLP techniques have been made feasible by changing the perspective about proteins and finding similarities between human language and protein sequences. For example, both feature an alphabet (20 amino acids in terms of proteins), have hierarchy in the organization, and evolve over

time [24]. TAPE [25], UniRep [26], ESM [27] and ProtTrans [28] are among the many successful studies that have applied NLP techniques to learn the dependencies in protein sequences and be able to numerically represent them in fixed vector formats (i.e. embeddings) that are rich in semantics and syntax information. Generative models (based on NLP or pure statistical methods) are also used in different tasks such as improved representation of protein sequences and generation of unforeseen protein sequences in nature. BioSeqVAE [29], ProtGPT2 [30], ProteinGAN [31] and RFdiffusion [31] are examples that generated de novo proteins via different model architectures such as variational autoencoder, transformer, adversarial neural networks and diffusion models, respectively.

Here, we provide a systematic overview of promising neural networks applicable to protein sequences. For each architecture, we introduce core mathematical details of the model before describing the implementation of these models towards protein engineering tasks. For each model type, we also provide commentary through selected case studies to describe practical considerations for integrating ML towards protein applications and to illustrate how specific model architectures can be advantageous for a given protein engineering task. The first type to be discussed is language models which have several features that make them strong candidates to be applied for protein sequence data. These techniques can handle variable sequence lengths, and they can track sequence long-term dependencies while maintaining the order of tokens (e.g. amino acid positions). We start with recurrent neural networks (RNNs), then introduce self-attention mechanism (when the model learns to selectively focus on important positions of the input sequence), and finally dive into the transformers and their variants. After discussing transformers, we examine three other generative models in detail: variational autoencoders (VAEs) [32], generative adversarial neural networks (GANs) [33] and diffusion models [34]. While VAE takes a probabilistic approach to learn the training data distribution, GANs use two competing neural networks to generate realistic samples. Diffusion models take a different approach by progressively adding noise to the data until it reaches the prior distribution. Then, new samples can be generated in the reverse diffusion process. Finally, we discuss the current challenges and future perspectives in applying such models

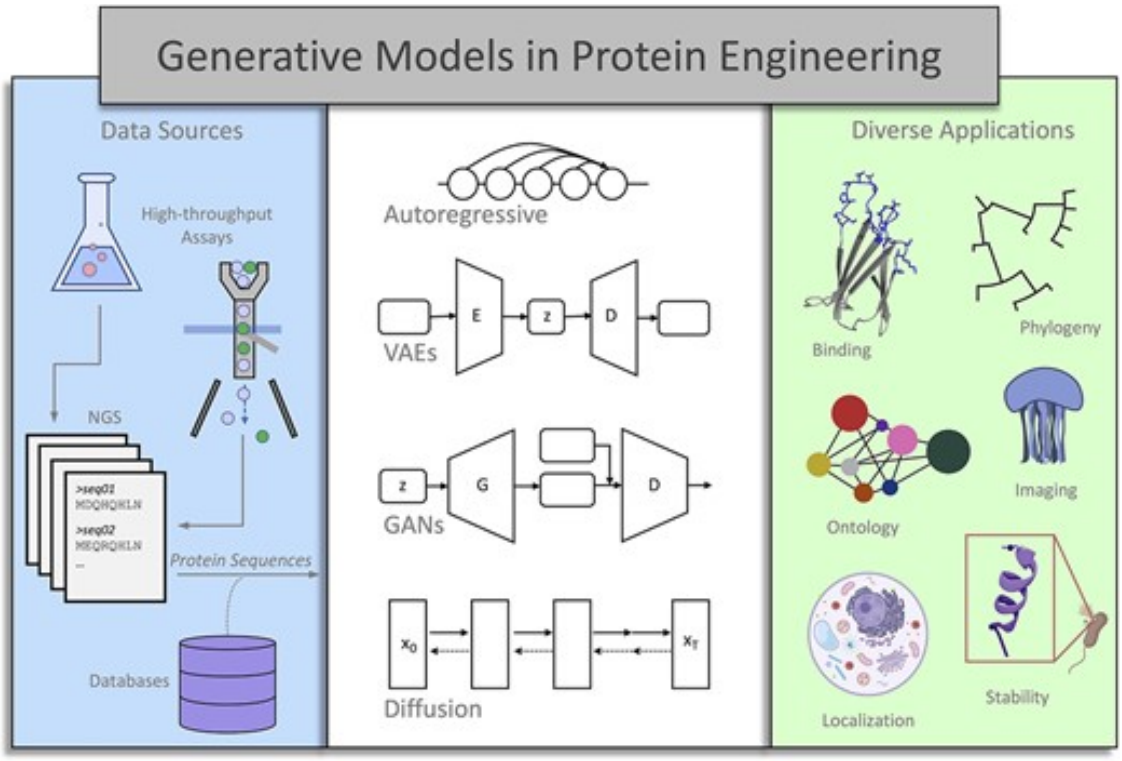


Figure 5.1 A diverse set of protein engineering applications benefit from the generative and discriminative potential of sequence models. These applications include stability, solubility, bioluminescence, binding capacity, phylogeny, gene ontology and protein localization. The schematic of the sequence models are represented here but their detailed description will be elaborated in their corresponding sections. The autoregressive model forecasts future values based on the previous values in the time series data. VAE is probabilistic modeling architecture that contains an encoder (E) and a decoder (D), compressing high-dimensional input data with the E and reconstructing the data from hidden dimension by D. This architecture along with variational inference techniques will lead to learning given data distribution and generating novel instances. In GANs, G represents a generative model that aims to generate realistic data from noise input, and D is a discriminator that acts as a critic to distinguish the real data from model-generated data. Diffusion models are a relatively new generative model that facilitates the generation of novel samples from a state of maximum randomness (at point XT) that is previously generated through the iterative addition of random noise to data distribution. These models have been demonstrated in diverse applications ranging from antibody binding prediction to protein localization prediction tasks in addition to novel protein sequence generation tasks.

to protein engineering. Figure 1 represents the overview of models and applications that are discussed in detail in this study. With this information, we hope researchers are better prepared to integrate these technologies into future investigations.

5.2 Sequence Forecasting – RNNs & ARs

Given that proteins consist of linear chains of discrete amino acids, the amino acid sequences can be treated as time-series data, with each amino acid acting as a discrete data point for each position (i.e. time point). Because of this attribute, models that forecast future values using past values can be used to sequentially generate amino acids based on previous amino acids in the sequence chain. In this section, we discuss two major models for sequence forecasting: recurrent neural networks (RNNs) and autoregressive models (ARs).

RNNs are a popular architecture in natural language processing and speech recognition because they hold 'memory' by having internal weights that store information from the past that can be updated with every new token processed. There are different types of RNNs including one-to-one, one-to-many, many-to-one, and many-to-many [35]. For protein sequence models, one-to-many RNNs are suitable for sequence forecasting. By giving a starting token to initialize RNNs, the trained model can sequentially produce tokens at the current time step using the output token from the previous time step (Figure 2A). Despite being an architecture well suited for sequential data, some disadvantages of RNNs need to be addressed. During prediction tasks on a given token, RNNs are not able to learn any effects of subsequent parts of the sequence because RNNs process sequences unidirectionally (usually from left to right). This could hurt predictive power because the interactions between amino acids within a protein occur in a three-dimensional space. Bidirectional RNNs (BRNNs) address the issue of unidirectionality and improve prediction performance using two connected layers: one-layer processes sequences in the forward direction and another layer processes sequences in the backward direction [36]. In this way, sequence information is learned from both directions. BRNNs are also successfully applied to unsupervised tasks by enabling their probabilistic interpretation to reconstruct the missing value [37].

Optimizing RNNs while tracking long-term dependencies within sequences can be challenging. Due to the repeated weight matrix multiplication, the weights from early parts of the sequence have a progressively lower influence on the final representation relative to the later parts. The repeated multiplication of small weights results in even smaller weights, causing the gradient vanishing

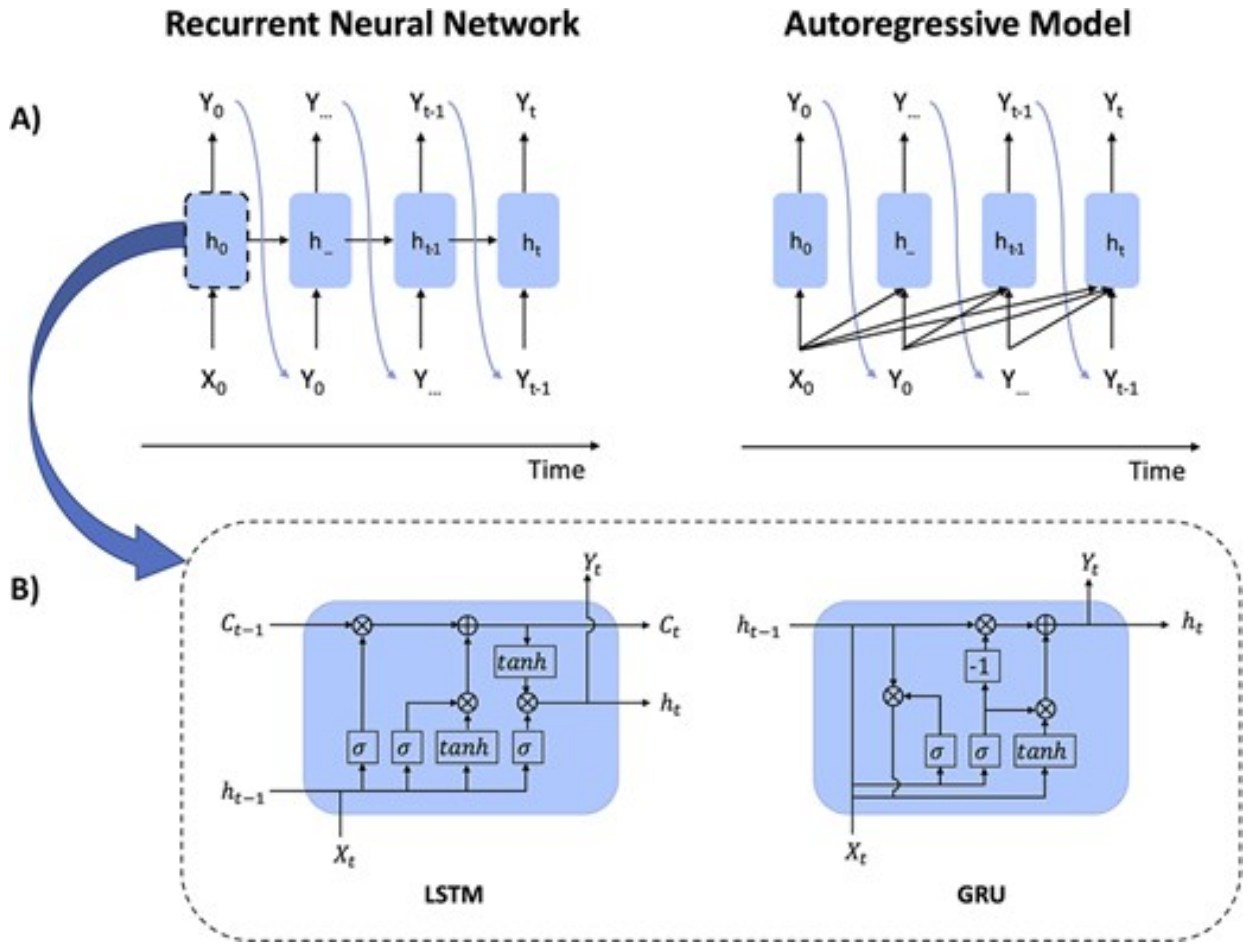


Figure 5.2 The architecture of generative recurrent neural networks versus autoregressive model. (A) An autoregressive model (AR) has a similar structure to a recurrent neural network (RNN). However, while RNN only depends on the current time step, AR utilizes information from the previous time steps as well as the current time step to predict the next token. (B) Two important RNN architectures for resolving vanishing gradient problems in training sequence data are LSTM and GRU. These networks contain gates to control the information flow. LSTM contains three gates: input gate, forget gate and output gate and GRU contains two gates: reset gate and update gate. Note that C indicates the cell state, and h is the hidden state in shown architectures.

problem in which the gradient eventually reaches a drastically small value. Therefore, RNNs tend to generate biased parameters that capture short-term dependencies, especially when dealing with long sequences. To enable long-term memory, Hochreiter and Schmidhuber introduced Long Short-Term Memory (LSTM) [37], and Chung and colleagues introduced Gated Recurrent Units (GRU) [38] (Figure 2B). Both networks have a gated cell that not only contains multiplication but also addition operations of sigmoid and hyperbolic tangent (tanh) functions to regulate the information flow. The sigmoid activation function—which outputs values between 0 and 1—serves

as a gate to keep relevant and discard irrelevant information. The information is pertinent for prediction when values are close to 1, and it is kept to future time steps. The tanh activation function regulates values to prevent vanishing and exploding gradients by limiting outputs between 1 and 1.

RNN and its variants have been implemented in protein design tasks for both generative and discriminative applications. For example, a generative LSTM-based model was trained to design de novo antimicrobial peptide [39]. The generated sequences elicited higher microbial activity than sampling randomly mutated peptides. In another study, LSTM units were implemented to generate antibody sequences with well-correlated negative log-likelihood and more than 100-fold affinity maturation [40]. Interestingly, an LSTM model trained on only f and y angles of each residue enabled helical protein design [41]. The authors demonstrated that dihedral angles are adequate features to design protein backbones without considering amino acids in a sequence. In another study for predicting protein secondary structures, bidirectional recurrent neural networks with GRU units were used to capture global features within sequence [42].

ARs share a similar structure with RNNs (Figure 2A). Both outputs at time t depend on input not only at time t but also from earlier time steps. However, RNNs use the hidden state weights from only the most recent time step, whereas ARs use actual inputs from the past to generate future values. AR models, as the name suggests, perform regression tasks over their own lagged variables (i.e. forecasting future values using linear combinations of past values). Protein sequence generation through ARs can be achieved by maximizing sequence likelihood through a tractable probability density function. This objective function is a product of conditional probabilities of tokens at each position that are conditioned on all previous tokens shown as Equation 1, where X is the full-length sequence, x is each token, I is the position number, and n is the sequence length. The objective function is decomposed from the joint probability of a full-length sequence using the chain rule of probability and Bayes' theorem. For each step generation, the features are past tokens, the label is the true token at the current time step, and the loss is the difference between the predicted token and the true token.

$$p(\mathbf{X}) = \prod_{i=1}^{n} p(x_i|x_1, x_2, \ldots, x_{i-1}) \tag{5.1}$$

Theoretically, the space complexity of ARs grows exponentially with forward processing of sequences. This complexity is represented as O(nk) in big O notation where k grows with increasing n for a sequence with length of n. In practice, ARs use a fixed number of parameters to specify each prior. This reduces the complexity to a polynomial O(nc) where c is a constant. However, this restricts ARs to represent all possible conditional distributions and limits model expressiveness. Lin et al. [43] proposed energy-based models and latent-variable autoregressive models as alternatives to alleviate limited distributional modeling of standard ARs.

Recent studies have implemented ARs for protein design. A model with one autoregressive layer paired with generalized logistic regression was used for mutational prediction, contact prediction, and sequence generation of a response-regulator protein family [44]. The negative log-ratio of joint probability of mutant and wildtype were used to indicate single mutational effects and the sum of log probabilities of single mutations were employed as double mutation likelihood for residue-residue contact prediction. The trained model generated sequences that were similar to natural sequences by comparing their principal components. Another study used a dilated convolutional and autoregressive model to model sequential constraints of long nanobodies [45]. They showed that their alignment-free model matches the accuracies of alignment-dependent models in the context of mutation effect prediction, thermostability prediction and fitness predictions for indels. In addition, their model yielded a designed library that contained stable and functional nanobodies with comparable biochemical properties and enhanced diversity to natural nanobody repertoire.

Though RNNs and ARs are powerful, their sequential operation results in linear-time O(n) complexity that makes their training time-consuming. Both RNNs and ARs employ supervised learning which helps models optimize with experience and yield high accuracy. However, it increases the chance to overfitting models if the training data is not well-representing the true data distribution. The LSTM and GRU address some limitations of RNNs, but RNNs are still inefficient

due to short-term memory and long gradient path. ARs have explicit probability density function to maximize sequence likelihood, but the computation of a series of conditional probabilities requires significant computational resources. The loss of information during training also hinders the overall performance of RNNs and ARs. These issues are significant when considering whether RNNs andARs should be implemented for sequence generation tasks.

5.2.1 Protein Engineering Highlights of RNNs and ARs

A comprehensive collection of notable applications of sequence forecasting models in protein engineering is provided in Table 1 below. Following this, a case study is presented to elucidate the details of a selected paper marked in the table.

Table 5.1 Summary of highlighted applications of sequence forecasting models for protein engineering.

Protein Engineering Task	**Advancements**	**Model Type**	**Training Data Source(s)**	**Year Ref.**
Classification of sequences based on predicted iron sequestration capabilities, protease activity, GPCR activity and p450 activity	Ten diverse, unannotated sequences predicted to exhibit iron sequestration activity were experimentally validated. As measured by accuracy, precision, recall and F1 scores, the RNN model outperforms logistic regression and random forest models.	RNN	UniProt	Jan. 2017 [46]
Predict solvent accessibility	8.8% mean absolute error and 74.8% Pearson's correlation coefficient value for predicting solvent accessibility were observed.	Stacked Bidirectional LSTMs	PISCES Database	May 2018 [42]
Structural class prediction	Using a low-dimension feature space (18-D), classification accuracies ranging from 84.2% to 95.9% were observed.	RNN	PDB25, FC699, 640, 498, 277 and 204	June 2021 [47]
Novel artificial protein sequence generation	Libraries of artificially generated monobodies mutases were experimentally validated	DCA	1259 natural monobodies mutase sequences	July 2020 [48]
Antibody binding pocket prediction	Outperformed CNN/RNN models and random-forest models	Bidirectional LSTMs	Structural Antibody Database (SabDab)	Dec. 2021 [49]

5.2.1.1 Prediction of Antibody Paratope with Bidirectional LSTMs

A deep learning model for predicting the antigen binding sites of antibodies was developed through the implementation of bidirectional LSTMs in a transformer neural network (discussed in Section Sequence Design with Attention Mechanism: Transformer-Based Language Models)

[49]. The DeepANIS (Antibody Interacting Site) model was able to elucidate the relationships among residues of the loop sequences of complementarity determining regions (CDRs) of a given antibody (https://github.com/HideInDust/DeepANIS). Trained on only 277 antibody–antigen complexes from the Protein Sequence Culling Server (PISCES) database, the authors demonstrated the ability of a transformer neural network using bidirectional LSTMs to outperform alternative CNN-based and random-forest-based models for paratope prediction. This architecture also enabled the developers to perform these predictions using the concatenated CDR loop sequences of a given antibody as the only input. Alternative models require either the CDRs to be provided as separate sequences for each CDR loop or additional information about the antigen or antibody.

5.3 Sequence design with attention mechanism: Transformer-based language models

Transformer models consist of a specific neural network architecture that transforms the input sequences to output sequences using a series of operations (e.g. matrix multiplications, scaled dot product attention and feed forward neural network). Transformers have given rise to various sequence-to-sequence models such as machine translation, question answering (chat bot) and text summarization. Their specific design enables parallel operation (constant-time O(1) complexity), resulting in faster and more efficient performance than ARs and RNNs (linear-time O(n) complexity). This parallelization improves uniform learning across each position of a sequence by eliminating short-term dependencies that disproportionately weigh later parts of the sequence compared to earlier parts.

The original transformer introduced by Vaswani et al. [15] consists of an 'encoder' that encodes a complete sentence to a representation and a 'decoder' that decodes a target sentence with the contextual representation (Figure 2). Both the encoder and decoder contain multiple self-attention and feed forward neural network units. Self-attention is a key component in transformers that enables the model to know which tokens are important in processing the given token (e.g. epistatic interaction in protein sequences). The feed forward networks are then used for adding non-linear operations to the network in training. Note that the order information of tokens gets lost due to parallel computing. Thus, transformers have an additional embedding called positional encoding,

which encodes the exact position of tokens using as many sinusoidal functions as embedding dimensions.

To train a transformer to translate text from language A to language B, the encoder uses language A as an input to generate a representation. The decoder uses language B as an input and combines this with encoder-generated representation to learn the correct mapping of words in two distinct languages. Similarly, protein-specific de novo drug design can be treated as a translational problem [50]. The authors used a transformer as a biological translator to generate novel molecule binders given amino acid sequences only. For a question-answer task, the encoder input is the question, and the decoder input is the answer. Through a connected encoder and decoder, the transformer learns to give an answer based on a specified question. Protein–protein interactions are analogous to question-answer pairs in syntax. This strategy was utilized to generate signal peptides via available organism data in Swiss-Prot [51]. Their experimental results showed that the generated peptides are functional and diverse.

As noted in Figure 3, transformers capture the influence of other tokens (e.g. through epistasis) on the query token with a self-attention mechanism. The self-attention computation starts with three inputs: query, key, and value; analogous to those in retrieval systems. When retrieving an item, the machine takes a request (query) against a list of descriptions of items (keys) and returns top matches (values). In protein chains, we retrieve attention from a sequence first by having queries (amino acid requests) multiplied with transposed keys (amino acid identities) to obtain attention weighting. Then, the scaled and normalized attention weighting is multiplied with values (amino acid representations) to obtain attention (Figure 3). Often, the attention layer is split into several heads in parallel to capture attention from different subspaces. The multi-head attention layer combined with a fully connected feed-forward network with layer normalization in between builds an attention block; these blocks are then combined to form the encoder. Since the inputs (query, key and value) of the encoder are from the same sequence, it generates a self-attentive representation of that sequence. The attention block of the decoder has an additional layer: masked multi-head attention layer, which is placed before the multi-head attention layer. The inputs of the masked

attention layer are from the decoder, meaning that it is self-attentive. The inputs of the following attention layer are from both the encoder and decoder (query from decoder; key and value from encoder), meaning that it is cross-attentive. Due to this self- and cross-attention mechanism, the decoder generates a target sequence considering both the encoder and the decoder.

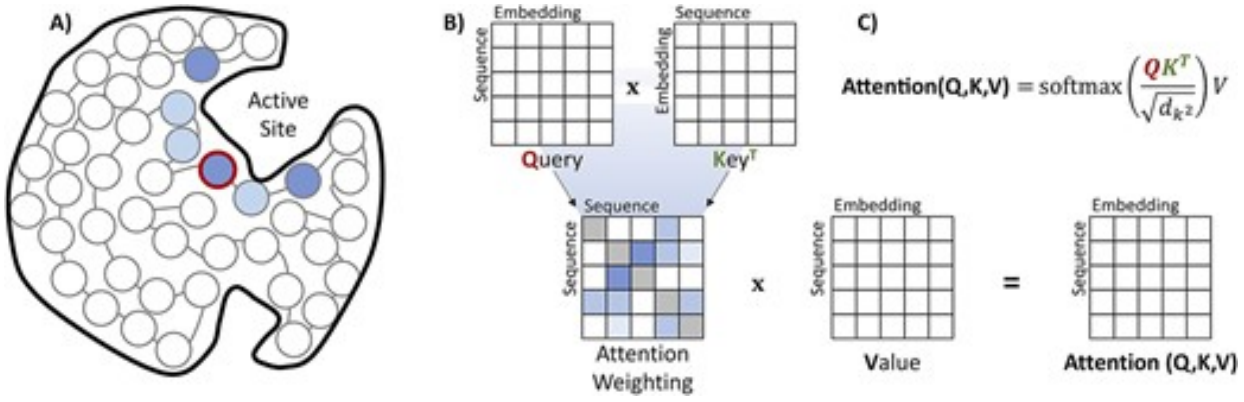


Figure 5.3 Visualization of attention mapping and attention computation. (A) Based on the protein fold, amino acids in different positions have varied epistitatic effects on each other. The highlighted circle refers to a query amino acid in a protein active site. The color gradient shows how attention can capture the influence of other amino acids (tokens) on the queried token. (B) Attention computation requires three components: key, query and value. By calculating scaled dot-product attention scores, the model chooses which areas of the sequence it needs to prioritize for the prediction task.

Attention mechanism has been applied to understand the semantics and syntax of protein language. It has been implemented between sets of gene ontology terms to predict protein–protein interactions [52]. This mechanism has also been employed with a convolutional neural network (CNN) to predict protein contact [53, 54]. Similarly, in another report, CNN with attention mechanism improved protein-drug interaction prediction [55]. CNN was also used to obtain feature metrics of the proteins and ligands. Attention mechanism was then implemented to assign weights to each atom or amino acid. Their model evaluation of benchmark datasets showed improvements compared to previous baselines.

The transformer decoder is autoregressive by nature, owing to its masked self-attention layer. By masking out the attention of future tokens, the decoder decodes target sequences to infer the attention of past tokens. This is achieved through the addition of attention weighting and a mask matrix, whose upper triangular is filled with negative infinity and lower triangular is filled with

zeros. While the decoder is autoregressive during testing, it is non-autoregressive at training time. During training, the decoder generates tokens at all time steps simultaneously, not relying on tokens at previous time steps. The autoregressive attribute of the decoder allows transformers to be used in generative applications. Figure 4 represents the overall schematic of transformer architecture and its variants.

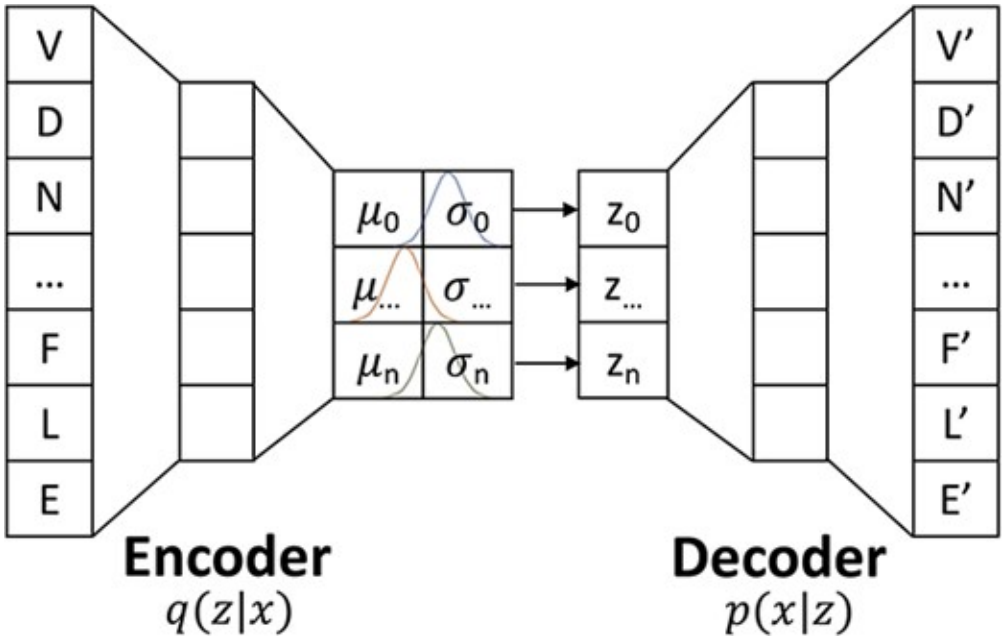


Figure 5.4 Architecture overview for the transformer and two important transformer-based language models: Bidirectional Encoder Representation from Transformers (BERT) and Generative Pre-Training (GPT). The transformer utilizes an encoder-decoder method for handling language tasks. However, BERT uses encoder blocks only and GPT only includes decoder blocks. The difference in their architecture is mainly due to their training objective. In pretraining, BERT takes a bidirectional approach while GPT is based on an autoregressive method.

The ability of transformers to generate text representations has led to the development of various transformer-variant models. Bidirectional Encoder Representation from Transformers (BERT) and Generative Pre-Training (GPT) [56, 57]. They can generate meaningful representations which can be used for downstream, task-specific modeling (e.g. named entity recognition, question-answering and text generation). Pre-training models with a large corpus (unsupervised training) followed by fine-tuning with task-specific objectives (supervised training) result in improved performance in different language modeling tasks. Note that BERT and GPT also have shown good performance

in few-shot (i.e. when there are only few labeled data are available) and zero-shot (i.e. when the model generalizes to the new task with no further data for training) settings [57, 58, 59]. The architecture of BERT is composed of layers of original transformer encoders. The pre-training of BERT uses an approach prevalent in masked language modeling (MLM). By masking out tokens in an input sentence, the models are trained to predict masked tokens using their context. This is achieved by minimizing the cross-entropy loss between masked and actual tokens. This context-dependent training makes representation bidirectional, which is why BERT is a popular architecture for representation learning. In contrast, GPT comprises a series of transformer decoder architecture without the multi-head cross-attention layer due to the absence of encoders. In contrast to BERT's MLM, GPT uses a casual language modeling (CLM) approach that predicts masked tokens, only considering tokens on the left side. By left-shifting each token in input sequences, GPT does not have access to the actual token that is going to predict. Therefore, the representation generated from GPT pre-training is unidirectional and self-attentive, making it a popular model for text generation. By having the task layer directly working on pre-trained representation, the number of layers, learnable parameters and training time are reduced. The training of task layer occurs simultaneously with the fine-tuning of pre-trained models to improve the compatibility between a representation and a given task. Based on the architecture of task layers, they handle either sequence-to-sequence (sequence generation) or sequence-to-scalar (sequence classification) tasks.

Generally, BERT is not optimal for text generation, and GPT is limited to only unidirectional interactions. Lewis et al. [60] proposed Bidirectional AutoRegressive Transformers (BART), which combines the strengths of BERT and GPT to perform sequence-to-sequence denoising effectively, ensuring it fits well within the margins. The BART encoder learns from corrupted sequences that introduce noises to the model through masking, insertion, deletion, infilling, permutation, and rotation. The BART decoder learns to reconstruct original sequences autoregressively. The encoder and decoder work together to recognize and remove intentionally added noise. Hence, BART is a useful architecture for sequence noise reduction and feature extraction. Another challenge in training is capturing long-term dependencies for sequence data whose length is much greater than

its embedding dimension. The BART-derived Performer model was proposed to reduce the cost of training the attention mechanism which scales linearly instead of quadratically with sequence length [61]. This model presents an unbiased estimation of a regular attention matrix with which the estimation is uniformly convergent.

5.3.1 Protein Engineering Highlights of Transformer-based Language Models

Table 2 below presents a wide range of impactful applications of transformer-based language models in protein engineering. Following this table, two case studies selected from the table are discussed.

5.3.2 Pre-training of Deep Bidirectional Protein Sequence Representations with Structural Information

The pre-training scheme PLUS was able to outperform leading pre-training models that are based solely on language models (e.g. UniRep, P-ELMo) by integrating protein-specific structural information with amino acid sequence data (https://github.com/mswzeus/PLUS) [65]. Structural information was obtained from protein family labels among the Pfam dataset. This provided a more accurate and less computationally intensive route compared with using sequence similarity to predict protein function. Additionally, masked language modeling is performed in a similar manner used in BERT to extract syntactic and semantic information. In this study, an informative comparison was made wherein PLUS was used to pre-train bidirectional RNN (PLUS-RNN) and Transformer (PLUS-TFM) architectures. Despite much of the literature indicating that attention-based models are superior, the PLUS-RNN architecture was found to be advantageous over the PLUS-TFM in this study due to the RNN-based implementation more accurately capturing local amino acid sequence motifs. For PLUS-RNN, bidirectional representations of amino acid sequences were used to capture context in both the right-to-left and left-to-right directions. In doing so, the PLUS-RNN model achieved higher performance than similarly sized transformer in protein-level classification and regression tasks and amino acid-level classification tasks. Higher performance was observed even against the leading task-specific models in predicting homology, stability, fluorescence and transmembrane residues.

Table 5.2 Summary of highlighted applications of transformer-based models for protein engineering.

Protein Engineering Task	**Advancements**	**Model Type**	**Training Data Source(s)**	**Year Ref.**
Protein sequence language modeling	Approached protein sequence modeling as a language translation task using one RNN to encode the protein sequence and another RNN to decode the previously encoded sequence.	Transformer	CAFA3	Oct. 2017 [62]
DNA-binding protein prediction	Incorporated sequence context with attention mechanism to improve prediction performance over random-forest, RNN, and support vector machine models.	CNN-Bidirectional RNN	UniProt	Nov. 2019 [63]
Signal peptide sequence generation	A diverse library of 53 artificially generated signal peptides were generated and validated in *Bacillus subtilis*.	Transformer	Swiss-Prot	Aug. 2020 [51]
Novel sequence generation for enhanced GB-1 variants	Used Gibbs Sampling to generate new sequences from BERT language model.	BERT	GB1	Jan. 2021 [64]
Homology, solubility, subcellular localization, stability, fluorescence, secondary structure, and topology prediction	Demonstrated a pre-training strategy that incorporates sequence data with structural information which improved performance on protein fitness prediction tasks compared to similar-size language models.	Bidirectional RNN	Pfam	Sept. 2021 [65]
Variant Effect Prediction	Introduced a new approach to incorporating specialized attention heads and sequence context information.	Transformer	UniRef100	June 2022 [66]
Novel sequence generation	Generated novel protein sequences that mimic properties of natural proteins (e.g., stability, dynamics). Pre-trained model enables rapid, accessible sequence generation on desktop machines.	Transformer	UniRef50, Swiss-Prot	July 2022 [30]
GFP fluorescence intensity, stability	Introduced key design constraints to Transformer model architecture in their regularized latent space optimization (ReLSO) approach to protein sequence modeling.	Transformer	GB1, Gifford, GFP, TAPE	Oct. 2022 [67]

5.3.3 ProtGPT2 is a Deep Unsupervised Language Model for Protein Design

ProtGPT2 is an autoregressive Transformer model capable of generating highly diverse protein sequences [30]. Among the generated sequences, amino acid propensities and fraction of disordered regions are consistent with proteins found in nature, yet the generated sequences are highly distinct from natural proteins. ProtGPT2 provides a useful platform for finetuning based on a particular protein family, function, or fold of interest. Using a Transformer decoder model with byte-pair encoded input sequences enabled self-supervised training on nearly 50 million unlabeled protein sequences from UniRef and Swiss-Prot. With 738 million parameters, ProtGPT2 allows users to generate novel sequences in mere seconds on a desktop computer (https://huggingface.co/nferruz/ProtGPT2).

5.4 Pre-trained language models & Embeddings

Transfer Learning (TL) is a ML technique to transfer useful knowledge learned from a source domain to another related domain (i.e. target domain). This is particularly useful when there is a lack of labeled data in the target domain and obtaining labeled data is time-consuming and costly. Inductive learning, transudative learning and unsupervised learning are specific transfer learning approaches for different applications. Inductive TL improves the target predictive function via the information learned from the source domain prediction task. Note that both the domain and tasks are different but related. Transudative TL aims to improve prediction in target tasks using the learned knowledge from the source domain. However, the learning tasks need to be the same while domains are different. For unsupervised TL, the target domain prediction function still benefits from the source domain when the learning tasks are not the same and there is no labeled data in both the source and the target domain [68].

The use of TL in protein engineering applications can increase the efficiency and generalizability of the downstream tasks via transferring the domain knowledge learned in pretraining to the prediction task. One highly explored and successful application of TL in protein engineering is the use of pretrained language models for predicting protein properties (e.g. thermostability, kinetic activity, binding affinity and disordered regions) from its sequence. These models are trained over

a large number of unlabeled sequences in protein databases such as UniProt [69], UniRef [70] and SRA [71]. With NLP techniques such as masked-token prediction and next-token prediction, these models extract useful information from their training data to be used in downstream tasks. Note that the trained model can either be directly used or its information can be extracted to a fixed-size continuous vector (i.e. an embedding). These embeddings are unique for each input sequence, and they contain structural, evolutionary, statistical and biophysical information about the proteins. This is considered a breakthrough in ML-guided protein engineering tasks where pretrained models alleviate the lack of data and improve performance of ML models. Unified representation (UniRep) is among the early pretrained models which was trained via an mLSTM model over 25 million sequences to distill biophysical and evolutionary information of proteins and represent it in a fixed size representation. The UniRep model has shown generalizations to distant regions of fitness landscape in addition to low number data requirements for viable predictions [72].

5.4.1 Protein Engineering Highlights of Pre-trained Language Models & Embeddings

Table 3 contains a collection of highlighted applications of pre-trained language models and embeddings in protein engineering, along with an added case study for further understanding.

5.4.2 Unsupervised Learning on 250M Protein Sequences Results in Deriving Biological Insights

BERT and GPT are versatile if trained appropriately and have been successfully implemented in the protein sequence domain. Rives et al. [27] trained BERT with 250 million protein sequences to generate representation that contains biological properties. They applied downstream tasks such as remote homology, linear projection, secondary structure prediction, and contact prediction to demonstrate the rich information captured by their deep contextual language model, Evolutionary Scaling Modeling (ESM) (https://github.com/facebookresearch/esm). Additionally, they explored how sequence diversity and model size impact performance. To enable transfer learning of a model to a new task with no additional supervision, extended ESM architectures such as ESM-1v and ESM2 were proposed for variant effect prediction (i.e. mapping sequence changes to functional changes) and capturing high-resolution structural features, respectively [78].

Table 5.3 Summary of highlighted applications of transfer learning & embeddings for protein engineering.

Protein Engineering Task	**Advancements**	**Model Type**	**Training Data Source(s)**	**Year Ref.**
Stability Prediction, fitness (GFP brightness)	Introduced RNN-based unified representation (UniRep) to improve efficiency in protein prediction tasks.	Transformer	UniRef50	Dec. 2019 [26]
Secondary Structure, Subcellular localization, and solubility prediction	Developed the SeqVec model of protein sequences that represents sequences as continuous vectors to predict biophysical and biochemical properties.	ELMo	UniRef50	Dec. 2019 [73]
Secondary Structure, Contact, Homology, Fluorescence, and Stability Prediction	Demonstrated the usefulness of multi-task benchmarks like TAPE to evaluate protein transfer learning models.	Transformer	CBS513, CASP12, TS115, avGFP, ProteinNet, SCOP 1.75	Dec. 2019 [25]
Protein Family Classification, Protein Interaction Prediction	Using the training procedure developed by RoBERTa, PRoBERTa offers a more generalized pre-training framework that outperformed in protein interaction prediction tasks.	Transformer	UniProtKB/Swiss-Prot	Jun. 2020 [74]
Protein folding, binding site, and substitution matrix prediction	Demonstrated how attention can capture protein features from BERT-based models for protein engineering tasks.	Transformer	Pfam, BFD, UniRef100	March 2021 [75]
Secondary Structure, Subcellular Localization, and Solubility Prediction	Demonstrated the usefulness of pre-trained embeddings for various prediction tasks without requiring the use of MSAs.	ARs	UniRef, BDF	July 2021 [28]
Secondary Structure, Contact, Homology, Fluorescence, and Stability Prediction	Improved protein representations by incorporating pairwise masked language model to encode co-evolutionary information.	Transformer	Pfam, UniRef50	Oct. 2021 [76]
Sequence Profile Construction	Performed sequence profile reconstruction with the pre-trained ProtAlbert for sequences with limited homology to database sequences.	Transformer	UniRef, BDF	Aug. 2022 [77]
Structure Prediction	Despite not requiring the use of MSAs, ESM Fold—trained on UniRef sequences—offers rapid structure generation from a single input protein sequence.	Transformer	UniRef50 and UniRef90	March 2023 [78]

There are several attempts to model protein sequences via language model techniques to apply the learned information about protein sequences to protein engineering tasks; some of the most successful ones are listed above. Embedding methods can alleviate the lack of labeled data and improve generalization. In addition, training over self-supervised methods with more parameters leads to capturing more nuanced information about the language of proteins [78]. While transfer learning has shown great promise in protein engineering applications, there is a need for a deeper understanding of what information is learned in pretraining and transferred to the downstream prediction tasks [79]. For example, some studies have observed similar or superior performance for protein fitness prediction without the use of embedding methods [80, 81].

5.5 Probabilistic Modeling of Sequence-VAEs

Unlike transformers that treat protein sequences as a language, variational autoencoders (VAEs) treat sequences as a parameterized multivariate distribution [82]. VAE architecture consists of taking high-dimensional data, reducing it to a low-dimensional representation (encoder) and then reconstructing the representation into the original dimensionality as the input data (decoder) (Figure 5). This encoder-decoder bottleneck structure is also a hallmark of standard autoencoders (AEs). The latent space representation of AEs is a fixed length vector where each value (dimension) is associated with a single learned feature from data. However, the latent representation of VAEs are probability distributions (which are continuous and smooth) for each data attribute. By randomly sampling a vector from latent state distributions, the VAE decoder acts as a generative model that can generate new data instances (e.g. novel protein sequences). The VAE encoder is a recognition model with the ability to recognize statistical distributions that describe variations in data.

The latent representation of VAEs is forced to be continuous and smooth by training the encoder to output pairs of mean and standard deviation (probability distributions) which are subsequently sampled by the decoder. Compared with discrete variable representation of AEs, the continuous distribution representation of VAEs allows the decoder to learn that both a single value and its nearby values refer to the same class. Accordingly, representations of the same class are clustered together as a distribution in latent space, and nearby representations have similar reconstructions. The gap

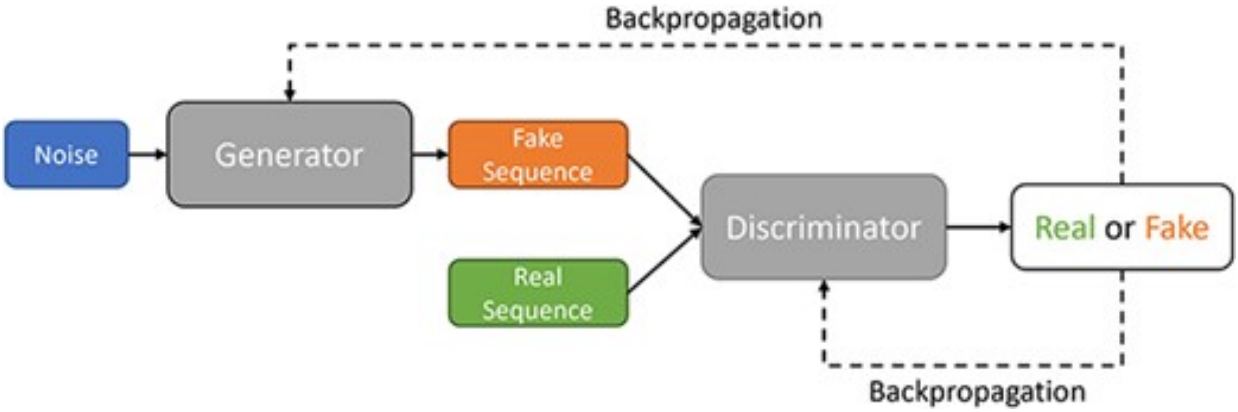


Figure 5.5 GANS architecture for generating sequence data; a model that learns to sample from the given data distribution which contains two separate and opposed networks: generator and discriminator. The generator aims to generate synthetic data from noise which can't be distinguished from the real data by the discriminator. The discriminator on the other hand gets optimized to identify synthetic data from the real data. Evolving together, the model finally will be able to generate samples very similar to the real training data.

between classes in latent space is troublesome, as the decoder has no training data to learn features of those space. Therefore, VAEs incorporate a term in the loss function, the Kullback–Leibler (KL) divergence, which measures how one distribution is different from another. By minimizing the KL-divergence between the learned latent distribution and a prior distribution (e.g. Gaussian), VAEs regularize the latent space. This regularization promotes a continuous latent space which allows VAEs to interpolate values smoothly from one class to another. The reconstruction loss encourages the formation of data points similar to the original input and KL-divergence regularizes the latent space. This incorporation results in a well-structured and information-rich latent space where VAEs can sample from.

Instinctively, for latent variables z that generates observation x, we maximize data likelihood $p(x)$ by maximizing $\int p(z|x)p(x)\,dz$. However, this integral is intractable and cannot be directly optimized. Instead, VAEs use a derivative data likelihood to model $p(x)$ with encoder distribution $p(z|x)$, decoder distribution $p(x|z)$, and latent variables $p(z)$. The posterior distribution, $p(z|x)$, which refers to attributes of latent variables from observation is also intractable, but we can apply variational inference to approximate this density function [83]. By defining a tractable encoder distribution $q(z|x)$ and minimizing KL-divergence between $p(z|x)$ and $q(z|x)$, we obtain the

objective function of VAEs with derivation shown below:

$$D_{KL}(q(z|x)\|p(z|x)) = D_{KL}(q(z)\|p(z)) - E_q\left[\log p(x|z)\right] + \log p(x) \tag{5.2}$$

$$\log p(x) - D_{KL}(q(z|x)\|p(z|x)) = E_q[\log p(x|z)] - D_{KL}(q(z|x)\|p(z)) = \text{ELBO} \tag{5.3}$$

$$\log p(x) \geq probabilitiesextELBO \tag{5.4}$$

Since $D_{KL}[p(z|x)\|q(z|x)]$ is intractable and KL divergence is always positive, we can maximize tractable Evidence Lower-Bound (ELBO) in order to maximize log data likelihood. Within this objective function, $E_q[\log p(x|z)]$ corresponds with the reconstruction loss, and $D_{KL}(q(z|x)\|p(z))$ corresponds KL-divergence loss mentioned in previous paragraph.

Several VAE-derived models have been developed to address common issues like attribute entanglement and posterior collapse. Higgins et al. [84] proposed Beta-VAEs to facilitate learning of the disentanglement of data attributes. By introducing a hyperparameter that penalizes KL divergence loss, the latent representation is forced to adjust the trade-off between reconstruction and regularization. Razavi et al. [85] proposed a method for preventing a common issue in training VAEs, posterior collapse. Posterior collapse happens when the posterior fails to capture the true posterior of the latent variables, and the model gets ineffective in generating diverse and high-quality samples. Their proposed method, delta-VAE, restricts parameters of the posterior to establish minimum KL divergence between prior and posterior.

5.5.1 Protein Engineering Highlights of VAEs

Explore Table 4 for an overview of how VAE models have been implemented for protein engineering applications.

5.5.1.1 Deep Generative Models for T Cell Receptor Protein Sequences

Davidsen et al. [87] demonstrated the capability of VAE models to generate T-cell receptor (TCR) sequences with similar characteristics to real sequences (https://github.com/matsengrp/vampire/). Rather than modeling the probability of a given sequence to undergo VDJ recombination that approaches the properties of the mature TCR repertoire, the architectures of VAEs enables the direct modeling of the distribution of the mature TCR repertoire. In addition to generating novel

Table 5.4 Summary of highlighted applications of VAE models for protein engineering.

Protein Engineering Task	Advancements	Model Type	Training Data Source(s)	Year Ref.
Metallo protein sequence generation and metal binding site prediction	Generated novel metallo proteins that feature copper and calcium binding sites.	VAE	UniRef90	Nov. 2018 [86]
Novel T cell receptor sequence generation	VAE model was able to estimate cohort frequency, learn VDJ recombination regulation, generalize unexplored sequence space, and generate novel T cell receptor sequences.	VAE	Adaptive Biotech immuneAccess	Sept. 2019 [87]
Protein dynamics prediction	Introduced a novel approach to unsupervised learning to train VAEs.	VAE	MoDEL	May 2020 [88]
Novel protein sequence generation	Implemented “Deep Exploration Network" architecture to generate novel sequences with polyadenylation sites, splicing sites, transcription enhancer binding sites, and enhances GFP fluorescence.	VAE	MPRA, APA, avGFP	July 2020 [89]
Novel luciferase sequence generation	Novel luciferase sequences generated from VAE model resulted in increased solubility while maintaining bioluminescence.	VAE	InterPro	Feb. 2021 [90]
Novel protein sequence generation	Proposed a new evaluation metric in their comparison of Potts, VAE and site-independent generative models.	VAE	UniProt/TREMBL, Pfam	Nov. 2021 [91]
Ancestral sequence reconstruction	Demonstrated the ability to capture higher-order epistatic effects through VAE-generated phylogenetic trees.	VAE	CFP, EvolveAGene4 Simulations, PFAM	Feb. 2023 [92]

TCR sequences, the VAE-based models were able to predict the frequency of a TCR in a given cohort and learn the rules of V(D)J recombination. The training data of TCR sequence repertoires were sourced from Adaptive Biotechnologies' ImmunoSEQ assay. Despite some requiring <100 lines of Python code, these simple VAE models were found to outperform previous models that implemented more complicated graphical models that mimic the biological process of V(D)J recombination.

5.6 Sequence Generation through Min-Max Gaming–GANs

Up to this point, we discussed generative models that use explicit probability density functions. RNNs and Ars have a tractable function, and VAEs have an approximate function to maximize likelihood. Here, we turn to generative adversarial networks (GANs), an implicit probabilistic

model that directly generates new data instances by defining a stochastic procedure [33].

GANs employ a two-player game approach to replicate training data distributions without assumptions about their priors. One player is the generator network, and the discriminator network is the other player. The objective of the generator is to generate realistic data instances from random noise to fool the discriminator. On the other hand, the objective of the discriminator is to distinguish real and fake data from the training set and the generator, respectively (Figure 6). By having these two networks competing, the generator learns to generate (fake) data that is close to the (real) training samples, and the discriminator provides feedback to the generator for improvement. This approach allows two networks to evolve with each other so that, ideally, the generator can generate synthetic samples that are indistinguishable from real samples. To train two networks jointly, GANs have a minmax objective function shown below:

$$\min_{\theta_g} \max_{\theta_d} V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}} \left[\log D_\theta(x)\right] + \mathbb{E}_{z \sim p_{\text{noise}}} \left[\log(1 - D_\theta(G_{\theta_g}(z)))\right] \tag{5.5}$$

The minimax function can be interpreted as a function that minimizes the loss that the opponent maximally gives. In GANs, the generator with parameters θ_g wants to minimize the objective value $V(D, G)$ such that the probability of the discriminator output fake data $D_\theta(G_{\theta_g}(\text{noise}))$ is close to 1. This indicates that the generator successfully fooled the discriminator by classifying fake data to real. Conversely, the discriminator with parameters θ_d aims to maximize the objective value such that the probability of the discriminator output real data $D_\theta(x)$ is close to 1, and the probability of the discriminator output fake data $D_\theta(G_{\theta_g}(\text{noise}))$ is close to 0. The training with this minimax function is equivalent to have the generator performing gradient descent on term $\log(1 - D_\theta(G_{\theta_g}(\text{noise})))$ and the discriminator performing gradient ascent on $V(D, G)$. However, the generator of GANs is likely to get stuck in the early stage of training (caused by small gradients) when generated samples are easy to be classified as fake. In practice, the generator performs gradient ascent on term $\log(D_\theta(G_{\theta_g}(\text{noise})))$ instead of gradient descent on term $\log(1 - D_\theta(G_{\theta_g}(\text{noise})))$. In this manner, GANs have steep gradient to drive learning by maximizing the likelihood of the discriminator being wrong instead of minimizing the likelihood of the discriminator being correct.

Developing a loss function for GANs that leads to more stable and better learning is still an active research area. Arjovsky et al.[93] proposed Wasserstein Loss in which the discriminator maximizes $D_\theta(x) - D_\theta(G_{\theta_g}(\text{noise}))$, and the generator maximizes $D_\theta(G_{\theta_g}(\text{noise}))$. This means that the discriminator is not a classifier but a 'critic' that maximizes the difference between proxy number of fake and real data, while the generator maximizes the output of discriminator given generated (fake) data. Usually, the trained discriminator is discarded, and the trained generator is kept for new data generation.

5.6.1 Protein Engineering Highlights of GAN Models

Table 5 illustrates a selection of key GANs models applications in protein engineering, with an added case study for deeper analysis.

Table 5.5 Summary of highlighted applications of GAN models for protein engineering.

Protein Engineering Task	**Advancements**	**Model Type**	**Training Data Source(s)**	**Year**	**Ref.**
Antimicrobial peptide sequence generation	Implemented a GAN model to generate novel antimicrobial peptides	GAN	UniProt	Feb. 2019	[94]
Target-drug binding affinity prediction	Used unlabeled data to train GAN model in a semi-supervised manner	GAN	[97], KIBA	Jan. 2020	[95]
Antibody design	Experimentally validated GAN-generated antibody sequences after initial phage display library screening	GAN	Observed Antibody Space Repository	April 2020	[96]
Ancestral sequence reconstruction	Performed ancestral sequence reconstruction on H3N2 influenza proteins and predicted the evolution of these proteins to improve pathogen forecasting	GAN	NCBI Influenza Virus Resource	Aug. 2020	[97]
Synthetic data generation, gene ontology prediction tasks	Improved gene ontology prediction with GAN model by creating synthetic feature samples	FFPred-GAN	modENCODE	Sept. 2020	[98]
Gene ontology classification	Used WGAN to improve gene ontology term correlation performance	WGAN	SwissProt	Feb. 2021	[99]
Novel malate dehydrogenase sequence generation	Experimentally validated novel malate dehydrogenase sequences with up to 106 mutations	GAN	UniProt	April 2021	[31]

5.6.2 GAN Architecture Enables the Generation of Synthetic Samples to Improve Training

Data augmentation with high-quality synthetic sample data points can help overcome the challenges of developing models that predict protein function. Wan and Jones [98] demonstrate the ability to generate these high-quality synthetic protein feature samples using their GAN-based FFPred-GAN. In addition to using the FFPred model to determine protein biophysical information from protein sequences, FFPred-GAN implemented a WGAN with gradient penalty to learn the distribution of the training protein data set distribution. FFPred-GAN enabled significantly higher accuracy in all the three domains of the gene ontologies domains (i.e. cellular component, molecular function and biological process) without demanding significant computational resources to generate both negative and positive synthetic samples (https://github.com/psipred/FFPredGAN).

5.7 Diffusion Models

Diffusion is a recently developed and rapidly ascending model in the generative AI domain and has shown competitive performance with established benchmarks. This novel method offers better distribution convergence and more diversity in the generated samples. The diffusion model's underlying principle is adopted from non-equilibrium thermodynamics in which the diffusion process increases the system's entropy, driving it towards a state of maximum randomness [34]. Therefore, in the context of generative modeling, diffusion models can gradually transform noisy signals into coherent data structures (i.e. reversing the noise). These models have shown promising results in image synthesis, image inpainting (i.e. filling missing regions in images) and text generation. For example, Dall-E2 [100], a text-to-image framework generated by OpenAI, incorporates a diffusion model during training to generate realistic and high-quality images. Their model resulted in up to four times improvement in resolution compared to the original Dall-E trained with GPT3 architecture [101].

Effective training in generative diffusion models requires a detailed understanding of its main components and foundational concepts. In this section, we describe the core concepts, models, and the main mathematical formulations that have been used for training the diffusion models. Finally, we examine the evolutionary improvements of these models since their introduction in

2015. The forward diffusion process is the transition from data distribution to a prior distribution (e.g. isotropic Gaussian). This is a Markov chain process, and each step only depends on its previous step (i.e. progressively adding noise). For example, we can generate a noisy image at $t = 1$ by adding a small amount of Gaussian noise to the pixel values for the image at $t = 0$, repeating this process for subsequent time steps until the data distribution transforms into a prior distribution. The forward diffusion step parametrization can be shown below:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \tag{6}$$

$$q(x_{1:T}|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1}) = \prod_{t=1}^{T} \mathcal{N}(x_t; \sqrt{1-\beta_t}(x_{t-1}), \beta_t I) \tag{7}$$

where t is the time step and it ranges from 1 to T, x_0 is the instance sampled from the true data distribution, β_t is the variance scheduler, and I is the identity matrix. Given the equations above, the conditional probability distribution of each step given the previous step is assumed to be a conditional Gaussian distribution with mean $\sqrt{1-\beta_t}(x_{t-1})$ and variance $\beta_t I$. Also, the noised image distribution can be directly obtained at each timestep using a reparameterization trick in a closed form [102]. The Backward Diffusion Process represents the challenging task of transforming the noised distribution back to the data distribution. Once accomplished, new data instances can be generated by sampling from the noise distribution. In the backward diffusion process, the model starts with pure Gaussian noise and in each step learns the Gaussian transition parameters with the aid of a parametrized model (e.g., neural networks). Note that this network should have a similar input and output dimension (e.g. U-NET [103] architecture). The backward step parametrization is represented in Equation 8 and Equation 9.

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)\right) \tag{8}$$

$$p_\theta(x_0 : T) = p_\theta(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t) \tag{9}$$

These equations have two main differences with the forward diffusion parametrization: (i) the time trajectory is reversed and (ii) the Gaussian distribution parameters must be learned via a parametrized model.

The diffusion model loss function is a negative log-likelihood (NLL) loss that measures the discrepancy between the true data distribution and the learned distribution. Minimizing the NLL given the model parameter is intractable, but it can become tractable via variational inference techniques. Similar to maximizing the ELBO as discussed in the context of VAEs, the evidence lower bound formulation for training diffusion models after applying Bayes rule and simplifying is a tractable loss function shown in equation 10.

The loss function for denoising diffusion probabilistic models (DDPM) is given by:

$$L = D_{KL}\left(q(x_T|x_0)\|p(x_T)\right) + \sum_{t=2}^{T} D_{KL}\left(q(x_{t-1}|x_t, x_0)\|p(x_{t-1}|x_t)\right) - \log p_\theta(x_0|x_1) \tag{10}$$

In denoising diffusion probabilistic models (DDPM), the authors explored and reformulated the loss function above where the variance was held constant, and the neural network was designed to predict the noise only at each time step. This results in a simple and easily implementable loss function represented in equation 11: the mean squared error between the actually added noise in the forward process and predicted noise by the model.

$$L = E_{t,x_0}\left[\|\epsilon - \hat{\epsilon}_\theta(x_t, t)\|^2\right] \tag{11}$$

Diffusion has a more intricate path in model development and refinement compared to the other mentioned generative models. The idea of using a diffusion process in deep unsupervised learning was proposed in 2015 by Sohl-Dickstein et al. [34] in which the data distribution was destroyed gradually via an 'iterative' forward process and Markov chain method. The authors argued that the reverse diffusion process (restoring data distribution from known distribution (e.g. normal distribution)) yields a tractable generative model when applied with a sufficient number of steps. The reasoning behind this was that small perturbations in data are more tractable for prediction than one-time distribution prediction. In 2020, Ho et al. [102] introduced a series of novel enhancements

to this technique, leading to high-quality image synthesis through denoizing diffusion probabilistic models (DDPMs). In their research, the authors employed a linear noise scheduler and innovatively chose to predict the image noise during each iteration in the backward process. Building upon these advancements, the model's performance was further elevated by incorporating

as a learned parameter in the normal distribution instead of a fixed number. Also, introducing non-linear noise-schedulers (i.e. cosine scheduler) resulted in effective preservation of the data distribution in early nosing steps throughout the forward diffusion process [104, 105].

One main breakthrough in diffusion generative model development was made by Song et al. [106] by incorporating a stochastic differential equation (SDE) framework. The 'score function' in their methodology refers to the gradient of the log probability density. In the forward process, the data distribution gets perturbed in continuous space (in contrast to earlier diffusion models with finite noising steps) via the suggested SDE formulation which does not have trainable parameters. Reverse SDE can be solved analytically with methods like Euler-Maruyama after handling the score function term [107]. The authors addressed this by modeling the score function using a neural network, which then can be plugged into the reverse SDE formula. Equations 12 and 13 show the main forward and reverse formulation used in an SDE process.

The stochastic differential equations (SDEs) used in the models are given by:

$$dx = f(x,t)\,dt + g(t)\,dw \tag{12}$$

$$dx = f(x,t) - \left[g^2(t)\nabla \log p_t(x)\right]\,dt + g(t)\,d\tilde{w} \tag{13}$$

The inclusion of SDEs in score-based generative models led to enhanced flexibility, particularly by eliminating the constant prior in favor of utilizing the density gradient. This method provided a controlled generation process and exact likelihood calculations. Although this proposed method enabled efficient and high-quality sampling, the authors noted a slower sampling compared to GANs over their tested dataset.

Diffusion models have been adopted into protein engineering applications recently, and they have shown incredible performance in generating novel protein structures and sequences. In this

complex landscape, diffusion models offer distinct advantages among generative models: diversity, fine-grained control in generation, stability in training, a more favorable platform for conditioning, and high compatibility for sequence and structure co-design [108, 109, 110, 111]. While they are generally more computationally intensive than other generative models, the probabilistic nature of diffusion models allows for the generation of diverse protein conformations from initial noise distribution. This inherent uncertainty is particularly beneficial and offers a more realistic modeling approach since proteins are dynamic and adopt multiple conformations. Given these unique features in their architecture and training procedure, diffusion models are potentially an invaluable tool for navigating the intricate energy landscape that proteins operate within.

5.7.1 Protein Engineering Highlights of Diffusion Models

Explore Table 6 for an array of recently developed diffusion model applications in protein engineering, extended with two analytical case studies.

5.7.1.1 ProteinGenerator Enables the Joint Generation of Protein Sequence and Structure

The authors implemented DDPM with coordinated guidance on sequence and structure resulting in improved generation (github.com/RosettaCommons/protein_generator) [110]. They leveraged RoseTTAFold's [116] capability to simultaneously generate protein sequences and structures. Drawing inspiration from RoseTTAFold Joint Inpainting, they adopted this ability for the diffusive creation of consistent sequence-structure pairs. Fine-tuning to retrieve noised native protein sequences and simultaneously ensuring the accuracy of structure prediction enabled guidance from both sequential and structural domains. In the unconditional generation, ProteinGenerator was able to generate pairs of sequence structures close to the native proteins. Note that various structural properties and amino acid frequencies were obtained by sampling from different noise distributions. The model architecture also enabled high versatility and as a result compatibility with different conditioning and classifier-guidance methods. In conditioning, additional constraints were added to the generation process. For instance, the model was conditioned for generating high amino acid frequencies (e.g. cystine for forming disulfide bonds to increase stability, histidine for pH sensitivity) while satisfying the corresponding structure folding. In another example, DeepGOPlus

Table 5.6 Summary of highlighted applications of diffusion models for protein engineering.

Protein Engineering Task	**Advancements**	**Model Type**	**Training Data Source(s)**	**Year**	**Ref.**
Protein sequence inpainting and generation	Demonstrated the use of equivariant denoising using the invariant point attention technique to jointly model protein sequence and structures	Transformer-Based Diffusion Model	PDB	May 2022	[112]
Joint Sequence-Structure Generation	Modeled structure and sequence of full protein complexes in a computationally efficient manner	Graph Neural Network-Based Diffusion Model	PDB, UniProt	Dec. 2022	[109]
Protein Backbone Joint Sequence-Structure model development	Demonstrated the feasibility of developing generative diffusion models through a comparison of sequence-only, structure-only, and joint sequence-structure models	CARP	PDB	May 2023	[113]
Joint Sequence-Structure Generation	Developed a model capable of generating both novel sequences and novel protein backbones via RoseTTAFold	DDPM	INDI, SCOP	May 2023	[110]
De Novo Protein Sequence Generation with desired structural features	Generated novel protein sequences with desired secondary-structure features using an attention-based diffusion model	Attention-Based Diffusion Model	–	July 2023	[114]
Antibody joint Sequence-structure Modeling	Improved joint protein sequence and structure generation using both domain knowledge and physics-based constraints	SE(3)-based Diffusion Model	pOAS Database, HER2 Binder data set	July 2023	[115]

Gene Ontology (GO) classifier was used to guide the generation process. The classifier provides scores or gradients that can be used to influence the outputs of the main model and as a result, generate functionally rich sequences.

5.7.1.2 Chroma Enables the Generation of Novel Protein Complexes via its Joint Sequence and Structure Model

Another highly successful implementation of the diffusion-based framework was shown in Chroma which enabled jointly modeling the sequence and structure of full protein complexes (https://github.com/lucidrains/chroma-pytorch) [109]. The authors introduced sophisticated computational techniques and conditional sampling to adeptly manage computational challenges while crafting proteins with specific attributes. Rooted in diffusion modeling and graph neural networks, this versatile generative model excels in refining noisy structures while preserving the intricate 3D details inherent in protein configurations. This model facilitates programmable protein design as it can condition proteins on different shapes, symmetry, textual prompts, and various properties. Remarkably, Chroma's capability to generate protein complexes holds significant value as most of the protein functions such as binding occur through protein interactions. Furthermore, the authors indicated that a large protein (e.g. with > 3000 residues) can be generated within minutes via an appropriate GPU (e.g. NVIDIA V100).

5.8 Discussion

Generative models—such as VAEs, autoregressive, GANs, and diffusion models—have shown significant promise in the protein engineering domain to generate novel and functional sequences. This ongoing research has mitigated long-standing challenges in designing proteins with improved properties, generating interfaces for protein–protein interactions, establishing rules for high-fitness protein variants and capturing phylogenetic relationships between proteins. These models aim to learn the underlying data distribution and generate novel instances via sampling from the learned distribution. Distinct model structures are employed to learn the given data distribution by directly modeling or approximating the probability density function. VAEs are probabilistic generative models that approximate the explicit density function via variational inference. Upon learning the

underlying distribution of the given dataset, the VAE can generate novel samples similar to input data. VAEs have been used in various protein engineering tasks including improving fitness (e.g. thermal stability, solubility, bioluminescence and binding) and capturing phylogenetic relationships via learned latent space relationships. Autoregressive models calculate the explicit density function where each token is conditioned on the previous tokens. Autoregressive models have also led to successful outcomes in generating sequences with improved fitness, paratope prediction, and protein localization. Unlike VAEs and autoregressive models that use restricted neural networks in approaching the intractable normalizing constant, GANs model the generation process only. As a result, they are not used for likelihood estimation, yet they have superior potential in generating high-quality instances. Two networks (generator and discriminator) are used in GANs that sample from the density function without calculating or estimating the function itself (i.e. implicit density estimation). GANs provide promising results in diverse tasks such as gene ontology correlation, binding affinity, phylogeny prediction, antimicrobial peptide generation and developing rules for antibody solubility and thermal stability. Diffusions are a more recent class of generative models adopted from thermodynamics equilibrium. The idea is if the noise in the data happens gradually, it can be reversed. Therefore, data distribution can be approximated from pure noise in the reverse diffusion process.

While each of these generative models has obtained promising outcomes in terms of protein design applications, they differ in their training process, output quality and generated output diversity. In general, given their efficient architecture, VAEs are potentially easier to train, yet they might lead to lower quality outputs (e.g. blurry images for image generation) compared to other generative models [117]. Note that recent architecture developments have tried to overcome common issues in VAEs (e.g. posterior collapse and reconstruction-regularization trade off), yet these solutions may require more computational resources. For instance, beta-VAE [84], hierarchical VAE [118, 119] and VQ-VAE [120] are distinct types of VAE models to address common issues in traditional VAEs. Beta-VAE adds a hyperparameter in the loss function to obtain more disentangled representations. Hierarchical VAE aims to preclude posterior over-regularization

by incorporating hierarchical priors in the model. Finally, VQ-VAE has been shown to generate high quality data and prevent posterior collapse by learning discrete representations and autoregressive prior (versus continuous learned representations and static prior in original traditional VAE). Similarly, there are improved variants for GANs and autoregressive models to boost generated data attributes and resolve model restrictions. Examples of autoregressive developments include GPT-3 [58], Reformer [121] and Big Bird [122] which use more parameters in training, reversible sequence-to-sequence architecture, and sparse attention mechanism, respectively. For GANS, improved variants include CycleGAN [123], LsGAN [124] and VEEGAN [125] for training without paired data, resolving vanishing gradient issues in training and reducing mode collapse to increase generated data diversity, respectively. Although diffusion models have been developed recently, their architecture is rapidly evolving. For example, subspace diffusion has shown improved sampling quality and reduced computational cost via restricting diffusion by its projection to subspaces [126]. Denoising diffusion policy optimization (DDPO) is another architecture development in diffusions which solved the denoising process as a multi-step decision-making problem [127]. The mentioned architectures are a few variants among a pool of architectures and their performance depends on the specific application and data attributes (e.g. number of samples in training, data complexity and input data length).

Despite the newfound opportunities provided by generative models in this realm, the remaining challenges in generative sequence modeling include validating the generated sequences, navigating the rugged landscape in pursuit of sequences with desired features, de-novo binder design application, effectively infusing biological priors into models and strategically combining distinct generative models to enhance sampling quality and diversity. In many cases, wet-lab experiments are required to assess the quality of the generated sequence in terms of basic required properties (e.g. stability and expression) to more design-based properties (e.g. affinity and specificity). This by itself has hindered model optimization as there is no immediate and definitive feedback for the quality of generated sequences (versus rapidly assessing the visual quality of a general image-based data generated from these models). With that being said, there are computational tools to aid in filtering

the generated sequences and increasing the success rate in experimental characterization. For example, Alphafold2 for structural prediction [128], discriminative models to assign probabilities to sequences based on their fitness [129], and self-supervised models for few and zero-shot predictions [134] are among the extremely beneficial tools for analyzing the generated sequences in silico.

In this paper, we provided an overview of the architecture and underlying assumptions of four commonly used generative models (VAEs, Autoregressive models, GANs and diffusion models). By analyzing the strengths and limitations of each model, we hope that researchers are better equipped to make informed decisions when selecting the appropriate model for specific data and objectives. We also elaborated on specific protein engineering applications for each of these models, highlighting their potential to generate novel protein sequences with improved properties. With the exponential growth of biological and protein sequence datasets, increasing efficiency of generative models, and improved methods for generating and validating de novo sequences, we envision a promising future for the development of effective protein design and engineering applications.

5.8.1 Key Points

- To address the gap between the growing number of machine learning (ML) models and their application to protein engineering tasks, we have reviewed recent protein engineering applications of generative ML models.
- The architecture and mathematical background of three generative models (diffusion models, generative adversarial neural networks and variational autoencoders) are described in depth with a focus on applications towards protein design (e.g. to predict protein properties and to generate protein design rules and sequences).
- The architecture and application of language machine learning models (namely, recurrent neural networks, autoregressive and transformers) are also described, particularly in the context of treating protein design tasks on amino acid sequences similarly to human language tasks on strings of text.
- Incorporating transfer learning and embeddings can improve the efficiency and generalizability of ML modeling tasks.

BIBLIOGRAPHY

[1] JM Webster, R Zhang, SS Gambhir, et al. Engineered two-helix small proteins for molecular recognition. *Chem Bio Chem*, 10:1293–1296, 2009.

[2] CS Eke, E Jammeh, X Li, et al. Early detection of alzheimer's disease with blood plasma proteins using support vector machines. *IEEE J Biomed Health Inform*, 25:218–226, 2021.

[3] Y Luan and Y Yao. The clinical significance and potential role of c-reactive protein in chronic inflammatory and neurodegenerative diseases. *Front Immunol*, 9:1302, 2018.

[4] R Bam, PS Lown, LA Stern, et al. Efficacy of affibody-based ultrasound molecular imaging of vascular b7-h3 for breast cancer detection. *Clin Cancer Res*, 26:2140–2150, 2020.

[5] J Małecki, S Muszyński, and BG Sołowiej. Proteins in food systems—bionanomaterials, conventional and unconventional sources, functional properties, and development opportunities. *Polymers*, 13:2506, 2021.

[6] DB Janssen and JP Schanstra. Engineering proteins for environmental applications. *Curr Opin Biotechnol*, 5:253–259, 1994.

[7] K Kuroda and M Ueda. Molecular design of the microbial cell surface toward the recovery of metal ions. *Curr Opin Biotechnol*, 22:427–433, 2011.

[8] D Prakash, P Gabani, AK Chandel, et al. Bioremediation: a genuine technology to remediate radionuclides from the environment. *J Microbial Biotechnol*, 6:349–360, 2013.

[9] JM Jez. Toward protein engineering for phytoremediation: possibilities and challenges. *Int J Phytoremediation*, 13:77–89, 2011.

[10] X Jia, Y Li, T Xu, and K Wu. Display of lead-binding proteins on escherichia coli surface for lead bioremediation. *Biotechnol Bioeng*, 117:3820–3834, 2020.

[11] MD Diem, L Hyun, F Yi, et al. Selection of high-affinity centyrin fn3 domains from a simple library diversified at a combination of strand and loop positions. *Protein Eng Des Sel*, 27:419–429, 2014.

[12] AW Golinski, KM Mischler, S Laxminarayan, et al. High-throughput developability assays enable library-scale identification of producible protein scaffold variants. *Proc Natl Acad Sci*, 118:e2026658118, 2021.

[13] M Zacharias. Protein–protein docking with a reduced protein model accounting for side-chain flexibility. *Protein Sci*, 12:1271–1282, 2003.

[14] R Merkl and R Sterner. Reconstruction of ancestral enzymes. *Perspect Sci*, 9:17–23, 2016.

[15] A Vaswani, N Shazeer, N Parmar, et al. Attention is all you need. *arXiv preprint*, page arXiv:1706.03762, 2017.

[16] B Ghojogh and A Ghodsi. Attention mechanism, transformers, bert, and gpt: Tutorial and survey. *Open Science Framework*, 2020.

[17] Y LeCun, Y Bengio, and G Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[18] D Ofer, N Brandes, and M Linial. The language of proteins: Nlp, machine learning protein sequences. *Comput Struct Biotechnol J*, 19:1750–1758, 2021.

[19] K Wang, R Zhou, Y Li, and M Li. Deepdtaf: a deep learning method to predict protein–ligand binding affinity. *Brief Bioinform*, 22:bbab072, 2021.

[20] G Li, KS Rabe, J Nielsen, and MKM Engqvist. Machine learning applied to predicting microorganism growth temperatures and enzyme catalytic optima. *ACS Synth Biol*, 8:1411–1420, 2019.

[21] S Khurana, R Rawi, K Kunji, et al. Deepsol: a deep learning framework for sequence-based protein solubility prediction. *Bioinformatics*, 34:2605–2613, 2018.

[22] S Hashemifar, B Neyshabur, AA Khan, and J Xu. Predicting protein–protein interactions through sequence-based deep learning. *Bioinformatics*, 34:i802–i810, 2018.

[23] L Wang, H-F Wang, S-R Liu, et al. Predicting protein-protein interactions from matrix-based protein sequence using convolution neural network and feature-selective rotation forest. *Sci Rep*, 9:9848, 2019.

[24] N Ferruz, S Schmidt, and B Höcker. A deep unsupervised language model for protein design. *2022*, page 2022.03.09.483666, 2022.

[25] R Rao, N Bhattacharya, N Thomas, et al. Evaluating protein transfer learning with tape. *Adv Neural Inf Process Syst*, 32:9689–9701, 2019.

[26] EC Alley, G Khimulya, S Biswas, et al. Unified rational protein engineering with sequence-based deep representation learning. *Nat Methods*, 16:1315–1322, 2019.

[27] A Rives, J Meier, T Sercu, et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proc Natl Acad Sci*, 118:e2016239118, 2021.

[28] A Elnaggar, M Heinzinger, C Dallago, et al. Prottrans: toward understanding the language of life through self-supervised learning. *IEEE Trans Pattern Anal Mach Intell*, 44:7112–7127, 2022.

[29] Z Costello and HG Martin. How to hallucinate functional proteins. *arXiv*, page arXiv:1903.00458v1, 2019.

[30] N Ferruz, S Schmidt, and B Höcker. Protgpt2 is a deep unsupervised language model for protein design. *Nat Commun*, 13:4348, 2022.

[31] D Repecka, V Jauniskis, L Karpus, et al. Expanding functional protein sequence spaces using generative adversarial networks. *Nat Mach Intell*, 3:324–333, 2021.

[32] DP Kingma and M Welling. Auto-encoding variational bayes. *arXiv*, page arXiv:1312.6114v11, 2022.

[33] IJ Goodfellow, J Pouget-Abadie, M Mirza, et al. Generative adversarial networks. *arXiv*, page arXiv:1406.2661, 2014.

[34] J Sohl-Dickstein, E Weiss, N Maheswaranathan, and S Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *Proceedings of the 32nd International Conference on Machine Learning*, pages 2256–2265, 2015.

[35] A Sherstinsky. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Phys Nonlinear Phenom*, 404:132306, 2020.

[36] M Schuster and KK Paliwal. Bidirectional recurrent neural networks. *IEEE Trans Signal Process*, 45:2673–2681, 1997.

[37] S Hochreiter and J Schmidhuber. Long short-term memory. *Neural Comput*, 9:1735–1780, 1997.

[38] J Chung, C Gulcehre, K Cho, and Y Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv*, page arXiv:1412.3555, 2014.

[39] AT Müller, JA Hiss, and G Schneider. Recurrent neural network model for constructive peptide design. *J Chem Inf Model*, 58:472–479, 2018.

[40] K Saka, T Kakuzaki, S Metsugi, et al. Antibody design using lstm based deep generative model from phage display library for affinity maturation. *Sci Rep*, 11:5852, 2021.

[41] S Sabban and M Markovsky. Ramanet: computational de novo helical protein backbone design using a long short-term memory generative neural network. *F1000 Research Full*, 9:671552, 2020.

[42] B Zhang, J Li, and Q Lü. Prediction of 8-state protein secondary structures by a novel deep learning architecture. *BMC Bioinformatics*, 19:293, 2018.

[43] C-C Lin, A Jaech, X Li, et al. Limitations of autoregressive models and their alternatives.

Association for Computational Linguistics, 2021.

[44] J Trinquier, G Uguzzoni, A Pagnani, et al. Efficient generative modeling of protein sequences using simple autoregressive models. *Nat Commun*, 12:5800, 2021.

[45] J-E Shin, AJ Riesselman, AW Kollasch, et al. Protein design and variant prediction using autoregressive generative models. *Nat Commun*, 12:2403, 2021.

[46] X Liu. Deep recurrent neural network for protein function prediction from sequence. *arXiv*, page arXiv:1701.08318, 2017.

[47] B Panda and B Majhi. A novel improved prediction of protein structural class using deep recurrent neural network. *Evol Intell*, 14:253–260, 2021.

[48] WP Russ, M Figliuzzi, C Stocker, et al. An evolution-based model for designing chorismate mutase enzymes. *Science*, 369:440–445, 2020.

[49] P Zhang, S Zheng, J Chen, et al. Deepanis: Predicting antibody paratope from concatenated cdr sequences by integrating bidirectional long-short-term memory and transformer neural networks. *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 118–124, 2021.

[50] D. Grechishnikova. Transformer neural network for protein-specific de novo drug generation as a machine translation problem. *Scientific Reports*, 11:321, 2021.

[51] Z. Wu, K. K. Yang, M. J. Liszka, et al. Signal peptides generated by attention-based neural networks. *ACS Synthetic Biology*, 9:2154–2161, 2020.

[52] I. Ieremie, R. M. Ewing, and M. Niranjan. Transformergo: predicting protein–protein interactions by modeling the attention between sets of gene ontology terms. *Bioinformatics*, 38:2269–2277, 2022.

[53] C. Chen, T. Wu, Z. Guo, and J. Cheng. Combination of deep neural network with attention mechanism enhances the explainability of protein contact prediction. *Proteins: Structure, Function, and Bioinformatics*, 89:697–707, 2021.

[54] K. O'Shea and R. Nash. An introduction to convolutional neural networks. arXiv:1511.08458, 2015.

[55] Q. Zhao, H. Zhao, K. Zheng, and J. Wang. Hyperattentiondti: improving drug–protein interaction prediction by sequence-based deep learning with attention mechanism. *Bioinformatics*, 38:655–662, 2022.

[56] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Association for*

Computational Linguistics (ACL), 2019.

[57] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training, 2018.

[58] T. B. Brown, B. Mann, N. Ryder, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 1877–1901, 2020.

[59] M. Tsimpoukelli, J. L. Menick, S. Cabi, et al. Multimodal few-shot learning with frozen language models. In *Proceedings of Neural Information Processing Systems (NeurIPS)*, volume 34, pages 200–212, 2021.

[60] M. Lewis, Y. Liu, N. Goyal, et al. Bart: Denoising sequence-to-sequence pre-training for natural language generation. *Transactions on Comprehension*, 58:7871–7880, 2019.

[61] K. Choromanski, V. Likhosherstov, D. Dohan, et al. Masked language modeling for proteins via linearly scalable long-context transformers. arXiv:2006.03555, 2020.

[62] R. Cao, C. Freitas, L. Chan, et al. Prolango: protein function prediction using neural machine translation based on a recurrent neural network. *Molecules*, 22:1732, 2017.

[63] S. Hu, R. Ma, and H. Wang. An improved deep learning method for predicting dna-binding proteins based on contextual features in amino acid sequences. *PLoS One*, 14:e0225317, 2019.

[64] S. R. Johnson, S. Monaco, K. Massie, and Z. Syed. Generating novel protein sequences using gibbs sampling of masked language models. arXiv:2021.01.26.428322, 2021.

[65] S. Min, S. Park, S. Kim, et al. Pre-training of deep bidirectional protein sequence representations with structural information. *IEEE Access*, 9:123912–123926, 2021.

[66] P. Notin, M. Dias, J. Frazer, et al. Tranception: Protein fitness prediction with autoregressive transformers and inference-time retrieval. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, pages 16990–17017, 2022.

[67] E. Castro, A. Godavarthi, J. Rubinfien, et al. Transformer-based protein generation with regularized latent space optimization. *Nature Machine Intelligence*, 4:840–851, 2022.

[68] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22:1345–1359, 2010.

[69] UniProt. A hub for protein information. *Nucleic Acids Research*, 43:D204–D213, 2023.

[70] B. E. Suzek, H. Huang, P. McGarvey, et al. Uniref: comprehensive and non-redundant uniprot reference clusters. *Bioinformatics*, 23:1282–1288, 2007.

[71] K. Katz, O. Shutov, R. Lapoint, et al. The sequence read archive: a decade more of explosive growth. *Nucleic Acids Research*, 50:D387–D390, 2022.

[72] S. Biswas, G. Khimulya, E. C. Alley, et al. Low-n protein engineering with data-efficient deep learning. *Nature Methods*, 18:389–396, 2021.

[73] M. Heinzinger, A. Elnaggar, Y. Wang, et al. Modeling aspects of the language of life through transfer-learning protein sequences. *BMC Bioinformatics*, 20:723, 2019.

[74] A. Nambiar, S. Liu, M. Hopkins, et al. Transforming the language of life: transformer neural networks for protein prediction tasks. *Journal of Computational Biology*, 30:95–111, 2020.

[75] J. Vig, A. Madani, L. R. Varshney, et al. Bertology meets biology: interpreting attention in protein language models. arXiv:2006.15222, 2021.

[76] L. He, S. Zhang, L. Wu, et al. Pre-training co-evolutionary protein representation via a pairwise masked language model. arXiv:2110.15527, 2021.

[77] A. Behjati, F. Zare-Mirakabad, S. S. Arab, and A. Nowzari-Dalini. Protein sequence profile prediction using protalbert transformer. *Computational Biology and Chemistry*, 99, 2021.

[78] Z. Lin, H. Akin, R. Rao, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379:1123–1130, 2023.

[79] M. Mardikoraem and D. Woldring. Protein fitness prediction is impacted by the interplay of language models, ensemble learning, and sampling methods. *Pharmaceutics*, 15(5), 2023.

[80] A. Shanehsazzadeh, D. Belanger, and D. Dohan. Is transfer learning necessary for protein landscape prediction? arXiv:2011.03443, 2020.

[81] B. J. Wittmann, Y. Yue, and F. H. Arnold. Informed training set design enables efficient machine learning-assisted directed protein evolution. *Cell Systems*, 12:1026–1045.e7, 2021.

[82] S. Sinai, E. Kelsic, G. M. Church, and M. A. Nowak. Variational auto-encoding of protein sequences. arXiv:1712.03346, 2018.

[83] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: a review for statisticians. *Journal of the American Statistical Association*, 112:859–877, 2017.

[84] I. Higgins, L. Matthey, A. Pal, et al. Beta-vae: learning basic visual concepts with a constrained variational framework. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

[85] A. Razavi and O. Vinyals. Preventing posterior collapse with δ-vaes. arXiv:1901.03416, 2019.

[86] J. G. Greener, L. Moffat, and D. T. Jones. Design of metalloproteins and novel protein folds using variational autoencoders. *Scientific Reports*, 8:16189, 2018.

[87] K. Davidsen, B. J. Olson, III DeWitt, W. S., et al. Iv deep generative models for t cell receptor protein sequences. *Elife*, 8:e46935, 2019.

[88] A.-I. Albu and G. Czibula. Analysing protein dynamics using machine learning based generative models. In *Proceedings of the 2020 IEEE 14th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, pages 135–140, 2020.

[89] J. Linder, N. Bogard, A. B. Rosenberg, and G. Seelig. A generative neural network for maximizing fitness and diversity of synthetic dna and protein sequences. *Cell Systems*, 11:49–62.e16, 2020.

[90] A. Hawkins-Hooker, F. Depardieu, S. Baur, et al. Generating functional protein variants with variational autoencoders. *PLoS Computational Biology*, 17:e1008736, 2021.

[91] F. McGee, S. Hauri, Q. Novinger, et al. The generative capacity of probabilistic protein sequence models. *Nature Communications*, 12:6302, 2021.

[92] L. S. Moreta, O. Rønning, and A. S. Al-Sibahi. Ancestral protein sequence reconstruction using a tree-structured ornstein-uhlenbeck variational autoencoder. In *International Conference on Learning Representations (ICLR)*, 2022.

[93] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. arXiv:1701.07875, 2017.

[94] A. Gupta and J. Zou. Feedback gan for dna optimizes protein functions. *Nature Machine Intelligence*, 1:105–111, 2019.

[95] L. Zhao, J. Wang, L. Pang, et al. Gansdta: predicting drug-target binding affinity using gans. *Frontiers in Genetics*, 10:1243, 2020.

[96] T. Amimeur, J. M. Shaver, R. R. Ketchem, et al. Designing feature-controlled humanoid antibody discovery libraries using generative adversarial networks. arXiv:2020.04.12.024844, 2020.

[97] D. S. Berman, C. Howser, T. Mehoke, and J. D. Evans. Mutagan: A seq2seq gan framework to predict mutations of evolving protein populations. *Virus Evolution*, 2020.

[98] C. Wan and D. T. Jones. Protein function prediction is improved by creating synthetic feature samples with generative adversarial networks. *Nature Machine Intelligence*, 2:540–550, 2020.

[99] S. F. Seyyedsalehi, M. Soleymani, H. R. Rabiee, and M. R. K. Mofrad. Pfp-wgan: protein function prediction by discovering gene ontology term correlations with generative

adversarial networks. *PLoS One*, 16:e0244430, 2021.

[100] A. Ramesh, P. Dhariwal, A. Nichol, et al. Hierarchical text-conditional image generation with clip latents. arXiv:2204.06125, 2022.

[101] A. Ramesh, M. Pavlov, G. Goh, et al. Zero-shot text-to-image generation. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pages 8821–8831, 2021.

[102] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. arXiv:2006.11239, 2020.

[103] W. Weng and X. Zhu. Inet: convolutional networks for biomedical image segmentation. *IEEE Access*, 9:16591–16603, 2021.

[104] A. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. arXiv:2102.09672, 2021.

[105] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 8780–8794, 2021.

[106] H. Song, B. J. Bremer, E. C. Hinds, et al. Inferring protein sequence-function relationships with large-scale positive-unlabeled learning. *Cell Systems*, 12:92–101.e8, 2021.

[107] M. Bayram, T. Partal, and G. Orucova Buyukoz. Numerical methods for simulation of stochastic differential equations. *Advances in Differential Equations*, 2018.

[108] L. Yang, Z. Zhang, Y. Song, et al. Diffusion models: a comprehensive survey of methods and applications. arXiv:2209.00796, 2023.

[109] J. Ingraham, M. Baranov, Z. Costello, et al. Illuminating protein space with a programmable generative model. bioRxiv:2022.12.01.518682, 2022.

[110] S. L. Lisanza, J. M. Gershon, S. Tipps, et al. Joint generation of protein sequence and structure with rosettafold sequence space diffusion. bioRxiv:2023.05.08.539766, 2023.

[111] B. Ni, D. L. Kaplan, and M. J. Buehler. Generative design of de novo proteins based on secondary-structure constraints using an attention-based diffusion model. *Chem*, 9:1828–1849, 2023.

[112] N. Anand and T. Achim. Protein structure and sequence generation with equivariant denoising diffusion probabilistic models. arXiv:2205.15019, 2022.

[113] R. Vinod, K. K. Yang, and L. Crawford. Joint protein sequence-structure co-design via equivariant diffusion, 2022.

[114] C.-H. Yu, W. Chen, Y.-H. Chiang, et al. End-to-end deep learning model to predict and design secondary structure content of structural proteins. *ACS Biomaterials Science and Engineering*, 8:1156–1165, 2022.

[115] K. Martinkus, J. Ludwiczak, K. Cho, et al. Abdiffuser: full-atom generation of in-vitro functioning antibodies. arXiv:2308.05027, 2023.

[116] M. Baek, F. DiMaio, I. Anishchenko, et al. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373:871–876, 2021.

[117] A. Dosovitskiy and T. Brox. Generating images with perceptual similarity metrics based on deep networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 29, 2016.

[118] A. Klushyn, N. Chen, R. Kurle, et al. Learning hierarchical priors in vaes. arXiv:1905.04982, 2019.

[119] C. K. Sønderby, T. Raiko, L. Maaløe, et al. Ladder variational autoencoders. arXiv:1602.02282, 2016.

[120] A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. arXiv:1711.00937, 2018.

[121] N. Kitaev, Ł. Kaiser, and A. Levskaya. Reformer: the efficient transformer. arXiv:2001.04451, 2020.

[122] M. Zaheer, G. Guruganesh, K. A. Dubey, et al. Big bird: transformers for longer sequences. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 17283–17297, 2020.

[123] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv:1703.10593, 2020.

[124] X. Mao, Q. Li, H. Xie, et al. Least squares generative adversarial networks. arXiv:1611.04076, 2017.

[125] A. Srivastava, L. Valkov, C. Russell, et al. Veegan: Reducing mode collapse in gans using implicit variational learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

[126] B. Jing, G. Corso, R. Berlinghieri, and T. Jaakkola. Subspace diffusion generative models. In *Computer Vision – ECCV 2022*, volume 13683, pages 274–289. Springer Nature Switzerland, 2022.

[127] K. Black, M. Janner, Y. Du, et al. Training diffusion models with reinforcement learning.

arXiv:2305.13301, 2023.

[128] J. Jumper, R. Evans, A. Pritzel, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596:583–589, 2021.

[129] A. Strokach and P. M. Kim. Deep generative modeling for protein design. *Current Opinion in Structural Biology*, 72:226–236, 2022.

CHAPTER 6

EVOSEQ-ML: ADVANCING DATA-CENTRIC MACHINE LEARNING WITH EVOLUTIONARY-INFORMED PROTEIN SEQUENCE REPRESENTATION AND GENERATION

[1]

Abstract

In protein engineering, machine learning (ML) advancements have led to significant progress, including protein structure prediction (e.g., AlphaFold), sequence representation through language models, and novel protein generation. However, the impact of data curation on ML model performance is underexplored. As more sequence and structural data become available, a data-centric approach is increasingly favored over a model-centric method. A data-centric approach prioritizes high-quality, domain-specific data, ensuring ML tools are trained on datasets that accurately reflect biological complexity and diversity. This paper introduces a novel methodology that integrates ancestral sequence reconstruction (ASR) into ML models, enhancing data-centric strategies in the field. ASR uses computational techniques to infer ancient protein sequences from modern descendants, providing diverse, stable sequences with rich evolutionary information. While multiple sequence alignments (MSAs) are commonly used in protein engineering frameworks to incorporate evolutionary information, ASR offers deeper insights into protein evolution. Unlike MSAs, ASR captures mutation rates, phylogenic relationships, evolutionary trajectories, and specific ancestral sequences, giving access to novel protein sequences beyond what is available in public databases by natural selection. We employed two statistical methods for ASR: joint Bayesian inference and maximum likelihood. Bayesian approaches infer ancestral sequences by sampling from the entire posterior distribution, accounting for epistatic interactions between multiple amino acid positions to capture the nuances and uncertainties of ancestral sequences. In contrast, maximum likelihood methods estimate the most probable amino acids at individual positions in isolation. Both methods provide extensive ancestral data, enhancing ML model performance in protein sequence

[1]This chapter is adapted from content published in "ICLR 2024 Workshop on Generative and Experimental Perspectives for Biomolecular Design". For more information, visit https://openreview.net/forum?id=jYaBeydI5P.

generation and fitness prediction tasks. Our results demonstrate that generative ML models training on either Bayesian or maximum likelihood approaches produce highly stable and diverse protein sequences. We also fine-tuned the evolutionary scale ESM protein language model with reconstructed ancestral data to obtain evolutionary-driven protein representations, and downstream stability prediction tasks for Endolysin and Lysozyme C families. For Lysozyme C, ancestral-based representations outperformed the baseline ESM in KNN classification and matched the established InterPro method. In Endolysin, our novel ASR-Dist method performed on par with or better than the baseline and other fine-tuning approaches across various classification metrics. ASR-Dist showed consistent performance in both simple and complex classification models, suggesting the effectiveness of this data-centric approach in enhancing protein representations. This work demonstrates how evolutionary data can improve ML-driven protein engineering, presenting a novel data-centric approach that expands our exploration of protein sequence space and enhances our ability to predict and design functional proteins.

Keywords: Protein Engineering, Generative Models, Language Models, Fine-Tuning, Ancestral Sequence Reconstruction, Data-Centric Models

6.1 Introduction

Recent advancements in machine learning (ML) highlight the critical role of data quality and diversity in model performance, shifting focus towards data-centric approaches that complement algorithmic innovations [1, 2]. This evolving perspective underscores the importance of curating diverse datasets in developing effective ML models. Unlike model-centric methods that are widely accessible through open-source platforms, data-centric methods offer tailored insights to specific applications which enhances model learning capabilities beyond scoring metrics [3][4]. This trend is evident in the development of the GPT model, which evolved from focusing on architectural improvements to prioritizing data quality and data collection strategies [5]. As a result, focusing on data-centric approaches promises a profound contribution toward ML-driven problem-solving in distinct domains. These approaches have already exhibited potential in fields ranging from finance to healthcare, improving model reliability, scalability, trustworthiness, and generalizability

[6, 7, 8, 9, 10]. This shift towards data-centric approaches is especially significant in areas demanding intricate analysis, like protein engineering, where such strategies could enhance the understanding of the complex protein fitness landscape [11, 12].

In protein engineering, the adoption of data-centric approaches is paramount due to the field's unique challenges. With minor alterations to protein sequences significantly impacting functionality and stability, the protein fitness landscape is complex and rugged [13, 14, 15]. Despite advances in applying ML that have facilitated the discovery of high-fitness proteins [16, 17], these models struggle with imbalanced datasets and the scarcity of data across protein families. This underscores the necessity of reevaluating our strategies towards curating training data[11, 12, 18, 19, 20, 21]. A focus on enhancing the diversity and quality of datasets is crucial, as it ensures that ML models are trained on data that accurately reflect the intricate biological properties of proteins. Establishing a method to provide more effective training datasets for ML models will pave the way for strategically advancing protein engineering campaigns.

Ancestral sequence reconstruction (ASR) offers one possible data-centric approach to protein engineering. ASR utilizes computational techniques to infer ancient protein sequences from modern descendants, thereby enriching our datasets with high-quality, diverse, and stable sequences [22, 23, 24, 25]. Built on the foundation of evolutionary biology and molecular phylogenetics, ASR constructs phylogenetic trees using substitution models to map out evolutionary relationships and predict ancestral states. ASR's approach to assigning posterior probabilities to amino acids at various positions allows for the exploration of a vast array of sequence combinations. For example, with just 15 positions that each have four high-likelihood amino acids, ASR can generate up to a billion unique sequences (refer to Figure 6.1).

Recent studies across various protein families have highlighted ASR's effectiveness in dealing with statistical uncertainties [26, 27]. These studies demonstrate that sequences generated from ASR predictions, which sample a broad distribution of amino acids at ambiguously reconstructed sites, can be functional and, in some cases, more efficient or offer novel functionalities compared to sequences based solely on the most likely amino acid predictions. This robustness to uncertainty

is a crucial aspect of ASR's reliability. Even when incorporating alternative amino acids at up to 30% of residues, reconstructed proteins often maintain their overall function **[ref]**. While quantitative parameters such as binding affinities or enzymatic rates may show some variation, qualitative functional characteristics typically remain consistent. This robustness extends to the effects of key historical mutations, which often produce similar functional shifts regardless of the specific ancestral background. Importantly, different methods for incorporating uncertainty, such as the "AltAll" approach (which creates a worst-case scenario protein) and Bayesian sampling, have been employed to test this robustness. These findings suggest that while precise quantitative estimates may vary, qualitative functional inferences from ASR are generally reliable, allowing researchers to make confident claims about ancient protein functions and evolutionary trajectories despite sequence ambiguity. Therefore, with the potential to generate large-scale high-quality sequences, ASR represents a potentially impactful approach to incorporating data-centric strategies in protein engineering. Recent studies have demonstrated the promising potential of leveraging uncertainty in ancestral sequence reconstruction (ASR) within machine learning frameworks for protein engineering. Notably, the work by Colin Jackson's lab (2024) [28, 29] introduced multiplexed ASR (mASR), a novel method that utilizes ASR to obtain protein representations that are family-specific. Their approach employs a custom transformer model (LASE) trained on ancestral sequences reconstructed using the maximum a posteriori (MAP) character at each site. While this method has shown significant improvements in protein representation learning and downstream task performance, particularly demonstrating a smoother fitness landscape and computational efficiency for the phosphodiesterase (PTE) family, it has limitations. The primary disadvantage of the MAP-based approach is its consideration of each site in isolation, potentially overlooking important epistatic effects and dependencies between amino acid sites.

In this study, we explore the integration of ASR-reconstructed data into a comprehensive ML framework to address two primary objectives: (1) the generation of novel protein sequences through generative modeling and (2) the enhancement of protein classification via fine-tuned language models. For the generative ML model, we implemented a variational autoencoder

(VAE) using reconstructed sequence data obtained through both Bayesian (via BAli-Phy) and maximum likelihood (via IQ-TREE) methods in ASR for the ethylene-forming enzyme (EFE) isolated from *Pseudomonas syringae* pv. *phaseolicola* PK2 [30]. The structures and thermal stabilities of the generated sequences were predicted using AlphaFold2 [31] and FoldX [32], respectively. Additionally, sequences generated from ancestral data were compared to those generated from modern sequences by metrics of stability sequence and semantic diversity. We compared sequences generated from ancestral data to those from modern sequences using metrics of structural stability, sequence variability, and semantic diversity. For fine-tuning, we utilized evolutionary scale modeling (ESM) protein representations [33], comparing evolutionary-driven to modern sequences for two protein families: Lysozyme C and Endolysin. The evaluation of family-specific representations on stability prediction tasks revealed promising results across these protein families. Our method, ASR-Dist, demonstrated robust performance in both Lysozyme C and Endolysin, consistently matching or surpassing the baseline models tested. These innovations advance the use of evolutionary information in protein engineering, offering new tools for sequence generation and representation learning, and enhancing exploration of the protein fitness landscape.

6.2 Methods

6.2.1 Generative Model for Novel Protein Sequence Generation

We aimed to test whether incorporating evolutionary data into the training of generative models yields protein sequences that are both novel and structurally stable. To this end, we selected the ethylene-forming enzyme (EFE) from PK2 as our focus and employed a VAE as our generative ML model. Subsequent computational analyses (i.e., AlphaFold [31], FoldX [32], UMAP visualization [34]) were conducted on the sequences generated by this model to assess their quality.

6.2.1.1 Data Processing and Ancestral Sequence Reconstruction

We extracted the evolutionary information of EFE using our AP-LASR [35] software, a tool designed to reconstruct ensembles of ancestral proteins by leveraging the phylogenetic tree of the query protein sequence. This tool has facilitated ASR by fully automating the reconstruction process from initial BLAST search, multiple sequence alignment by MAFFT [36] to tree phylogeny

predictions via IQ-TREE2 [37].

AP-LASR outputs several key datasets: sequences of modern proteins, ancestral proteins (ASR-Max), and near-ancestral proteins (ASR-Dist). The ASR-Dist dataset is created by sampling from a posterior probability distribution in the ASR.state file that was generated via IQ-TREE. For our project, the threshold for picking amino acids from this distribution was set at 0.2 to strike an optimal balance between maintaining ancestral properties and sequence diversity (i.e., amino acids with a posterior probability greater than 0.2 for a given position were included for sampling in the dataset)[26, 38]. From the data generated by AP-LASR, we sampled the sequences from four nodes that represented high-stability ancestors obtained from various evolutionary timescales Node10, Node13, Node 253, and Node 384 (Figure S1).

To understand how sequence diversity in our training set would impact the robustness and generalizability of our model, two distinct datasets of ancestral protein sequences were crafted for training our ML model. The "Homogeneous Dataset" is comprised of sequences equally sampled from ancestral nodes, whereas the "Diverse Dataset" was created by passing the initial dataset through CD-Hit with a 0.9 similarity threshold [39]. The sampled sequences for each node were initially aligned using MAFFT [21]. Subsequently, the sequences from all ancestral nodes were pooled and re-aligned with MAFFT.

We compared a Bayesian inference method (BAli-Phy) against a maximum likelihood approach (IQ-TREE) to evaluate how the choice of probabilistic sequence reconstruction methods influences generative model performance. BAli-Phy uses a Bayesian inference method to simultaneously estimate of sequence alignment, tree phylogeny, and model parameters, providing full posterior distributions of ancestral sequences. It does not assume a fixed alignment or tree topology, making it more accurate and computationally intensive.

In brief, the same multiple sequence alignment generated by AP-LASR for IQ-TREE was used to initialize eight independent Markov chain Monte Carlo (MCMC) runs in BAli-Phy. The optimal amino acid substitution model for the runs, lg08 [40], was determined using IQ-TREE's ModelFinder [41]. After running simulations for seven days, three runs were selected for convergence

by minimizing the average standard deviation of split frequencies (ASDSF) and the maximum of the standard deviation of split frequencies (MSDSF). For convergence analysis, the first 2,000 iterations of each run were discarded to account for the bias of the initial random starting guess (i.e., burn-in). The ensemble of ancestors at each ancestral node were subsequently retrieved from the BAli-Phy runs, which produces a single alignment for every iteration which can be sampled. ETE4 Toolkit [42] was used to navigate the phylogeny and define the children sequences for all ancestral nodes[2], and Dendropy [43] was used to retrieve all ancestral nodes from the BAli-Phy output[3]. For more targeted protein generation, we sampled the sequences from four nodes representing high-stability ancestors obtained from various evolutionary timescales (Node10, Node13, Node253, and Node384, as shown in Figure S6.1).

6.2.1.2 Generative Model Training

Variational Autoencoder (VAE) was selected for its proficiency in generating new data points that are coherent with the training data [44, 45]. We employed one-hot encoding to transform the sequences into a format suitable for computational processing. Feature extraction from these one-hot encoded sequences was performed using a 1D Convolutional Neural Network (CNN) layer, allowing us to capture the local sequence patterns effectively. The architecture of our VAE was designed with a latent space dimensionality of 100, ensuring sufficient complexity to capture the nuances of protein sequence variability. Additionally, we incorporated batch normalization within the network to facilitate smoother and more stable learning dynamics. This combination of 1D CNN for feature learning and batch normalization for optimization contributed to refining the model's ability to generate meaningful protein sequences.

6.2.1.3 Evaluation

To gauge the ability of our models to generate stable and viable sequences, we utilized AlphaFold2 to verify that the predicted 3D structures of the generated PK2 ancestors exhibited folding patterns similar to the wild-type structure. Randomly selected sequences generated by AlphaFold2 were superimposed onto wild-type structures (PDB ID: 5V2Y) using PyMol

[2]https://github.com/etetoolkit/ete
[3]https://github.com/jeetsukumaran/DendroPy

to calculate RMSD and compare folding patterns. Following structural prediction, stability calculations (ddG) were carried out using FoldX. This evaluation phase was crucial, as it allowed us to ascertain not just the novelty of the generated sequences but also their practical applicability in terms of structural integrity and thermal stability. We then compared the distribution of generated structure stability measurements followed by the Dunn statistical test for measuring the obtained results' significance. The sequences generated by a model trained on evolutionary-reconstructed sequences were compared to those generated by a model trained on modern sequences. The comparison focused on three aspects: structural quality, sequence diversity, and semantic diversity.

To assess structural quality, we used the predicted Local Distance Difference Test (pLDDT) scores which were extracted using the alphapickle library [4]. Ranging from 0 to 100, pLDDT is used to assess the reliability of the predicted atomic positions in the protein structure. We used the Levenshtein edit distance [46, 47](i.e., the minimum number of single-character edits required to change one string into another) to quantify the divergence between training sequences and generated sequences across different datasets. This allowed us to assess the sequence variability introduced by the generation process. To characterize semantic diversity, which takes into account the biological or chemical properties of the sequences rather than just their raw sequence differences, the sequence representations obtained via ESM language model were visualized with UMAP, a dimensional reduction technique that represents the data manifold in lower dimensions [34].

6.2.2 Evolutionary-Derived Protein Sequence Representation

In this section, we detail our approach to creating family-specific protein representations using fine-tuned methods. We divide the fine-tuning into two categories: regular fine-tuning, which adjusts model parameters using the InterPro protein family dataset, and evolutionary fine-tuning, which incorporates specific evolutionary data relevant to protein families to guide the tuning process. The performance of these fine-tuned representations is evaluated by comparing them against the baseline ESM2 (Evolutionary Scale Modeling) [33] representation in a protein stability prediction task.

[4] https://github.com/mattarnoldbio/alphapickle

6.2.2.1 Data Processing

We fine-tuned the ESM2 for Endolysin and Lysozyme C, for which we obtained labeled datasets for stability prediction in FireProtDB [5]. Our data processing involved assembling three distinct unlabeled datasets to fine-tune the ESM model for each protein family: (i) a collection of InterPro-derived sequences that encompass an expanded set of modern proteins based on family affiliations [6], (ii) a collection of the single most-probable ancestors (ASR-Max) taken from every internal node of the phylogenetic tree built from modern sequence BLAST results, and (iii) a collection of ancestral ensembles taken from internal nodes (ASR-Dist). The ASR-Dist approach enriches the dataset with a broader spectrum of evolutionary possibilities, offering a more robust dataset that surpasses ASR-Max in both diversity and volume. This comprehensive dataset integrates extensive evolutionary insights, significantly enhancing the model's training base. We sampled 1,000 sequences inferred from each high-quality ancestral node (which we defined as having SH-aLRT > 80% and ultrafast bootstrapping > 95%) reconstructed in ASR and removed repeat sequences. It is important to note, depending on the application and computational resources, that the number of ASR-Dist sequences can be tuned by modifying the number of sequences sampled from each node in the phylogenetic tree. Table 1 represents dataset information for both protein families tested. The prediction task was stability classification (i.e., determining if a given sequence is stable with a $\Delta\Delta G < -0.5$ kcal/mol or unstable with a $\Delta\Delta G > 0.5$ kcal/mol).

Table 6.1 Dataset Quantification for Fine-Tuning Task.

Protein	**Interpro ID**	**Interpro Seqs**	**ASR-MAX Seqs**	**ASR-Dist Seqs**	**Classification Seqs (FireProtDB)**
Endolysin	IPR034690	12K	302	40K	647
Lysozyme C	IPR036328	9K	452	36K	338

6.2.2.2 Classification Model Training

For model fine-tuning, we employed the ESM2 model (esm2_t12_35M_UR50D) trained with 35M parameters which generates 480 embedding dimensions and contains 12-layer representations.

[5] https://loschmidt.chemi.muni.cz/fireprotdb/
[6] https://www.ebi.ac.uk/interpro/

We unfroze its last two layers to adapt its learning to our specific datasets. A batch size of 32 sequences was utilized to optimize the training process, alongside the implementation of early stopping to mitigate the risk of over-fitting. This fine-tuning phase was critical, allowing us to tailor the model to our evolutionary-informed datasets. Post-tuning, we extracted the embedding from each of the four datasets to further refine our approach to protein family classification, employing KNN, Random Forest, and XGBoost algorithms to assess the model's predictive performance within each representation derived from distinct fine-tuning methods.

6.2.2.3 Evaluation of Classification Models

The evaluation of our fine-tuned language model focused on its ability to classify protein families accurately, employing a suite of classification metrics to gauge performance comprehensively. Precision, recall, balanced accuracy, Area Under the Curve (AUC), and the F1 score were calculated for each protein family (Endolysin and Lysozyme C) across the representations. For more robust training, we performed 5-fold cross-validation for the datasets. Then the trained models were tested on a held-out test set which was 30% of the initial data. Note that, for robustness, we repeated this on 20 distinct random states and reported the mean and standard deviation for the obtained results among all the classification scores.

6.3 Results

The critical importance of high-quality data in protein engineering, particularly for structure prediction and improving generalization metrics, is widely recognized. This paper explores the potential of integrating underutilized yet rich evolutionary information to enhance both generative models and language models in the field. For generative models, the selection of training data that accurately captures the prior distribution is crucial in determining the quality of the sequences produced. In parallel, the efficacy of fine-tuning language models, a state-of-the-art method for ML predictive tasks, is intrinsically linked to the caliber of the dataset used for refinement. This investigation aims to demonstrate how evolutionary information can be strategically employed in ML platforms to facilitate more informed exploration and navigation of the protein fitness landscape. By applying this approach to both sequence generation and protein classification tasks, the study

seeks to leverage evolutionary data in protein engineering, potentially opening new avenues for designing proteins with enhanced stability and functionality.

6.3.1 Ancestral Sequence Generation Using Different Probabilistic Methods: Maximum Likelihood vs Bayesian

This section explores the differences in the qualities of generated sequences obtained using Bayesian inference (BAli-Phy) and maximum likelihood approaches (IQ-TREE) for ancestral sequence reconstruction. Our objective was to understand how different methods for reconstructing ancestral sequences impact the quality of the reconstructed sequences and the sequences generated using the ancestral sequences as the training dataset. While BAli-Phy employs a Bayesian approach to simultaneously reconstruct the phylogenetic tree and sequence alignment, it is more computationally intensive compared to IQ-TREE, which uses a maximum likelihood approach. Our results, presented in Figure 6.2, illustrate the stability, sequence variability, and semantic diversity of the proteins generated using these two methods. Notably, we utilized all reconstructed nodes from both methods (i.e., without selecting for nodes with high stability) to provide a comprehensive overview of their performance. The comparison revealed no significant differences in sequence variability, semantic diversity patterns and stability profiles between the two methods.

The comparable performance of IQ-Tree and BAli-Phy across stability, variability, and semantic diversity metrics challenges expectations based on their distinct mathematical approaches (i.e., maximum likelihood and Bayesian inference). A key factor potentially contributing to this similarity is the sampling strategy. By incorporating sequences from all nodes in the phylogenetic tree, both methods likely capture a wide range of ancestral states, including those with varying degrees of certainty. This comprehensive sampling may balance out the theoretical advantages of each approach, resulting in similar overall performance. The stability results indicate that both methods generate sequences with comparable thermodynamic properties, suggesting IQ-Tree's point estimates are as robust as BAli-Phy's distribution-based estimates for maintaining protein stability. The equivalent sequence variability and semantic diversity imply that IQ-Tree's maximum likelihood approach explores sequence space as effectively as BAli-Phy's Bayesian method.

For the EFE family from PK2, our results show that IQ-Tree and BAli-Phy perform similarly across key metrics. Given this, we chose to use IQ-Tree for further analyses due to its faster computation time. To improve reconstruction quality, we focused on sampling from high-confidence nodes in the phylogenetic tree. This approach combines IQ-Tree's efficiency with a strategy to minimize uncertainty in our ancestral sequence predictions.

6.3.2 Incorporating Evolutionary Information in Sequence Generation Results in Novel & Stable Proteins

Novel sequences were generated by sampling from the learned latent representations after loss minimization in the validation set. Three different sets of training data (modern, Ancestral Type1(homogeneous), Ancestral Type2(diverse)) were used to generate distinct sets of sequences. A representative subset of sequences (1,000 sequences per data type) was randomly sampled from each dataset for both training and generation populations. Our study's findings are promising, revealing that: (i) our datasets not only augment the volume of training data through the innovative integration of uncertainty in ML but also provide a richer set of sequences with inherently higher stability for training purposes. Moreover, (ii) the stability distribution of the sequences generated using ancestral data aligns closely with those of the training set, attesting to the potential of our method to replicate high-quality protein stability profiles in novel sequence creation. Also, the mean RMSD values were 0.43 for modern-derived structures, 0.42 for ancestral-derived structures (Ancestral 1), and 0.39 for further ancestral-derived structures (Ancestral 2), indicating that the generated structures maintained folding patterns similar to the wild-type. Figure 6.3 represents a comparative analysis of the thermal stability of proteins derived from both ancestral and modern sequences, examining stability within the training data and the sequences generated via VAE. The findings underscore the significant potential of evolutionary protein sequences. Within the training dataset, ancestral proteins exhibited markedly higher stability relative to modern proteins. Furthermore, the sequences generated from these ancestral proteins retained this enhanced stability. Another notable observation is the pronounced distribution shift towards increased stability in the generated sequences, particularly when using modern sequences as compared to ancestral ones.

Although the precise mechanisms underlying this shift remain unclear, it is possible that the generative model, when applied to modern sequences, mitigates noise from low-quality sequences (in terms of epistasis and folding). Conversely, the ancestral training data may inherently encompass stabilizing interactions. Consequently, the generated sequences from ancestral proteins consistently demonstrate superior stability compared to those derived from modern sequences.

We evaluated our generated proteins based on semantic diversity (i.e., the distance between training and generated sequences) and structural qualities (i.e., uncertainty quantification and intrinsic disorder region distributions). The results of these assessments are presented in Figure 6.4.

The comparative analysis of ancestral and modern PK2 sequences reveals several key differences. Ancestral sequences demonstrate improved structural stability, evidenced by fewer intrinsically disordered regions and consistently higher pLDDT scores. They also exhibit greater sequence conservation, as indicated by lower edit distances between training and generated datasets. Despite this conservation, ancestral sequences display broader semantic diversity in the UMAP projection. In contrast, modern sequences show more variability in structural quality and sequence composition but form tighter clusters in semantic space. These findings highlight the potential benefits of incorporating evolutionary data in generative models for protein design. Ancestral sequences appear to combine desirable properties such as structural stability with an expanded exploration of sequence space. This unique combination could prove valuable in generating novel protein variants that maintain crucial ancestral characteristics while introducing functional innovations. By leveraging the broader semantic diversity of ancestral sequences, generative models could potentially access a wider range of protein designs, opening up new avenues for engineering proteins with enhanced or altered functions while preserving their fundamental structural integrity.

6.3.3 Evolutionary protein representations demonstrate significant potential for enhancing classification tasks.

As detailed in the methods section, we fine-tuned the ESM2 model on three distinct datasets to obtain protein representations termed Inter-Pro, ASR-Max, and ASR-Dist. Intriguingly, the

representations derived from fine-tuning ESM2 with ancestral data exhibited comparable or enhanced performance relative to those derived from modern data in both predictive stability tasks,determining if a given sequence is stable ($\Delta\Delta G < -0.5\ kcal/mol$) or unstable ($\Delta\Delta G > 0.5\ kcal/mol$) for Endolysine and Lysozyme C proteins. The outcomes for both protein families are presented in Table 6.2 and Table 6.3. Our ASR-Dist method, which leverages uncertainty in ASR, has shown improved performance over other representations for KNN classification. However, it demonstrated comparable or improved performance over InterPro-derived representations for ensemble-based classifiers (i.e., Random Forest and XGBoost).

Table 6.2 Comparison of Classifier Performance Across Fine-Tuned Representations – Lysozyme C.

Classifier	**Dataset**	**Score (Mean±STD)**				
		Balanced Accuracy	**F1**	**Precision**	**Recall**	**ROC_AUC**
KNN	ESM-Base	0.51±0.02	0.05±0.07	0.23±0.33	0.03±0.04	0.49±0.07
	Interpro	0.70±0.05	0.54±0.10	0.74±0.12	0.44±0.12	0.81±0.04
	ASR-Max	0.71±0.05	0.57±0.10	**0.90±0.12**	0.43±0.10	0.81±0.04
	ASR-Dist	**0.73±0.05**	**0.61±0.09**	0.83±0.13	**0.50±0.09**	**0.86±0.05**
Random Forest	ESM-Base	0.51±0.02	0.06±0.07	0.38±0.45	0.03±0.04	0.52±0.06
	Interpro	0.75±0.06	0.64±0.10	0.80±0.09	0.55±0.13	0.92±0.02
	ASR-Max	0.73±0.06	0.60±0.10	0.78±0.11	0.50±0.12	0.93±0.02
	ASR-Dist	0.75±0.05	0.64±0.09	0.82±0.11	0.54±0.11	0.94±0.02
XGBoost	ESM-Base	0.51±0.01	0.03±0.05	0.23±0.37	0.02±0.03	0.53±0.06
	Interpro	0.78±0.05	0.67±0.08	0.80±0.10	0.60±0.13	0.94±0.02
	ASR-Max	0.77±0.05	0.67±0.09	0.78±0.12	0.60±0.09	0.94±0.03
	ASR-Dist	0.78±0.05	0.67±0.07	0.77±0.10	0.60±0.10	0.93±0.03

Table 6.3 Comparison of Classifier Performance Across Fine-Tuned Representations – Endolysin.

Classifier	**Dataset**	**Score (Mean±STD)**				
		Balanced Accuracy	**F1**	**Precision**	**Recall**	**ROC_AUC**
KNN	ESM-Base	**0.82±0.05**	**0.73±0.08**	0.85±0.08	0.66±0.10	**0.93±0.03**
	Interpro	0.77±0.05	0.66±0.09	0.84±0.10	0.55±0.10	0.90±0.03
	ASR-Max	0.80±0.05	0.69±0.08	0.78±0.09	0.62±0.09	0.91±0.04
	ASR-Dist	**0.82±0.05**	**0.73±0.08**	0.84±0.09	0.65±0.10	**0.94±0.03**
Random Forest	ESM-Base	0.83±0.05	0.75±0.08	0.84±0.08	0.68±0.10	0.96±0.02
	Interpro	0.82±0.04	0.73±0.07	0.85±0.06	0.65±0.09	0.96±0.02
	ASR-Max	0.83±0.05	0.74±0.07	0.83±0.07	0.67±0.09	0.96±0.02
	ASR-Dist	0.84±0.04	0.77±0.06	0.85±0.07	0.70±0.08	0.97±0.02
XGBoost	ESM-Base	0.85±0.04	0.77±0.05	0.85±0.06	0.72±0.08	0.95±0.04
	Interpro	0.84±0.05	0.76±0.07	0.85±0.07	0.70±0.09	0.95±0.04
	ASR-Max	0.84±0.05	0.75±0.07	0.83±0.07	0.69±0.10	0.95±0.03
	ASR-Dist	0.85±0.04	0.78±0.06	0.84±0.07	0.72±0.07	0.94±0.04

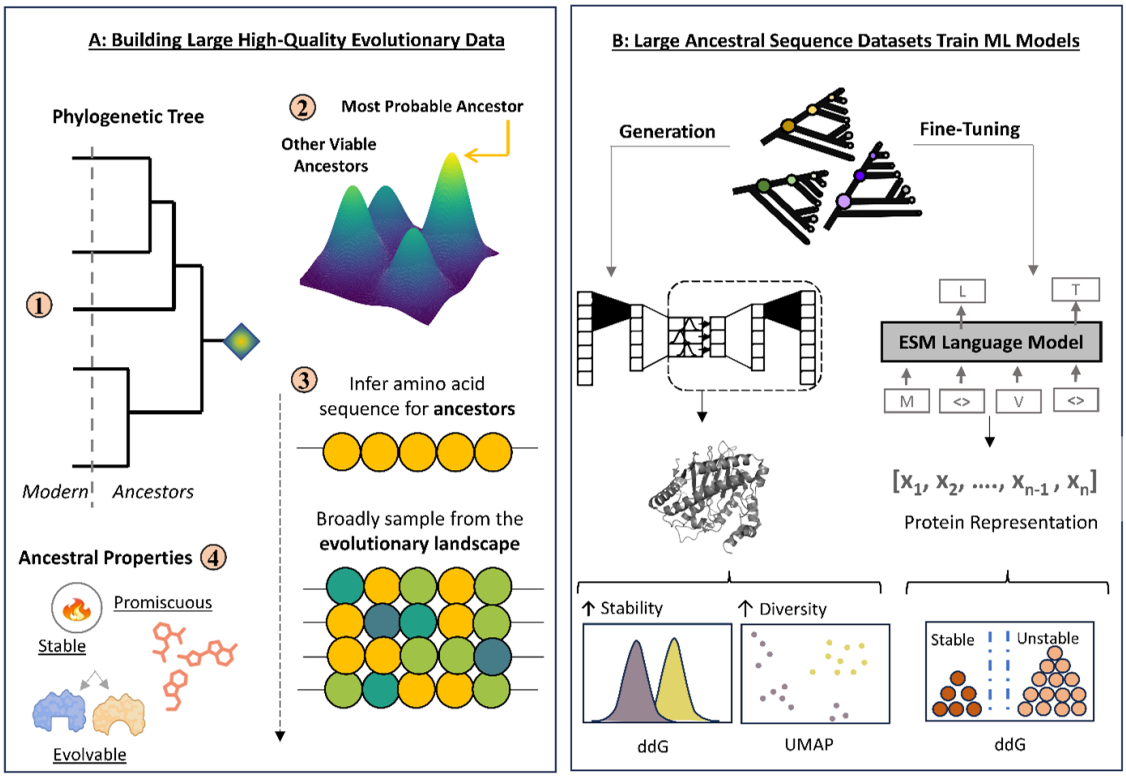


Figure 6.1 A. Generation of high-quality evolutionary data. The phylogenetic tree for the family of interest is generated via ASR to access the ancestral sequences. Ancestral sequences are often known to be stable, promiscuous, and evolvable. To generate ancestral sequences, IQ-TREE assumes a fixed alignment and tree topology, then employs marginal reconstruction, which calculates the most likely state (amino acid), shown as yellow nodes, for each site (residue position number) independently. Ensembles of ancestral sequences can then be produced by sampling from other probable amino acids at individual positions of the ancestral sequences. Additionally, BAli-Phy is used for sequence reconstruction to co-estimate phylogeny and alignment, allowing for simultaneous inference of multiple parameters, providing a robust framework for evolutionary analysis. In brief, the same multiple sequence alignment generated by AP-LASR for IQ-TREE is used to initialize eight independent Markov chain Monte Carlo (MCMC) runs in BAli-Phy. The optimal amino acid substitution model for the runs is determined using IQ-TREE's ModelFinder. After running simulations for seven days, three runs are selected for convergence by minimizing the average standard deviation of split frequencies (ASDSF) and the maximum of the standard deviation of split frequencies (MSDSF). For convergence analysis, the first 2,000 iterations of each run are discarded to account for the bias of the initial random starting guess (i.e., burn-in). The ensemble of ancestors at each ancestral node is subsequently retrieved from the BAli-Phy runs, which produces a single alignment for every iteration that can be sampled. B. The obtained evolutionary information is used as training data for sequence generation and family-specific protein sequence representation.

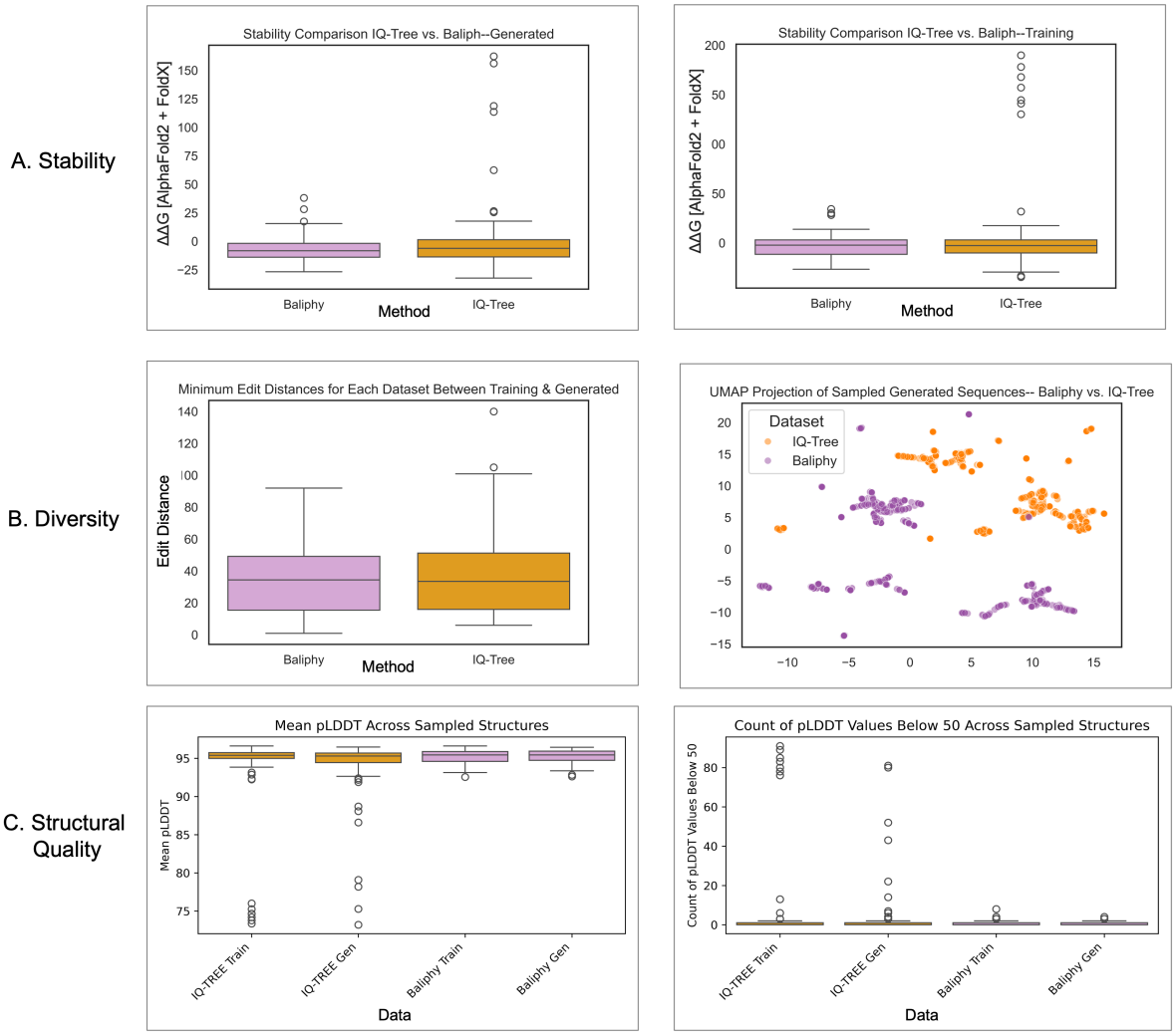


Figure 6.2 Maximum Likelihood (IQ-Tree) and Bayesian (BAli-Phy) approaches for ancestral sequence reconstruction of the EFE family from PK2 are comparable in terms of thermostability, sequence variability, and structural quality. A.Stability analysis using FoldX predictions of thermodynamic stability (($\Delta\Delta G$, kcal/mol) between generated and training sequences. No statistically significant differences were observed (p-value=0.17). B. Sequence variability illustrated by minimum edit distances between training and generated sequences, reveals similar diversity for both methods. Similarly, semantic diversity visualized via UMAP projections of sampled generated sequences indicate comparable patterns. C. Structural quality analysis using pLDDT scores demonstrated no significant differences between methods (p-value=0.37 for means and p-value=0.33 for IDRs).

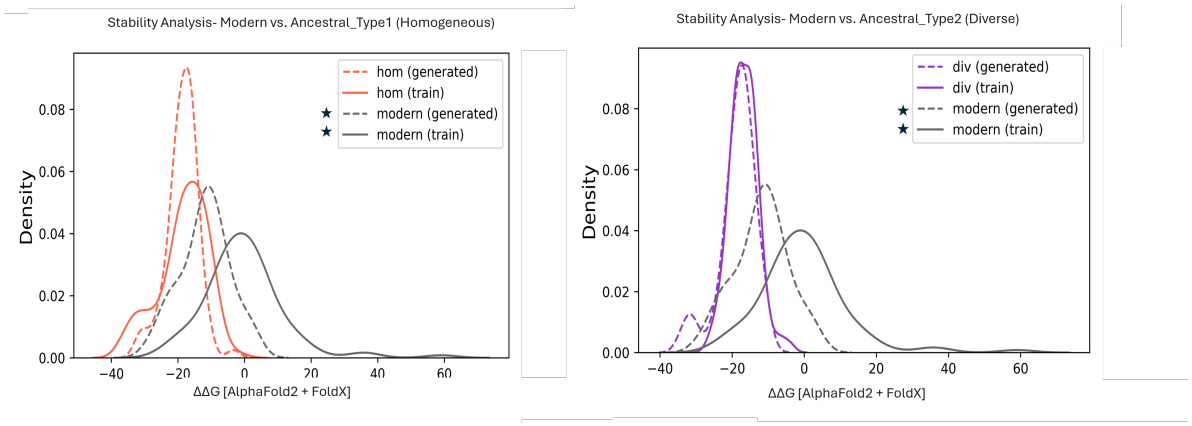


Figure 6.3 ASR-derived PK2 sequences exhibited higher thermal stability compared to modern sequences (both before and after generation via VAE.) A. Stability Analysis Modern vs. Ancestral_Type1 (Homogeneous). The density plot shows the thermodynamic stability ($\Delta\Delta G$ values - stability measurement relative to the stability of wild-type (ΔG)) of modern and homogeneous ancestral (AncType1) protein sequences. B. Stability Analysis Modern vs. Ancestral_Type2 (Diverse). Ancestral-based models consistently produced sequences with improved stability compared to modern sequences. Generated sequences closely mirror the stability profiles of their respective training sets. Notably, the generated sequences from both ancestral types showed no statistically significant difference from their training sets (p-values > 0.05), while differing significantly from modern sequences (p-values < 1e-8). The statistical significance of all comparisons is provided in Supplementary Table 6B.1.

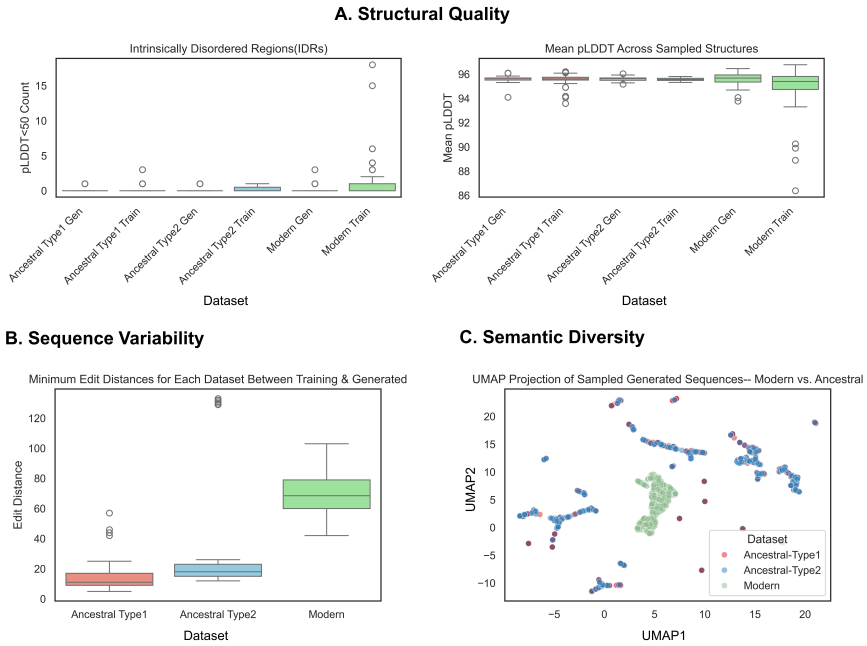


Figure 6.4 The Generation-Quality panel. Structural and sequence properties of ancestral and modern PK2 sequences before and after generative modeling. A) Structural Quality: Intrinsically Disordered Regions (IDRs) measured by pLDDT<50 count and mean pLDDT scores across sampled structures, showing fewer IDRs and higher mean pLDDT scores in ancestral sequences. B) Sequence Variability: Minimum edit distances between training and generated datasets, revealing significantly higher variability in modern sequences (p-value < 0.05). C) Semantic Diversity: UMAP projection of sampled generated sequences, illustrating distinct clustering patterns with ancestral sequences exhibiting broader distribution. Ancestral sequences demonstrate improved structural stability, more conserved sequence diversity, and increased semantic diversity compared to modern sequences, while modern sequences show higher sequence variability before and after training.

6.4 Discussion

In this work, we explored the integration of evolutionary information into machine learning models for protein engineering. Our approach combined ancestral sequence reconstruction (ASR) with generative modeling and language model fine-tuning to enhance both sequence generation and fitness prediction. The results demonstrated that sequences generated from evolutionary-informed datasets exhibited improved characteristics, particularly in thermal stability, compared to those derived from models trained solely on modern sequences. This highlights the potential of ASR to enrich training data for generative models, leading to more diverse and higher-quality outputs.

While our study provided valuable insights, it's important to note that we only scratched the surface of the vast sequence space available through ASR. For instance, considering the 186 high-quality nodes from our Lysozyme ancestral tree reconstruction, with 15 variable positions and 4 possible amino acids per position, the theoretical sequence space encompasses approximately 200 billion unique sequences. Our approach, utilizing clustering algorithms and strategic sampling methods, aimed to capture a representative subset of this diversity. Despite these efforts to efficiently sample the sequence space, there remains immense potential for further exploration and optimization of these evolutionary-informed datasets. Future work could focus on developing more sophisticated sampling techniques or exploring a larger portion of the available sequence space to potentially uncover even more beneficial protein variants.

To evaluate the utility of ancestral sequences on downstream stability prediction tasks, we fine-tuned the evolutionary scale ESM protein language model with reconstructed ancestral data to obtain evolutionary-driven protein representations for Endolysin and Lysozyme C families. For Lysozyme C, ancestral-based representations showed promising results, outperforming the baseline ESM in KNN classification and matching the established InterPro method for other classification models. In Endolysin, our novel ASR-Dist method performed comparably to the baseline and other fine-tuning approaches across various classification metrics. While not consistently outperforming existing methods, ASR-Dist demonstrated stable performance across both simple and complex classification models, suggesting potential in this data-centric approach for enhancing protein

representations.

This study advances protein engineering by integrating diverse ancestral sequence reconstruction (ASR) techniques into generative modeling and representation learning. It demonstrates the generation of sequences with ancestral-like properties using machine learning, a novel achievement in the field. While previous work like local ancestral sequence embedding (LASE) introduced ASR in representation learning using transformers trained from scratch, this approach explores fine-tuning existing models with various evolutionary and modern sequence datasets, addressing potential over-fitting issues and offering an accessible framework for researchers. Another key innovation is the implementation and comparison of two probabilistic approaches for phylogeny inference and ancestral sequence reconstruction: maximum likelihood and Bayesian inference. This comprehensive incorporation of ASR-driven protein sequences and their impact on both generative modeling and representation learning provides a nuanced view of evolutionary information's role in protein engineering.

While this study illuminates the potential of integrating evolutionary data into ML models, it also highlights the necessity for further investigation. We computationally validated the stability and diversity of the sequences generated for the generative task, but experimental validation is crucial to understanding the functional advantages that evolutionary signals may confer.

BIBLIOGRAPHY

[1] P. Singh. Systematic review of data-centric approaches in artificial intelligence and machine learning. *Data Sci. Manag.*, 6:144–157, 2023.

[2] J. Adeoye, L. Hui, and Y.X. Su. Data-centric artificial intelligence in oncology: A systematic review assessing data quality in machine learning models for head and neck cancer. *J. Big Data*, 2023.

[3] L. Miranda. Study notes on data-centric machine learning, 2023.

[4] D. Zha and U. States. Data-centric artificial intelligence: A survey, 2023.

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[6] H. Liu, M. Chaudhary, and H. Wang. Towards trustworthy and aligned machine learning, 2023.

[7] A.X. Wang, S.S. Chukova, C.R. Simpson, and B.P. Nguyen. Data-centric ai to improve early detection of mental illness. In *IEEE Work. Stat. Signal Process. Proc.*, volume 2023-July, pages 369–373, 2023.

[8] Johannes Jakubik, Michael Vössing, Niklas Kühl, Jannis Walk, and Gerhard Satzger. Data-centric artificial intelligence, 2024.

[9] Yunkun Zhang, Jin Gao, Zheling Tan, Lingfeng Zhou, Kexin Ding, Mu Zhou, Shaoting Zhang, and Dequan Wang. Data-centric foundation models in computational healthcare: A survey. *arXiv.org*, 2024.

[10] John Adeoye, L. M. Christy Hui, and Yu xiong Su. Data-centric artificial intelligence in oncology: a systematic review assessing data quality in machine learning models for head and neck cancer. *Journal of Big Data*, 2023.

[11] P.A. Romero and F.H. Arnold. Exploring protein fitness landscapes by directed evolution. *Nat. Rev. Mol. Cell Biol.*, 10:866–876, 2009.

[12] S.A. Kauffman and E.D. Weinberger. The nk model of rugged fitness landscapes and its application to maturation of the immune response. In *Mol. Evol. Rugged Landscapes Proteins, RNA Immune Syst.*, pages 135–175, 2018.

[13] Sonja Billerbeck. A computational walk to the hidden peaks of protein performance. *Synthetic Biology*, 2021.

[14] Mehrsa Mardikoraem and Daniel R. Woldring. Protein fitness prediction is impacted by the interplay of language models, ensemble learning, and sampling methods. *Pharmaceutics*, 2023.

[15] Shimon Bershtein, Michal Segal, Roy Bekerman, Nobuhiko Tokuriki, and Dan S. Tawfik. Robustness–epistasis link shapes the fitness landscape of a randomly drifting protein. *Nature*, 2006.

[16] Kevin K Yang, Zachary Wu, and Frances H Arnold. Machine-learning-guided directed evolution for protein engineering. *Nature methods*, 16(8):687—694, August 2019.

[17] Yuchi Qiu, Jian Hu, and Guo-Wei Wei. Cluster learning-assisted directed evolution. *Nature computational science*, 1(12):809–818, 2021.

[18] M.A. Mena and P.S. Daugherty. Automated design of degenerate codon libraries. *Protein Eng. Des. Sel.*, 18:559–561, 2005.

[19] W. Gao, S.P. Mahajan, J. Sulam, and J.J. Gray. Deep learning in protein structural modeling and design. *Patterns*, 1:100142, 2020.

[20] Iman Deznabi, Bulent Arabaci, Mehmet Koyuturk, and Oznur Tastan. Deepkinzero: Zero-shot learning for predicting kinase-phosphosite associations involving understudied kinases. *Bioinformatics*, 36:3652–3661, 2020.

[21] Joshua Meier, Roshan Rao, Reinier Verkuil, Jesse Liu, Tom Sercu, and Antoine Rives. Language models enable zero-shot prediction of the effects of mutations on protein function. *bioRxiv*, 2021(07):2021.07.09.450648, 2021.

[22] Yvonne Joho, Vanessa Vongsouthi, Matthew A. Spence, Jennifer Ton, Chloe Gomez, Li Lynn Tan, Joe A. Kaczmarski, Alessandro T. Caputo, Santana Royan, Colin J. Jackson, and Albert Ardevol. Ancestral sequence reconstruction identifies structural changes underlying the evolution of ideonella sakaiensis petase and variants with improved stability and activity. *Biochemistry*, 62(2):437–450, 2023.

[23] Yoseph Gumulya and Elizabeth MJ Gillam. Exploring the past and the future of protein evolution with ancestral sequence reconstruction: The "retro" approach to protein engineering. *Biochem J*, 474:1–19, 2017.

[24] Jonas Zaugg, Yoseph Gumulya, Elizabeth MJ Gillam, and Mikael Bodén. Chapter 21 of prospective and retrospective strategies. 2014.

[25] Linus Pauling, Emile Zuckerkandl, Thorkil Henriksen, and Rolf Lövstad. Chemical paleogenetics. *Acta chem scand*, 17:S9–S16, 1963.

[26] Geeta N Eick, Anderson David P Harms Michael J Bridgham, Jamie T, and Joseph W

Thornton. Robustness of reconstructed ancestral protein functions to statistical uncertainty. *Mol Biol Evol*, 34:247–261, 2017.

[27] Hila Bar-Rogovsky, Adi Stern, Osnat Penn, Itay Kobl, Tal Pupko, and Dan S Tawfik. Assessing the prediction fidelity of ancestral reconstruction by a library approach. *Protein Eng Des Sel*, 28:507–518, 2015.

[28] Dana M Matthews, Matthew A Spence, Adam C Mater, James Nichols, Sacha B Pulsford, Mahakaran Sandhu, Joe A B Kaczmarski, Charlotte M Miton, Nobuhiko Tokuriki, and Colin J Jackson. Leveraging ancestral sequence reconstruction for protein representation learning. *bioRxiv*, pages 2023–12, 2023.

[29] Vanessa Vongsouthi, Rosemary Georgelin, Dana Matthews, Jake Saunders, Brendon M Lee, Jennifer Ton, Adam M Damry, Rebecca L Frkic, Matthew A Spence, and Colin J Jackson. Ancestral reconstruction of polyethylene terephthalate degrading cutinases reveals a rugged and unexplored sequence-fitness landscape. *bioRxiv*, pages 2024–04, 2024.

[30] Kazuhiro Nagahama, Takahira Ogawa, Takao Fujii, Masato Tazaki, Sumio Tanase, Yoshimasa Morino, and Hideo Fukuda. Purification and properties of an ethylene-forming enzyme from pseudomonas syringae pv. phaseolicola pk2. *Microbiology*, 137(10):2281–2286, 1991.

[31] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

[32] Joost Schymkowitz, Jesper Borg, Francois Stricher, Robby Nys, Frederic Rousseau, and Luis Serrano. The foldx web server: an online force field. *Nucleic acids research*, 33(suppl_2):W382–W388, 2005.

[33] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

[34] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[35] James VanAntwerp, Mehrsa Mardikoraem, Nathaniel Pascual, and Daniel Woldring. Ap-lasr: Automated protein libraries from ancestral sequence reconstruction. *bioRxiv*, 2023.

[36] K. Katoh and D.M. Standley. Mafft multiple sequence alignment software version 7: improvements in performance and usability. *Molecular Biology and Evolution*, 30(4):772–780, April 2013. Epub 2013 Jan 16.

[37] Bui Quang Minh, Heiko A Schmidt, Olga Chernomor, Dominik Schrempf, Michael D Woodhams, Arndt von Haeseler, and Robert Lanfear. IQ-TREE 2: New Models and Efficient Methods for Phylogenetic Inference in the Genomic Era. *Molecular Biology and Evolution*, 37(5):1530–1534, 02 2020.

[38] M.A. Sennett and D.L. Theobald. Extant sequence reconstruction: The accuracy of ancestral sequence reconstructions evaluated by extant sequence cross-validation. *Journal of Molecular Evolution*, 92:181–206, 2024.

[39] Weizhong Li and Adam Godzik. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22(13):1658–1659, 2006.

[40] Si Quang Le and Olivier Gascuel. An Improved General Amino Acid Replacement Matrix. *Molecular Biology and Evolution*, 25(7):1307–1320, 03 2008.

[41] Subha Kalyaanamoorthy, Bui Quang Minh, Thomas KF Wong, Arndt Von Haeseler, and Lars S Jermiin. Modelfinder: fast model selection for accurate phylogenetic estimates. *Nature methods*, 14(6):587–589, 2017.

[42] Jaime Huerta-Cepas, François Serra, and Peer Bork. Ete 3: reconstruction, analysis, and visualization of phylogenomic data. *Molecular biology and evolution*, 33(6):1635–1638, 2016.

[43] Matthew Andres Moreno, Jeet Sukumaran, and Mark T. Holder. Dendropy 5: a mature python library for phylogenetic computing, 2024.

[44] C. Ziegler, Claude Sinner, and Faruck Morcos. Latent generative landscapes as maps of functional diversity in protein sequence space. *Nature Communications*, 2023.

[45] Mehrsa Mardikoraem, Zirui Wang, Nathaniel Pascual, and Daniel Woldring. Generative models for protein sequence modeling: recent advances and future directions. *Briefings in Bioinformatics*, 24(6):bbad358, 10 2023.

[46] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[47] Gonzalo Navarro. A guided tour to approximate string matching. *ACM Computing Surveys (CSUR)*, 33(1):31–88, 2001.

APPENDIX 6A

PK2 RECONSTRUCTED PHYLOGENIC TREE WITH SELECTED NODES FOR TRAINING THE GENERATIVE MODEL.

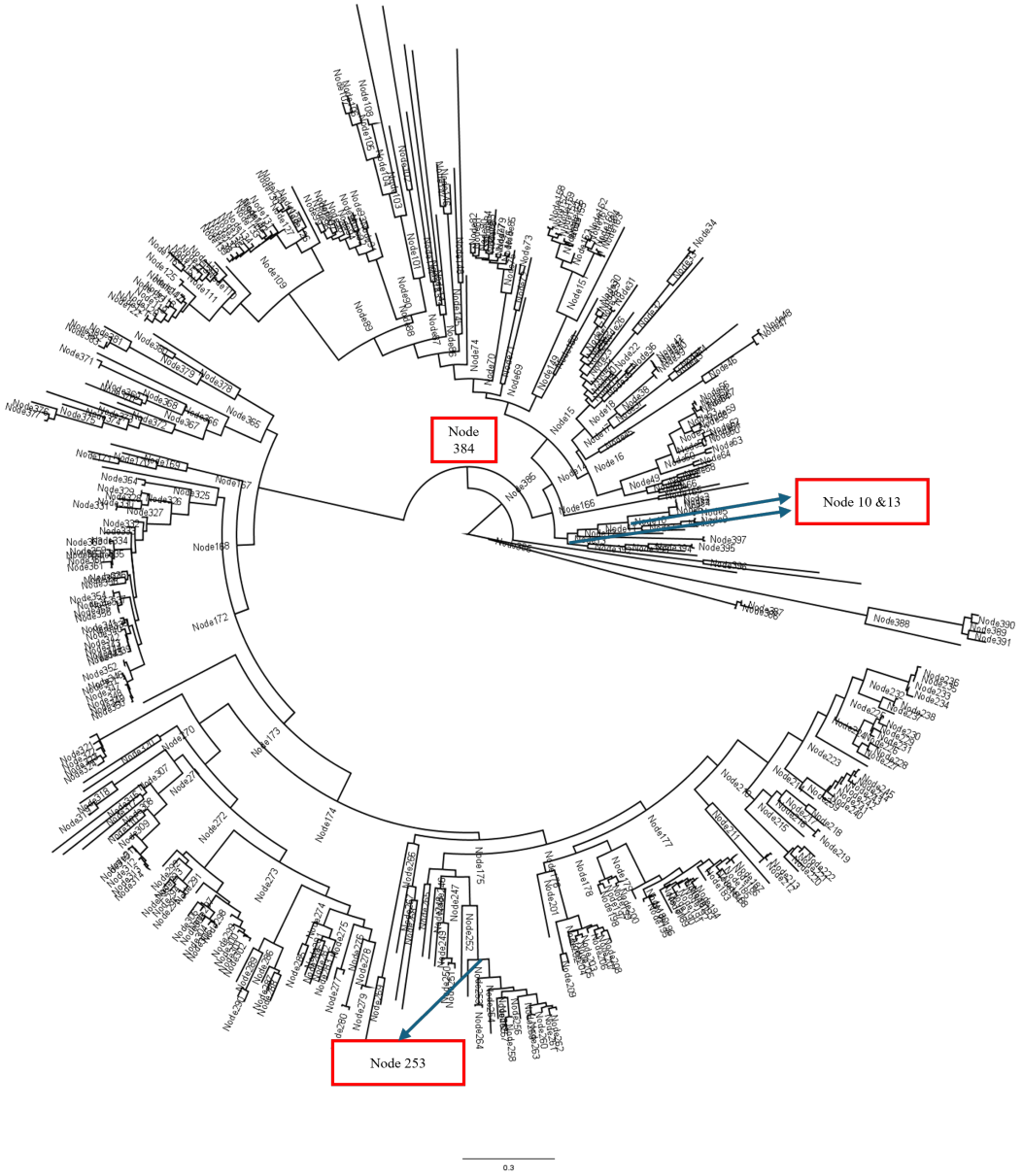


Figure 6A.1 The tree is generated via AP-LASR and visualized with Figtree software. Highlighted nodes are Node 10, Node 13, Node 253, and Node 384 which possessed high stability and were used as starting points to produce ancestral sequences for training data for the generative model.

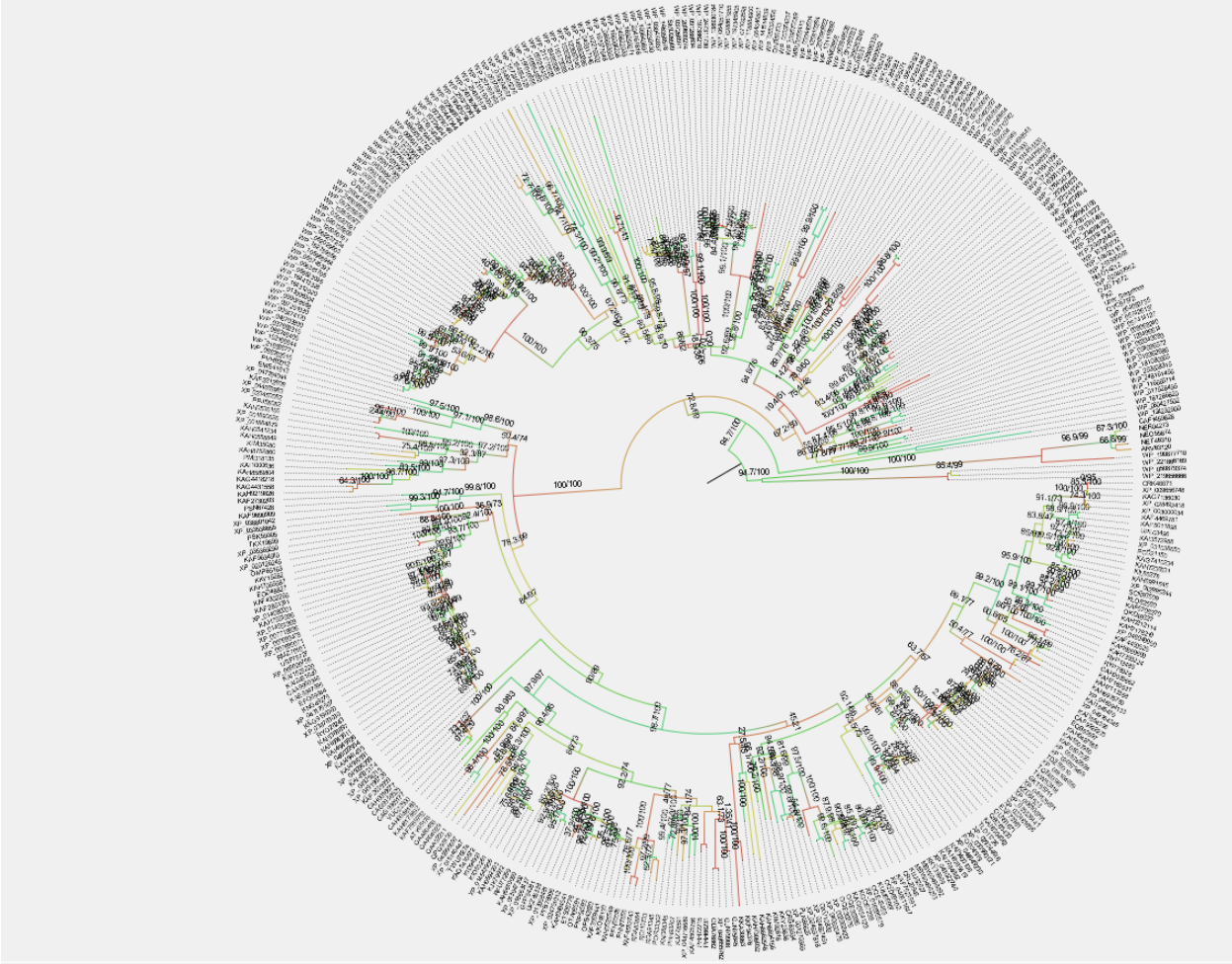


Figure 6A.2 Reconstructed Tree Quality Metrics for Branches.

APPENDIX 6B

STATISTICAL ANALYSIS FOR STABILITY OF GENERATED PROTEINS

Table 6B.1 Comparison of Model Performance Across Different Training and Generated Data Types.

	Modern-Generated	Ancestral-Type1-Generated	Ancestral-Type2-Generated	Modern-Training	Ancestral-Type1-Training	Ancestral-Type2-Training
Modern-Generated	1	5.40E-10	1.39E-08	1.21E-08	2.18E-07	6.66E-06
Ancestral-Type1-Generated	5.40E-10	1	0.6972	2.31E-31	0.2143	0.0608
Ancestral-Type2-Generated	1.39E-08	0.6972	1	3.61E-28	0.4168	0.1519
Modern-Training	1.21E-08	2.31E-31	3.61E-28	1	1.65E-27	2.59E-24
Ancestral-Type1-Training	2.18E-07	0.2143	0.4168	1.65E-27	1	0.5069
Ancestral-Type2-Training	6.66E-06	0.0608	0.1519	2.59E-24	0.5069	1

APPENDIX 6C

GENERATIVE MODEL ARCHITECTURE

```
ProteinVAE(
  (encoder): Encoder(
    (conv1): Conv1d(21, 64, kernel_size=(3,), stride=(1,),\ padding=(1,))
    (bn1): BatchNorm1d(64, eps=1e-05, momentum=0.1,\ affine=True)
    (flatten): Flatten(start_dim=1, end_dim=-1)
    (fc1): Linear(in_features=25856, out_features=10000, bias=True)
    (fc2): Linear(in_features=10000, out_features=5000, bias=True)
    (fc3): Linear(in_features=5000, out_features=2000, bias=True)
    (fc4): Linear(in_features=2000, out_features=500, bias=True)
    (fc5): Linear(in_features=500, out_features=100, bias=True)
    (z_mean): Linear(in_features=100, out_features=100, bias=True)
    (z_log_var): Linear(in_features=100, out_features=100, bias=True)
    (sampling): Sampling()
  )
  (decoder): Decoder(
    (fc1): Linear(in_features=100, out_features=500, bias=True)
    (fc2): Linear(in_features=500, out_features=2000, bias=True)
    (fc3): Linear(in_features=2000, out_features=5000, bias=True)
    (fc4): Linear(in_features=5000, out_features=10000, bias=True)
    (fc5): Linear(in_features=10000, out_features=25856, bias=True)
    (deconv1): ConvTranspose1d(64, 21, kernel_size=(3,),\ stride=(1,)
    , padding=(1,))
    (bn1): BatchNorm1d(21, eps=1e-05, momentum=0.1,\ affine=True)
    (dropout): Dropout(p=0.2, inplace=False)
  ))
```

APPENDIX 6D

STATISTICAL ANALYSIS FOR PROTEIN REPRESENTATION PERFORMANCES–LYSOZYME C

Table 6D.1 Results of Dunn Test in Case of Statistical Significance after Kruskal-Wallis for Different Metrics–Lysozyme. Only KNN results had statistical significance and needed post-hoc reports.

Group 1	Group 2	p-value	Classifier	Metric	Mean 1	Mean 2
InterPro	ASR-Max	0.000165	KNN	Precision	0.73	0.89
InterPro	ASR-Dist	0.003335	KNN	ROC AUC	0.81	0.86
ASR-Max	ASR-Dist	0.003965	KNN	ROC AUC	0.81	0.86
InterPro	ASR-Dist	0.021234	KNN	Precision	0.73	0.83
ASR-Max	ASR-Dist	0.105603	KNN	Precision	0.89	0.83
InterPro	ASR-Max	0.903116	KNN	ROC AUC	0.81	0.81

APPENDIX 6E

STATISTICAL ANALYSIS FOR PROTEIN REPRESENTATION PERFORMANCES–ENDOLYSIN

Table 6E.1 Results of Dunn Test in Case of Statistical Significance after Kruskal-Wallis for Different Metrics–Endolysin. Only KNN results had statistical significance.

Group 1	Group 2	p-value	Classifier	Metric	Mean 1	Mean 2
InterPro	ASR-Dist	0.000686	KNN	ROC AUC	0.90	0.93
InterPro	ASR-Dist	0.005173	KNN	Recall	0.55	0.65
InterPro	ASR-Dist	0.005527	KNN	Balanced Accuracy	0.76	0.81
ASR-Max	ASR-Dist	0.010570	KNN	ROC AUC	0.91	0.93
InterPro	ASR-Dist	0.020592	KNN	F1 Score	0.65	0.72
InterPro	ASR-Max	0.033665	KNN	Recall	0.55	0.62
InterPro	ASR-Max	0.067481	KNN	Balanced Accuracy	0.76	0.80
ASR-Max	ASR-Dist	0.155124	KNN	F1 Score	0.68	0.72
InterPro	ASR-Max	0.217757	KNN	F1 Score	0.65	0.68
ASR-Max	ASR-Dist	0.284875	KNN	Balanced Accuracy	0.80	0.81
ASR-Max	ASR-Dist	0.388140	KNN	Recall	0.62	0.65
InterPro	ASR-Max	0.524948	KNN	ROC AUC	0.90	0.91

CHAPTER 7

CONCLUSION

7.1 Overview

This thesis has presented a multifaceted approach to protein engineering, leveraging the power of machine learning (ML) and computational tools. The research addressed the complex challenge of designing proteins with desired functions through several innovative strategies.

Firstly, a robust methodology was developed for mapping next-generation sequencing data to experimental annotations, enhancing signal-to-noise ratios, and optimizing directed evolution starting points. The evaluation of various protein representation methods, including one-hot and physicochemical encodings, as well as language-based representations like ESM and UniRep, led to the creation of ensemble techniques that significantly improved ML performance. For proteins under 120 amino acids, this approach achieved a remarkable 94

The study then progressed to modeling protein-drug interactions, focusing on the therapeutically important organic anion-transporting polypeptides (OATPs). A comprehensive pipeline was established, integrating AlphaFold for structure prediction, molecular docking for interaction modeling, and a Heterogeneous Graph Neural Network (HeteroGNN) to capture both inter- and intra-molecular interactions. This approach revealed inconsistencies in reported OATP-drug interactions, particularly for OATP1B1, and identified key drug attributes such as topological surface area, heteroatom count, and hydrophobicity as critical factors. The research demonstrated that drug-based representations alone are insufficient for interpreting OATP-drug interactions, emphasizing the necessity of structure-based methods.

Furthermore, generative modeling to create novel proteins with desired properties was explored. By incorporating evolutionary insights through ancestral sequence reconstruction (ASR), both generative and language models were enhanced. A Variational Autoencoder (VAE) trained on ASR-derived sequences produced novel proteins that maintained the stability profiles of ancestral sequences, as validated using AlphaFold2 and FoldX. The integration of evolutionary-driven protein representations also improved downstream prediction tasks, particularly in protein stability

classification. Fine-tuning the ESM2 language model with ASR data created more effective protein embeddings, enhancing classification accuracy for multiple protein families.

In the following section, broader perspectives on the field of protein engineering and machine learning integration will be presented, along with suggestions for future improvements and research directions. These insights aim to guide the next generation of researchers in furthering the advancements made in this thesis and addressing remaining challenges in the field.

7.2 Perspective

Significant advancements have propelled the field of protein engineering forward in remarkable ways. AlphaFold3 has achieved unprecedented accuracy in predicting protein structures and interactions, while language models like ESM3 [1] have enhanced our understanding of protein sequences and functions. Diffusion models such as Chroma [2] have enabled the generation of novel protein-protein interaction domains with greater control, advancing programmable protein design. Recent studies have even demonstrated the generation of new CRISPR-Cas proteins that are more compatible with human cells [3], opening new avenues for gene editing technology. These developments highlight the transformative potential of integrating AI with biological research.

Despite these advancements, several critical challenges persist. The lack of standardized benchmarks and comprehensive metrics, given the vast and diverse nature of biological space, makes it difficult to establish universally applicable standards for evaluating ML models in protein engineering. Ensuring reliable and reproducible outcomes requires addressing data leakage and inconsistencies in ML implementations. Current methods often fail to fully account for the interconnected nature of proteins and their interaction profiles.

The gap between computational discoveries and experimental validation in protein engineering presents a significant bottleneck in the field. Unlike image generation, where results can be instantly assessed visually, protein engineering lacks immediate feedback mechanisms, significantly slowing down the iterative process of design, testing, and refinement. In Chapter 6, protein embeddings were used to analyze the semantic diversity of AI-generated proteins, and software such as AlphaFold was employed to assess their folding profiles, followed by FoldX for stability analysis. While these

approaches are beneficial for analyzing success, more generalizable and rapid methods are needed to improve the assessment of AI-generated proteins in terms of the likelihood of success for the intended properties. Ideally, these methods would provide a likelihood panel for both success and failure across different protein attributes, recognizing that protein engineering often involves balancing multiple properties simultaneously. This comprehensive view of potential outcomes would enhance both the speed and reliability of the protein engineering process.

Another significant challenge is the low signal-to-noise ratio in experimental data. NGS techniques and docking studies often produce noisy data. To address this issue, several strategies were implemented throughout this research. In Chapter 2, the PEAR software was utilized to reduce noise in obtained sequence data. Chapter 5 focused on mitigating structural noise for OATPs by analyzing the distribution of docking scores and selecting thresholds to maximize signal. Furthermore, collaborative work with the Harada lab at MSU-IQ investigated the impact of experimental conditions, such as incubation time, on data noise levels. These efforts demonstrate the potential for improving data quality through careful preprocessing and analysis of experimental parameters. Despite this progress, there remains a need for more advanced methods to identify and quantify various sources of noise, as well as to develop enhanced post-processing techniques for maximizing signal quality in protein engineering data.

Shifting emphasis from developing complex models to critically evaluating and enhancing data quality is crucial for ensuring model effectiveness and reliability. For example, more than 99% of the data in UniProt, a widely used public protein database, remains unannotated, and the quality of proteins is not determined [4]. This thesis addressed this issue by incorporating high-quality ancestral data into the generative model platform, leveraging evolutionary information to improve the data used. Despite this advancement, there is room for further improvement in making generative models conditional on co-optimizing attributes such as stability and fitness.

In conclusion, this research has shown that integrating machine learning with high-quality data and advanced computational tools can significantly enhance protein engineering efforts. The innovative approaches developed throughout this work offer a solid foundation for future

advancements, promising more efficient and accurate design of proteins with desired properties and functionalities. This progress paves the way for continued breakthroughs in the field, bringing us closer to overcoming existing challenges and unlocking new possibilities in protein engineering.

BIBLIOGRAPHY

[1] Tomas Hayes, Roshan Rao, Halil Akin, Nicholas J Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q Tran, Jonathan Deaton, Marius Wiggert, et al. Simulating 500 million years of evolution with a language model. *bioRxiv*, pages 2024–07, 2024.

[2] John B Ingraham, Max Baranov, Zak Costello, Karl W Barber, Wujie Wang, Ahmed Ismail, Vincent Frappier, Dana M Lord, Christopher Ng-Thow-Hing, Erik R Van Vlack, et al. Illuminating protein space with a programmable generative model. *Nature*, 623(7989):1070–1078, 2023.

[3] Jeffrey A. Ruffolo, Stephen Nayfach, Joseph Gallagher, Aadyot Bhatnagar, Joel Beazer, Riffat Hussain, Jordan Russ, Jennifer Yip, Emily Hill, Martin Pacesa, Alexander J. Meeske, Peter Cameron, and Ali Madani. Design of highly functional genome editors by modeling the universe of crispr-cas sequences. *bioRxiv*, 2024.

[4] Baohui Lin, Xiaoling Luo, Yumeng Liu, and Xiaopeng Jin. A comprehensive review and comparison of existing computational methods for protein function prediction. *Briefings in Bioinformatics*, 25(4):bbae289, 06 2024.