PREDICTION OF ENZYME KINETIC PARAMETERS AND SUBSTRATE SCOPES USING ARTIFICIAL INTELLIGENCE

Inaugural-Dissertation

zur Erlangung des Doktorgrades
der Mathematisch-Naturwissenschaftlichen Fakultät
der Heinrich-Heine-Universität Düsseldorf

vorgelegt von

Alexander Kroll
aus Ratingen, Deutschland

Düsseldorf, December 2022

aus dem Institut für Informatik
der Heinrich-Heine-Universität Düsseldorf

Gedruckt mit der Genehmigung der
Mathematisch-Naturwissenschaftlichen Fakultät der
Heinrich-Heine-Universität Düsseldorf

Berichterstatter:

1. Prof. Dr. Martin J. Lercher

2. Prof. Dr. Markus Kollmann

Tag der mündlichen Prüfung: 24. August 2023

ERKLÄRUNG

Ich versichere an Eides Statt, dass die Dissertation von mir selbständig und ohne unzulässige fremde Hilfe unter Beachtung der „Grundsätze zur Sicherung guter wissenschaftlicher Praxis an der Heinrich-Heine Universität Düsseldorf“ erstellt worden ist.

Düsseldorf, December 2022

Alexander Kroll

ACKNOWLEDGEMENTS

I am deeply grateful to my supervisor, Prof. Martin Lercher, for all of his guidance, advice, and support. He gave me the opportunity to improve my skills as a scientist and provided me with the freedom and encouragement to explore topics and ideas that piqued my interest, and for that I am extremely thankful.

I would also like to thank my colleagues in the CCB group for creating a friendly and constructive atmosphere. In particular, I would like to thank Deniz Sezer for his insightful discussions and his unwavering willingness to help and guide me whenever I needed it.

I am also grateful to Leandra Zanger for the many enjoyable hours we spent together on coffee breaks, for her incredible proofreading skills, for her listening ear and support during times when I felt stuck.

I want to thank my wife, Julia, for her endless support. I am grateful to her for celebrating my successes with me and for motivating and cheering me up when I was feeling frustrated. I deeply appreciate her understanding when I needed to put in extra hours of work. Thank you for never getting annoyed or bored when I couldn’t stop talking about my research, and thank you for patiently listening to all of my thoughts and problems.

Lastly, I would like to thank my mother, Martina. I cannot adequately express in words how grateful I am for her unconditional support. I am sure that I would have never been able to pursue a PhD without all of her help and all of the sacrifices that she has made for us. She is truly an inspiration, and I am forever grateful to her.

CONTENTS

1 SUMMARY

Cells are the building blocks for all living organisms on earth. In each cell, a complex network of biochemical reactions facilitates cellular metabolism, which is crucial for many biological functions. Metabolic network models are powerful tools that allow the simulation of cellular metabolism and can thus provide fundamental mechanistic insights. For example, metabolic network models can be used to predict environment-dependent growth rates, various phenotypic states under different cultural conditions, and the flow of metabolites through the metabolic reaction network [1]. Predicting more detailed quantities like optimal metabolite and enzyme concentrations or substrate-level regulatory mechanisms requires to incorporate information about enzyme kinetic parameters [2]. Unfortunately, even for model organisms, experimentally measured kinetic parameters are not available for the vast majority of enzymatic reactions [3, 4]. Prediction methods for kinetic parameters could help to overcome this issue, but previously developed methods can either be only applied to a small subset of enzymatic reactions [5, 6], lead to unrealistic values that are largely uncoupled from the true kinetic parameters [7–9], or they provide inaccurate predictions for enzymes that are not highly similar to proteins with measured kinetic parameters [10]. However, not only missing kinetic parameters but also missing functional information for enzyme-encoding genes lead to knowledge gaps in metabolic networks. Even in model organisms, large fractions of genes do not have high-quality functional annotations [11]. One the one hand, this can lead to important reactions missing in metabolic networks, and on the other hand, many reactions cannot be associated with the catalyzing enzyme.

In this thesis, I aim to overcome the issues of missing enzyme kinetic parameters and of not yet annotated enzymes with the use of machine and deep learning models. I developed the first general prediction model for the Michaelis-Menten constant K_M. The resulting model is applicable to any natural enzyme-substrate pair and achieves a coefficient of determination $R^2 = 0.53$ on a test set. Moreover, I developed a general model for predicting enzyme turnover numbers k_{cat} for natural reactions of wild-type enzymes. The model outperforms previously developed prediction models and achieves a coefficient of determination $R^2 = 0.40$ on a test set. To predict the function of not yet fully annotated enzymes, I developed a general model for predicting the substrate scopes of enzymes. The resulting model generalizes well even to enzymes that are not highly similar to enzymes in the training set, and it achieves an accuracy of over 91% on a test set.

To develop these general prediction models, it was necessary to create maximally informative numerical representations of the proteins and the small molecules relevant for the downstream prediction tasks. This was achieved by using and modifying state-of-the-art deep learning methods for converting the linear protein amino acid sequences and the structures of the small molecules into numerical vectors. For all of the developed prediction models, we only used input information that is easily accessible, which makes the prediction models broadly applicable.

2 INTRODUCTION

2.1 USING MATHEMATICAL MODELS TO GAIN INSIGHTS INTO CELLULAR METABOLISM

Cells are the basic structural units for all living organisms on earth. Complex organisms such as *Homo sapiens* consist of ~37 trillion eukaryotic cells, whereas the simplest and smallest living organisms, bacteria and archaea, consist of only a single prokaryotic cell. Even those unicellular microorganisms, which typically have a size of only approximately 1µm, are highly complex. For example, tiny prokaryotic cells can contain millions of copies of a few thousand different complex molecular machineries, the proteins [12, 13].

The fastest growing bacteria can replicate themselves in under 10 minutes [14]. To achieve such high growth rates, the bacterial organisms must produce all components of a cell, including the cell membrane, the genome, the ribosomes, and the proteins, with an incredible speed. All of these complex components are often synthesized by utilizing a single carbon source, such as small glucose molecules, which are imported into the cell by transport proteins. Starting from small molecules as the only available resource, how do cells manage to synthesize these enormous amounts of various large molecules within such a short time span? The metabolism of cells consists of huge numbers of interconnected pathways that together form large metabolic networks. Enzymes, which are proteins that catalyze chemical reactions, play a crucial role in these networks. In metabolic pathways, enzymes catalyze a sequence of chemical reactions, in which the product of one reaction serves as the substrate for the following reaction. Thereby enzymes fulfill two important functions. On the one hand, they accelerate reaction rates up to a million-fold [15], while on the other hand, they favour desired reactions, such that mainly metabolites are produced that are relevant for the cellular metabolism. Thus, only little energy is wasted by molecules that become reactants in irrelevant or even harmful reactions [16–19].

To achieve a fundamental understanding of life, it is necessary to gain detailed insights into the basic processes of living organisms, the metabolism of cells. Moreover, deepening the understanding of cellular metabolism can have an impact on many different research areas. For example, it can help treating diseases by identifying potential drug targets and by identifying potential side effects of drugs [20–23], or it can aid the production of molecules, e.g., substances that are required for the production of food, biofuels, and pharmaceuticals [24–26].

In recent years, many different mathematical models that aimed to simulate the highly complex metabolism of cells have been developed. However, obtaining mechanistic insights such as understanding environment-dependent growth rates and proteome allocations of cells remains a major problem in biology. In the following subsection, I will give a short overview about existing approaches for modelling cellular metabolism, and I am going to discuss their limitations.

Metabolic network models

Numerous mathematical frameworks have been developed for modelling and simulating cellular metabolism. These frameworks allow to mathematically represent large metabolic networks and to simulate biochemical activities as well as the growth of single cells [1]. To construct a metabolic network model for a specific cell, a network is created that is composed of all enzyme-catalyzed reactions that are known to be happening inside the cell. Additionally, information about all enzyme-encoding genes can be added to the model [2]. To gain insights about cells, for example about their various phenotypic states under different culturing conditions, an objective function can be optimized for the metabolic networks. Most commonly, the objective function is a biomass function, which is composed of all essential metabolites in the proportions needed for growth [1]. Optimizing the biomass function is often interpreted as maximizing growth rate [27], although it is yield that is really optimized [28].

Modelling frameworks that solely rely on information about the metabolic reaction network are powerful tools and have been successfully applied to a broad range of applications such as understanding microorganisms [29–31], predicting drug targets [32], understanding the effect of gene deletions [33], and for metabolic engineering [34–36]. However, to fully understand the physiology of cells, one needs to consider the non-linear dependence of biochemical reactions on the concentrations of the enzymes and reactants involved. To achieve this, enzyme kinetic parameters need to be incorporated into the model. The turnover number k_{cat} of an enzyme-catalyzed reaction is defined as the maximal rate of one active site of an enzyme for converting substrate molecules into product molecules per time unit. The Michaelis constant K_M is defined as the concentration of a substrate at which an enzyme operates at half of its maximal catalytic rate; it hence describes the affinity of an enzyme for a specific substrate, where lower K_M values indicate higher affinity.

Reactions that are catalyzed by enzymes with lower K_M values require lower metabolite concentrations to produce the same amount of products compared to reactions catalyzed by enzymes with higher

K_M values. Similarly, reactions catalyzed by enzymes with higher turnover numbers k_{cat} require lower enzyme concentrations to produce the same amount of products compared to enzymes with lower k_{cat}. Thus, to predict quantities such as optimal metabolite and enzyme concentrations of a cell, information about enzyme kinetic parameters is required.

Michaelis-Menten kinetics is one of the best-known models for enzyme kinetics. If we assume an irreversible reaction $S \to P$, where a single substrate S is converted into a product P by an enzyme E, the Michaelis-Menten rate law defines the reaction rate ν as

$$\nu = \frac{k_{cat}[E][S]}{K_M + [S]},$$

where $[E]$ and $[S]$ are the concentrations of the enzyme and the substrate [37]. This rate law can be generalized to reversible reactions and to reactions with multiple substrates and products. Hence, when using Michaelis-Menten kinetics to describe enzyme kinetics, all reaction rates can be calculated from the parameters K_M and k_{cat} and from the concentrations of the reactants and of the enzyme.

Many models and modelling approaches that incorporate kinetic parameters were developed in the past [8, 38–42]. Making accurate predictions with these kinetic models requires realistic measurements or estimates for either all k_{cat} values or for all K_M and k_{cat} values. Unfortunately, experimental measurements for enzyme kinetic parameters are difficult and time-consuming, and currently, the use of kinetic models is limited due to a lack of available K_M and k_{cat} values for most enzymatic reactions. Even for the biochemically best-characterized organism, *Escherichia coli*, *in vitro* measurements for K_M are available for less than 30% of all natural enzyme-substrate pairs [3], and *in vitro* k_{cat} values are available for only ~10% of approximately 2 000 enzymatic reactions [4]. As it is unrealistic to obtain all kinetic parameters for a large metabolic network experimentally, various efforts have been made to estimate or predict K_M and k_{cat}. However, these methods can either only provide parameters for a subset of reactions [5], lead to parameters that are not highly connected to the true kinetic parameters [8, 43, 44], or do not generalize well to enzymes for which no experimental measurements are available [10] (see below, "Predicting and estimating enzyme kinetic parameters"). Because of these limitations, it was previously not possible to fully parameterize any genome-scale metabolic reaction network with realistic kinetic parameters. Hence, prediction models that generalize well to unseen enzymes and provide K_M and k_{cat} values that are highly correlated with experimental measurements would be a major step towards parameterizing kinetic metabolic network models.

Another limitation of metabolic network models is missing information about enzymatic function for many gene products. For example, the *Escherichia coli* genome contains 4 623 annotated unique genes, approximately 1 600 of which have unknown functions [11]. This can lead to important and relevant reactions missing in metabolic networks. Conversely, even in well curated metabolic models, many reactions have not been associated with an enzyme yet. It is difficult to obtain suitable K_M and k_{cat} values for these reactions because kinetic constants are highly dependent on the catalyzing enzyme.

2.2 PREDICTING AND ESTIMATING ENZYME KINETIC PARAMETERS

In this subsection, I will give a short overview over existing approaches for predicting the enzyme kinetic parameters K_M and k_{cat}, and I will outline their limitations.

Fitting K_M and k_{cat} values using optimization processes

To obtain estimates of enzyme kinetic parameters, K_M and k_{cat} can be fitted during optimization processes [7–9]. For example, kinetic parameters can be obtained by choosing values that lead to a minimization of the differences between fluxes predicted by a metabolic network model and experimentally measured steady-state fluxes. These procedures typically lead to K_M and k_{cat} predictions with wide confidence ranges that are often not highly connected to experimental measurements [8, 43, 44].

Predicting K_M and k_{cat} using machine learning methods

An alternative approach for obtaining kinetic parameters without executing labor-intensive experiments is to predict these parameters using machine learning models, but only few previous such studies exist. Heckmann *et al.* [5] successfully predicted k_{cat} values for a small subset of reactions in *Escherichia coli* using machine learning. The predicted k_{cat} values improve the proteome allocation predictions of two kinetic metabolic models. However, their approach is limited to a single organism, and calculating the model's input requires knowledge about the enzymes' active sites, which is unavailable for the majority of enzymes. Because of these limitations, Heckmann *et al.*'s training data consist of only ~200 data points, which is rather small for such a complex prediction task. Recently, Li *et al.* [10] developed a deep learning model, DLKcat, that uses information about the enzyme's amino acid sequence and about one of the substrates of the reaction to predict k_{cat}. DLKcat is a general approach, which can in principle be applied to any enzymatic reaction. However, DLKcat does not use any information about the products or about more than one substrate of a reaction, and the model does not generalize well to enzymes that are not highly similar to enzymes in the training set [45].

For the prediction of Michaelis constants K_M, previously only small and substrate-specific models existed. Borger *et al.* [6] used linear regression models to predict K_M values of enzyme-substrate pairs for eight different substrates. The models were trained with knowledge on K_M measurements for the same substrate paired with different enzymes in the same organism and paired with the same enzyme in other organisms. A second approach for predicting K_M, which is also substrate-specific, was later developed by Yan *et al.* [46], who trained a neural network to predict beta-glucosidases for the substrate cellobiose. Both previous approaches require the training of a completely new model for every substrate with the requirement of experimentally measured K_M values being available for the substrate of interest. Hence, the described approaches are unsuitable for making predictions for not very well-studied reactions and enzyme-substrate pairs.

2.3 PREDICTING ENZYME FUNCTIONS

Predicting functions of not yet annotated proteins using artificial intelligence can help to overcome and close knowledge gaps in metabolic networks. Deep learning methods have previously been successfully applied to predict enzymes' EC classes [47–49], to predict functional domains within amino acid sequences [50], and to predict suitable substrates for whole EC classes [51]. These approaches help to gain insights into the function of uncharacterized enzymes. However, the exact catalyzed reaction can still remain unknown as the substrate scope for enzymes within the same EC class or for enzymes with the same functional domain can be highly diverse [52].

Models that are capable of predicting enzyme-substrate pairs are needed to predict enzymatic functions more precisely. Multiple such prediction models have been developed, but all of them can only be applied to small groups of enzymes [53–57]. These models were typically trained on dense and comprehensive training data sets containing positive and negative samples for a high fraction of all possible enzyme-substrate pairs. Transferring these approaches to new enzyme families requires the training of completely new models with many available training data points for the family of interest.

A general prediction model that is applicable to a broad range of enzymes without requiring many training samples for every enzyme family would be a huge step towards predicting the function of not yet characterized enzyme-encoding genes. Such a model would not only be useful to fill gaps in metabolic networks but also for pharmaceutical research and for bio-engineering metabolic pathways [24–26].

2.4 NUMERICAL PROTEIN AND METABOLITE REPRESENTATIONS

As described in the previous sections, general machine learning models that facilitate the prediction of enzyme kinetic parameters and of substrate candidates for enzymes would be a major advance towards closing knowledge gaps of metabolic reaction networks. Ideally, such general prediction models could be applied to a large and diverse set of many different enzymes. Machine learning models are mathematical functions that receive numerical features as their input. Hence, to achieve general prediction models, it is necessary to create meaningful numerical input representations with information relevant to the prediction tasks.

Numerical protein representations

Many approaches for creating numerical protein representations exist. Most are based on deep learning and can be broadly divided into two categories: models extracting information from the linear protein amino acid sequence [58–60] and models extracting information from the 3D protein structure [61–63]. As it is laborious and time-consuming to determine the 3D structure experimentally, the structure was previously unknown for the vast majority of proteins. However, due to the recent development of protein structure prediction tools [64, 65], 3D-based protein models could soon become very useful. Currently, representations based on the linear amino acid sequences are still most commonly used to numerically encode information about proteins.

The most successful sequence-based methods apply algorithms that were originally developed for natural language processing (NLP) tasks like translation or text classification [59, 60]. To apply NLP models to protein sequences, every amino acid in a sequence is interpreted as a word in a text or sentence. Such NLP models are often trained in a self-supervised way, e.g., they are trained to predict the next amino acid in a sequence [59] or to predict the type of an amino acid that has been masked in the input sequence [60]. Even though these models do not use any information about the structure of the folded proteins, it has been shown that sequence-based protein representations contain meaningful information about the protein structure and function. This was demonstrated by using these protein representations to predict amino acid contact maps and enzyme properties like EC classes or functional domains [59, 60, 62].

Numerical metabolite representations

As it is the case for protein representations, there are many different ways to numerically represent small molecules. One class of these approaches are expert-crafted representations, i.e., the functions and algorithms to create molecule representations are chosen by experts and are not learned by machine learning algorithms. These so-called fingerprints are often high-dimensional binary vectors containing information about the structure and the substructures of molecules [66–68].

An alternative to expert-crafted molecule vectors are machine learning-generated fingerprints. One approach to generate these fingerprints is through graph neural networks (GNNs) [69, 70], which are neural networks that were developed to process graph representations. Small molecules can be represented as graphs, by interpreting every atom of the molecule as a node of the graph and every bond as an edge. With these representations, molecules can be used as the input for a GNN, which is trained to predict a property of the inserted molecules. While processing a graph, the GNN converts the graph representation into a single numeric vector of fixed length, which is then used to predict the property of interest. These representations ideally contain all information about the molecules that is relevant for the prediction task. After model training, the GNN can be used to calculate task-specific representations for all molecules, which can serve as the input for other prediction models.

Numerical reaction representations

Making predictions for enzymatic turnover numbers k_{cat} with machine learning models requires the numerical representation of chemical reactions. To achieve this, binary expert-crafted molecular fingerprints [66–68] for all substrates and for all products can be combined into a single reaction fingerprint. Most machine learning models require input vectors of fixed length. Hence, even for varying numbers of substrates and products, the resulting reaction fingerprints should not vary in length. This can be achieved by first combining all substrate fingerprints into a single vector as well as combining all product fingerprints into a single vector, e.g., by element-wise summation of the fingerprints. The two resulting vectors for the substrates and the products can then be combined into a single reaction vector by concatenation or by subtracting the product vector from the substrate vector [67, 71].

2.5 DATABASES FOR MODEL TRAINING AND VALIDATION

Machine learning models that aim to solve complex problems such as predicting enzyme properties require large sets of training data to generalize well and to produce meaningful results. The largest databases containing information about enzyme kinetic parameters and enzyme functions are BRENDA, Sabio-RK, UniProt, and the GO annotation database.

BRENDA (BRaunschweig ENzyme DAtabase) [72] is one of the biggest databases containing experimental enzyme data, including K_M and k_{cat}. Information in BRENDA is curated and extracted from published papers. Sabio-RK [73] is another curated database containing information about enzymatic reactions and their kinetic parameters. UniProt (universal protein database) [74] is the largest existing database for proteins and contains information about the sequence, structure, and function of proteins, but also information about kinetic parameters of enzymes can be extracted from UniProt. The GO (Gene Ontology) annotation database [75] contains annotations for proteins, including annotations about the catalytic function of enzymes, and can thus be used to create a database for natural substrates of enzymes.

Using information from these databases requires data preprocessing. For example, k_{cat} values in BRENDA and the UniProt databases are usually not assigned to the full chemical reactions. Instead, for most measurements only one of the substrates of a reaction is given. However, as it is desirable to use information about the whole chemical reaction to predict k_{cat}, enzyme and substrate information can be used to map these data points to the corresponding reactions. Moreover, not all data points in BRENDA are assigned to a unique protein identifier. To map those data points with missing protein information to the catalyzing enzyme, the organism name and EC number can be used.

2.6 AIMS AND RESULTS OF THIS THESIS

Numerous kinetic genome-scale metabolic network models exist that require knowledge of enzyme kinetic parameters. However, as described above, a complete set of realistic parameters K_M and k_{cat} is not available for any of these models. Additionally, many models have knowledge gaps in their metabolic networks because either the function of enzymatic genes are unknown and thus reactions are missing from the metabolic network, or because reactions could not be associated with the correct enzyme-encoding gene yet. In this thesis, I present three different prediction models that are capable of either predicting missing enzymatic parameters K_M or k_{cat}, or predicting substrate candidates for wild-type enzymes. All models were trained with large datasets containing thousands of experimentally validated data points that I extracted from the Sabio-RK, UniProt, BRENDA, and GO databases.

In *Manuscript 1*, we developed the first general prediction model for the Michaelis constant K_M of enzyme-substrate pairs. A gradient boosting model was trained with thousands of experimentally obtained K_M values for a diverse set of many different enzymes and substrates. The resulting model uses numerical enzyme representations and task-specific substrate fingerprints as its input and can be applied to any natural enzyme-substrate pair with known enzyme amino acid sequence and with known substrate structure. To evaluate model performance, we used the coefficient of determination R^2, which quantifies the proportion of variance in the target variable that can be explained by the prediction model. On an independent test set with previously unseen enzyme-substrate pairs, the trained model achieves a coefficient of determination $R^2 = 0.53$, which means that more than 50% of the variance in K_M values can be predicted.

In *Manuscript 2*, I implemented a general machine learning model, the Turnover Number Prediction model - TurNuP -, for the prediction of turnover numbers k_{cat} for natural reactions of wild-type enzymes. I trained a gradient boosting model that uses state-of-the-art enzyme representations and numerical representations of the whole chemical reaction as its input. In contrast to previous methods, this model can be successfully applied to enzymatic reactions from any organism and even to enzymes that are not highly similar to enzymes in the training set. TurNuP outperforms previous models for the same task and achieves a coefficient of determination $R^2 = 0.40$ on an independent test set.

In *Manuscript 3*, we developed a binary classification model, the Enzyme Substrate Prediction model – ESP –, that receives task-specific enzyme and metabolite representations as its input and predicts whether the metabolite is a substrate for the given enzyme. The model was trained with over 18 000 experimentally validated enzyme-substrate pairs and achieves a high accuracy of over 91% on an independent test set. ESP is the first general model for predicting enzymes' substrate scopes that is not only applicable to single enzyme families or only to a small group of enzymes. The prediction model can be used to identify candidate substrates for enzymes with yet unknown substrate scope.

The following chapter consists of the three manuscripts that I described above, each detailing one of the prediction models, its construction, training, and validation. In the last chapter, I will conclude this thesis with an outlook, in which I first discuss current limitations of the presented prediction models and possibilities to further improve model performances. Moreover, I will discuss which prediction task could be solved in the future to further improve the utility of metabolic reaction networks.

3 MANUSCRIPTS

3.1 MANUSCRIPT 1

Manuscript 1 is identical to the original version of the paper as it is published in the journal PLOS Biology [3].

Contributions to Manuscript 1

I designed the study together with Martin Lercher. Martin Lercher defined the overall task and suggested additional analyses for the trained model. David Heckmann suggested the use of extended-connectivity fingerprints (ECFPs) to numerically represent metabolites, and Martin Engqvist suggested to use UniRep vectors as enzyme representations. All other design choices were made by me, including the selection of training data and machine learning algorithms as well as the choice of a graph neural network to create task-specific substrate representations. Except for the calculation of the UniRep vectors, which was implemented by Martin Engqvist, I implemented all other software. This includes the implementation and training of all machine learning models, the creation and curation of the dataset, as well as all analyses and visualizations. I wrote the original draft of the manuscript. The manuscript was reviewed and edited by David Heckmann, Martin Lercher, Martin Engqvist, and me.

PLOS BIOLOGY

METHODS AND RESOURCES

Deep learning allows genome-scale prediction of Michaelis constants from structural features

Alexander Kroll[1], Martin K. M. Engqvist[2], David Heckmann[1]*, Martin J. Lercher[1]*

1 Institute for Computer Science and Department of Biology, Heinrich Heine University, Dësseldorf, Germany, **2** Department of Biology and Biological Engineering, Chalmers University of Technology, Gothenburg, Sweden

* david.heckmann@hhu.de (DH); martin.lercher@hhu.de (MJL)

Abstract

The Michaelis constant K_M describes the affinity of an enzyme for a specific substrate and is a central parameter in studies of enzyme kinetics and cellular physiology. As measurements of K_M are often difficult and time-consuming, experimental estimates exist for only a minority of enzyme±substrate combinations even in model organisms. Here, we build and train an organism-independent model that successfully predicts K_M values for natural enzyme±substrate combinations using machine and deep learning methods. Predictions are based on a task-specific molecular fingerprint of the substrate, generated using a graph neural network, and on a deep numerical representation of the enzyme's amino acid sequence. We provide genome-scale K_M predictions for 47 model organisms, which can be used to approximately relate metabolite concentrations to cellular physiology and to aid in the parameterization of kinetic models of cellular metabolism.

OPEN ACCESS

Citation: Kroll A, Engqvist MKM, Heckmann D, Lercher MJ (2021) Deep learning allows genome-scale prediction of Michaelis constants from structural features. PLoS Biol 19(10): e3001402. https://doi.org/10.1371/journal.pbio.3001402

Academic Editor: Jason W. Locasale, Duke University, UNITED STATES

Received: December 11, 2020

Accepted: August 26, 2021

Published: October 19, 2021

Peer Review History: PLOS recognizes the benefits of transparency in the peer review process; therefore, we enable the publication of all of the content of peer review and author responses alongside final, published articles. The editorial history of this article is available here: https://doi.org/10.1371/journal.pbio.3001402

Data Availability Statement: All datasets generated and the Python code used to produce the results (in Jupyter notebooks) are available from https://github.com/AlexanderKroll/KM_prediction.

Introduction

The Michaelis constant, K_M, is defined as the concentration of a substrate at which an enzyme operates at half of its maximal catalytic rate; it hence describes the affinity of an enzyme for a specific substrate. Knowledge of K_M values is crucial for a quantitative understanding of enzymatic and regulatory interactions between enzymes and metabolites: It relates the intracellular concentration of a metabolite to the rate of its consumption, linking the metabolome to cellular physiology.

As experimental measurements of K_M and k_{cat} are difficult and time-consuming, no experimental estimates exist for many enzymes even in model organisms. For example, in *Escherichia coli*, the biochemically best characterized organism, in vitro K_M measurements exist for less than 30% of natural substrates (see Methods, "Download and processing of K_M values"), and turnover numbers have been measured in vitro for only about 10% of the approximately 2,000 enzymatic reactions [1].

K_M values, together with enzyme turnover numbers, k_{cat}, are required for models of cellular metabolism that account for the concentrations of metabolites. The current standard approach in large-scale kinetic modeling is to estimate kinetic parameters in an optimization process

Funding: This work was funded through grants to M.J.L. by the Volkswagenstiftung (in the "Life?" program) and by the Deutsche Forschungsgemeinschaft (CRC 1310 and, under Germany's Excellence Strategy, EXC 2048/1, Project ID: 390686111). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing interests: The authors have declared that no competing interests exist.

Abbreviations: D-MPNN, directed message passing neural network; DTBA, drug–target binding affinity; EC, Enzyme Commission; ECFP, extended connectivity fingerprint; FCNN, fully connected neural network; GNN, graph neural network; LogP, octanol–water partition coefficient; MRPE, mean relative prediction error; MSE, mean squared error; MW, molecular weight; QSAR, quantitative structure–activity relationship; ReLU, rectified linear unit.

[2–4]. These optimizations typically attempt to estimate many more unknown parameters than they have measurements as inputs, and, hence, the resulting K_M and k_{cat} values have wide confidence ranges and show little connection to experimentally observed values [2]. Therefore, predictions of these values from artificial intelligence, even if only up to an order of magnitude, would represent a major step toward more realistic models of cellular metabolism and could drastically increase the biological understanding provided by such models.

Only few previous studies attempted to predict kinetic parameters of natural enzymatic reactions in silico. Heckmann and colleagues [5] successfully employed machine learning models to predict unknown turnover numbers for reactions in *E. coli*. They found that the most important predictors of k_{cat} were the reaction flux catalyzed by the enzyme, estimated computationally through parsimonious flux balance analysis, and structural features of the catalytic site. While many *E. coli* k_{cat} values could be predicted successfully with this model, active site information was not available for a sizeable fraction of enzymes [5]. Moreover, neither active site information nor reaction flux estimates are broadly available beyond a small number of model organisms, preventing the generalization of this approach.

Borger and colleagues [6] trained a linear model to predict K_M values based on other K_M measurements for the same substrate paired with different enzymes in the same organism and with the same enzymes in other organisms; they fitted an independent model for each of 8 different substrates. Yan and colleagues [7] later followed a similarly focused strategy, predicting K_M values of beta-glucosidases for the substrate cellobiose based on a neural network. These 2 previous prediction approaches for K_M targeted individual, well-studied enzyme–substrate combinations with ample experimental K_M data for training and testing. Their strategies are thus unsuitable for less well-studied reactions and cannot be applied to genome-scale predictions.

A related problem to the prediction of K_M is the prediction of drug–target interactions, an important task in drug development. Multiple approaches for the prediction of drug–target binding affinities (DTBAs) have been developed (reviewed in [8]). Most of these approaches are either similarity-based, structure-based, or feature-based. Similarity-based methods rely on the assumption that similar drugs tend to interact with similar targets; these methods use known drug–target interactions to learn a prediction function based on drug–drug and target–target similarity measures [9,10]. Structure-based models for DTBA prediction utilize information on the target protein's 3D structure [11,12]. Neither of these 2 strategies can easily be generalized to genome-scale, organism-independent predictions, as many enzymes and substrates share only distant similarities with well-characterized molecules, and 3D structures are only available for a minority of enzymes.

In contrast to these first 2 approaches, feature-based models for drug–target interaction predictions use numerical representations of the drug and the target as the input of fully connected neural networks (FCNNs) [13–16]. The drug feature vectors are most often either SMILES representations [17], expert-crafted fingerprints [18–20], or fingerprints created with graph neural networks (GNNs) [21,22], while those of the targets are usually sequence-based representations. As this information can easily be generated for most enzymes and substrates, we here use a similar approach to develop a model for K_M prediction.

An important distinction between the prediction of K_M and DTBA prediction is that the former aims to predict affinities for known, natural enzyme–metabolite combinations. These affinities evolved under natural selection for the enzymes' functions, an evolutionary process strongly constrained by the metabolite structure. In contrast, wild-type proteins did not evolve in the presence of a drug, and, hence, molecular structures are likely to contain only very limited information about the binding affinity for a target without information about the target protein.

Despite the central role of the metabolite molecular structure for the evolved binding affinity of its consuming enzymes, important information on the affinity must also be contained in the enzyme structure and sequence. To predict K_M, it would be desirable to employ detailed structural and physicochemical information on the enzyme's substrate binding site, as done by Heckmann and colleagues for their k_{cat} predictions in *E. coli* [5]. However, these sites have only been characterized for a minority of enzymes [23]. An alternative approach is to employ a multidimensional numerical representation of the entire amino acid sequence of the enzyme, as provided by UniRep [24]. UniRep vectors are based on a deep representation learning model and have been shown to retain structural, evolutionary, and biophysical information.

Here, we combine UniRep vectors of enzymes and diverse molecular fingerprints of their substrates to build a general, organism-, and reaction-independent model for the prediction of K_M values, using machine and deep learning models. In the final model, we employ a 1,900-dimensional UniRep vector for the enzyme together with a task-specific molecular fingerprint of the substrate as the input of a gradient boosting model. Our model reaches a coefficient of determination of $R^2 = 0.53$ between predicted and measured values on a test set, i.e., the model explains 53% of the variability in K_M values across different, previously unseen natural enzyme–substrate combinations. In **S1 Data**, we provide complete K_M predictions for 47 genome-scale metabolic models, including those for *Homo sapiens*, *Mus musculus*, *Saccharomyces cerevisiae*, and *E. coli*.

Results

For all wild-type enzymes in the BRENDA database [25], we extracted organism name, Enzyme Commission (EC) number, UniProt ID, and amino acid sequence, together with information on substrates and associated K_M values. If multiple K_M values existed for the same combination of substrate and enzyme amino acid sequence, we took the geometric mean. This resulted in a dataset with 11,675 complete entries, which was split into a training set (80%) and a test set only used for the final validation (20%). All K_M values were $\log_{10}$-transformed.

Predicting K_M from molecular fingerprints

To train a prediction model for K_M, we first had to choose a numerical representation of the substrate molecules. For each substrate in our dataset, we calculated 3 different expert-crafted molecular fingerprints, i.e., bit vectors where each bit represents a fragment of the molecule. The expert-crafted fingerprints used are extended connectivity fingerprints (ECFPs), RDKit fingerprints, and MACCS keys. We calculated them with the python package RDKit [19] based on MDL Molfiles of the substrates (downloaded from KEGG [26]; a Molfile lists a molecule's atom types, atom coordinates, and bond types [27]).

MACCS keys are 166-dimensional binary fingerprints, where each bit contains the information if a certain chemical structure is present in a molecule, e.g., if the molecule contains a ring of size 4 or if there are fewer than 3 oxygen atoms present in the molecule [20]. RDKit fingerprints are generated by identifying all subgraphs in a molecule that do not exceed a particular predefined range. These subgraphs are converted into numerical values using hash functions, which are then used to indicate which bits in a 2,048-dimensional binary vector are set to 1 [19]. Finally, to calculate ECFPs, molecules are represented as graphs by interpreting the atoms as nodes and the chemical bonds as edges. Bond types and feature vectors with information about every atom are calculated (types, masses, valences, atomic numbers, atom charges, and number of attached hydrogen atoms) [18]. Afterwards, these identifiers are updated for a predefined number of steps by iteratively applying predefined functions to summarize aspects of neighboring atoms and bonds. After the iteration process, all identifiers are used as the input of a hash function to produce a binary vector with structural information about the molecule. The number of iterations and the dimension of the fingerprint can be chosen freely. We set them to the default values of 3 and 1,024, respectively; lower or higher dimensions led to inferior predictions.

To compare the information on K_M contained in the different molecular fingerprints independent of protein information, we used the molecular fingerprints as the sole input to elastic nets, FCNNs, and gradient boosting models. To the fingerprints, we added the 2 features molecular weight (*MW*) and octanol–water partition coefficient (*LogP*), which were shown to be correlated with the K_M value [28]. The models were then trained to predict the K_M values of enzyme–substrate combinations (**Fig** 1A). The FCNNs consisted of an input layer with the dimension of the fingerprint (including the additional features *MW* and *LogP*), 2 hidden layers, and a 1D output layer (for more details, see Methods). Gradient boosting is a machine learning technique that creates an ensemble of many decision trees to make predictions. Elastic nets are regularized linear regression models, where the regularization coefficient is a linear combination of the L_1– and L_2-norm of the model parameters. For each combination of the 3 model types and the 3 fingerprints, we performed a hyperparameter optimization with 5-fold

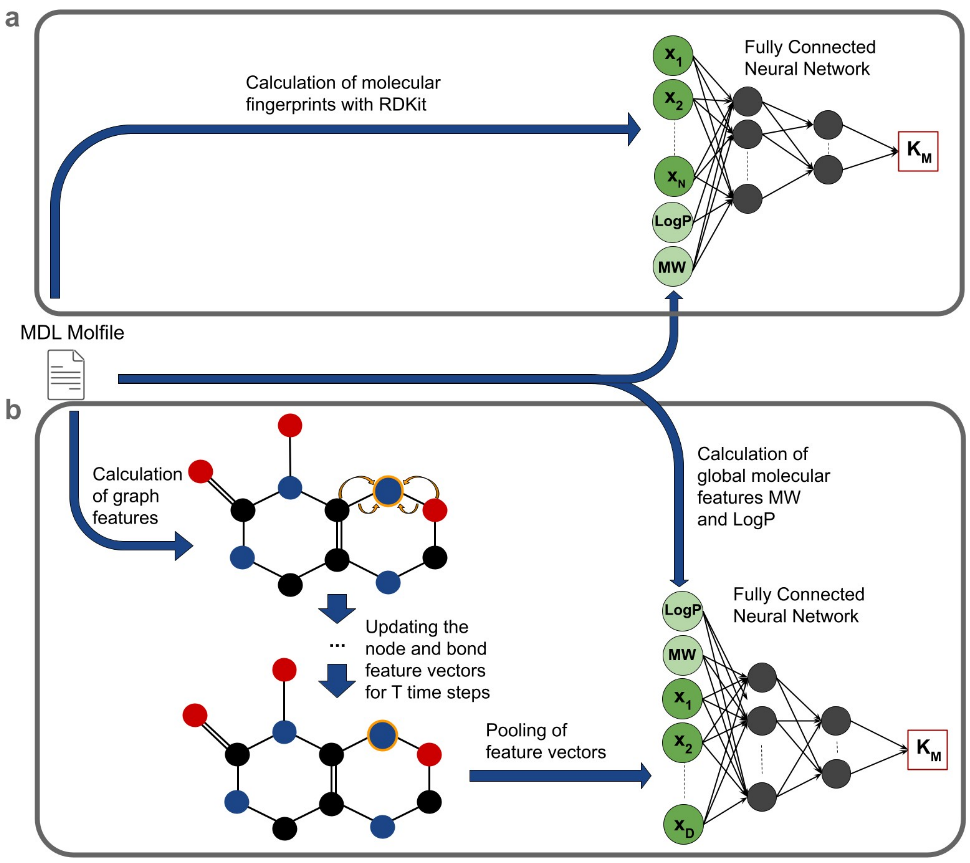

a
Calculation of molecular fingerprints with RDKit
Fully Connected Neural Network
X1
X2
XN
LogP
MW
KM
MDL Molfile
b
Calculation of graph features
Updating the node and bond feature vectors for T time steps
Pooling of feature vectors
Calculation of global molecular features MW and LogP
Fully Connected Neural Network
LogP
MW
X1
X2
XD
KM

Fig 1. Model overview. (a) Predefined molecular fingerprints. Molecular fingerprints are calculated from MDL Molfiles of the substrates and then passed through machine learning models like the FCNN together with 2 global features of the substrate, the *MW* and *LogP*. **(b)** GNN fingerprints. Node and edge feature vectors are calculated from MDL Molfiles and are then iteratively updated for *T* time steps. Afterwards, the feature vectors are pooled together into a single vector that is passed through an FCNN together with the *MW* and *LogP*. FCNN, fully connected neural network; GNN, graph neural network; *LogP*, octanol–water partition coefficient; *MW*, molecular weight.

https://doi.org/10.1371/journal.pbio.3001402.g001

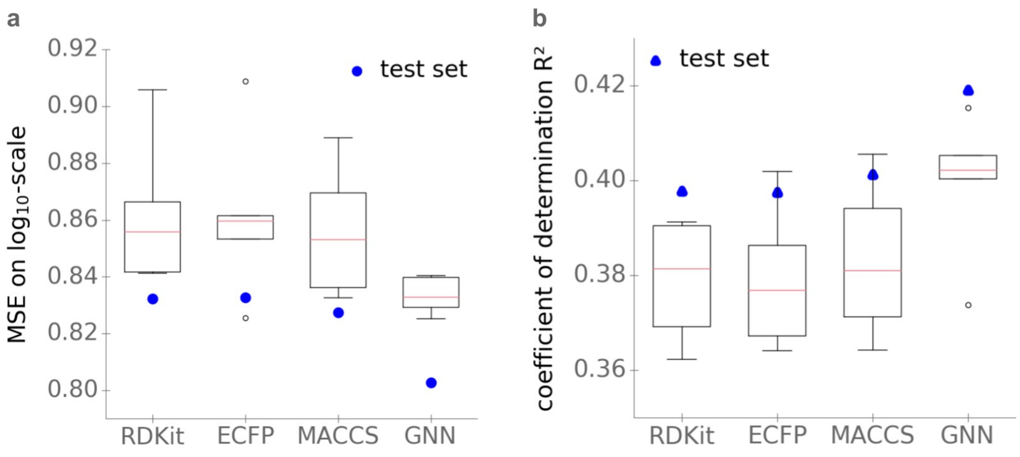


Fig 2. When using only substrate features as inputs, task-specific molecular fingerprints (GNN) lead to better K_M predictions than predefined, expert-crafted fingerprints. (a) *MSE* on $\log_{10}$-scale. **(b)** Coefficients of determination R^2. Boxplots summarize the results of the 5-fold cross-validations on the training set; blue dots show the results on the test set. The data underlying the graphs shown in this figure can be found at https://github.com/AlexanderKroll/KM_prediction/tree/master/figures_data. ECFP, extended connectivity fingerprint; GNN, graph neural network; MSE, mean squared error.

https://doi.org/10.1371/journal.pbio.3001402.g002

cross-validation on the training set, measuring performance through the mean squared error (MSE). For all 3 types of fingerprints, the gradient boosting model outperformed the FCNN and the elastic net (**S1–S3 Tables**).

The K_M predictions with the gradient boosting model based solely on the substrate ECFP, MACCS keys, and RDKit molecular fingerprints showed very similar performances on the test set, with $MSE = 0.83$ and coefficients of determination $R^2 = 0.40$ (**Fig 2**).

Best K_M predictions from metabolite fingerprints using graph neural networks and gradient boosting

Recent work has shown that superior prediction performance can be achieved through task-specific molecular fingerprints, where a deep neural network simultaneously optimizes the fingerprint and uses it to predict properties of the input. In contrast to conventional neural networks, these GNNs can process non-Euclidean inputs, such as molecular structures. This approach led to state-of-the-art performances on many biological and chemical datasets [21,22].

As an alternative to the predefined, expert-crafted molecular fingerprints, we thus also tested how well we can predict K_M from a task-specific molecular fingerprint based on a GNN (**Fig 1**; for details, see Methods, "Architecture of the graph neural network"). As for the calculations of the ECFPs, each substrate molecule is represented as a graph by interpreting the atoms as nodes and the chemical bonds as edges, for which feature vectors are calculated from the MDL Molfiles. These are updated iteratively for a fixed number of steps, in each step applying functions with learnable parameters to summarize aspects of neighboring atoms and bonds. After the iterations, the feature vectors are pooled into 1 molecular fingerprint vector. In contrast to ECFPs, the parameters of the update functions are not fixed but are adjusted during the training of the FCNN that predicts K_M from the pooled fingerprint vector (Methods). As for the predefined molecular fingerprints, we defined an extended GNN fingerprint by adding the 2 global molecular features *LogP* and *MW* to the model before the K_M prediction step.

To compare the learned substrate representation with the 3 predefined fingerprints, we extracted the extended GNN fingerprint for every substrate in the dataset and fitted an elastic net, an FCNN, and a gradient boosting model to predict K_M. As before, we performed a hyperparameter optimization with 5-fold cross-validation on the training set for all models. The gradient boosting model again achieved better results than the FCNN and the elastic net (**S1**–**S3 Tables**). The performance of our task-specific fingerprints is better than that of the predefined fingerprints, reaching an $MSE = 0.80$ and a coefficient of determination $R^2 = 0.42$ on the test set, compared to an $MSE = 0.83$ and $R^2 = 0.40$ for the other fingerprints (**Fig 2**). To compare the performances statistically, we used a one-sided Wilcoxon signed-rank test for the absolute errors of the predictions for the test set, resulting in $p = 0.0080$ (ECFP), $p = 0.073$ (RDKit), and $p = 0.062$ (MACCS keys). While the differences in the error distributions are only marginally statistically significant for RDKit and MACCS keys at the 5% level, these analyses support the choice of the task-specific GNN molecular fingerprint for predicting K_M.

It is noteworthy that the errors on the test set are smaller than the errors achieved during cross-validation. We found that the number of training samples has a great influence on model performance (see below, "Model performance increases linearly with training set size"). Hence, the improved performance on the test set may result from the fact that before validation on the test set, models are trained with approximately 2,000 more samples than before each cross-validation.

Effects of molecular weight and octanol–water partition coefficient

Before predicting K_M from the molecular fingerprints, we added the *MW* and the *LogP*. Do these extra features contribute to improved predictions by the task-specific GNN fingerprints? To answer this question, we trained GNNs without the additional features *LogP* and *MW*, as well as with only one of those additional features. **Fig 3** displays the performance of gradient boosting models that are trained to predict K_M with GNN fingerprints with and without extra features, showing that the additional features have only a small effect on performance: Adding both features reduces MSE from 0.82 to 0.80, while increasing R^2 from 0.41 to 0.42. The

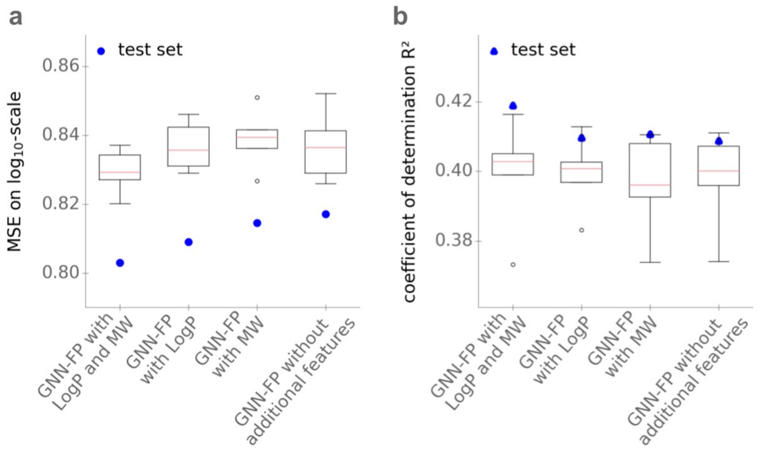


Fig 3. Adding *MW* and *LogP* as features has only a minor effect on the performance of the GNN in predicting K_M. (**a**) *MSE* on $\log_{10}$-scale. (**b**) Coefficients of determination R^2. Models use the GNN with additional features *LogP* and *MW*; with only one of the additional features; and without the 2 features. Boxplots summarize the results of the 5-fold cross-validations on the training set; blue dots show the results on the test set. The data underlying the graphs shown in this figure can be found at https://github.com/AlexanderKroll/KM_prediction/tree/master/figures_data. GNN, graph neural network; *LogP*, octanol–water partition coefficient; MSE, mean squared error; *MW*, molecular weight.

https://doi.org/10.1371/journal.pbio.3001402.g003

difference in model performance is not statistically significant ($p = 0.13$, one-sided Wilcoxon signed-rank test for the absolute errors of the predictions for the test set). This indicates that most of the information used to predict K_M can be extracted from the graph of the molecule itself. However, since the addition of the 2 additional features slightly improves K_M predictions on the test dataset, we include the features *MW* and *LogP* in our further analyses.

UniRep vectors as additional features

So far, we have only considered substrate-specific information. As K_M values are features of specific enzyme–substrate interactions, we now need to add input features that represent enzyme properties. Important information on substrate binding affinity is contained in molecular features of the catalytic site; however, active site identities and structures are available only for a small minority of enzymes in our dataset.

We thus restrict the enzyme information utilized by the model to a deep numerical representation of the enzyme's amino acid sequence, calculating an UniRep vector [24] for each enzyme. UniRep vectors are 1,900-dimensional statistical representation of proteins, created with an mLSTM, a recurrent neural network architecture for sequence modeling that combines the long short-term memory and multiplicative recurrent neural network architectures. The model was trained with 24 million unlabeled amino acid sequences to predict the next amino acid in an amino acid sequence, given the previous amino acids [24]. In this way, the mLSTM learns to store important information about the previous amino acids in a numerical vector, which can later be extracted and used as a representation for the protein. It has been shown that these representations lead to good results when used as input features in prediction tasks concerning protein stability, function, and design [24].

Predicting K_M using substrate and enzyme information

To predict the K_M value, we concatenated the 52-dimensional task-specific extended fingerprint learned with the GNN and the 1,900-dimensional UniRep vector with information about the enzyme's amino acid sequence into a global feature vector. This vector was then used as the input for a gradient boosting model for regression in order to predict the K_M value. We also trained an FCNN and an elastic; however, predictions were substantially worse (**S4–S6 Tables**), consistent with the results obtained when using only the substrate fingerprints as inputs.

The gradient boosting model that combines substrate and enzyme information achieves an $MSE = 0.65$ on a $\log_{10}$-scale and results in a coefficient of determination $R^2 = 0.53$, substantially superior to the above models based on substrate information alone. We also validate our model with an additional metric, r_m^2, which is a commonly used performance measurement tool for quantitative structure–activity relationship (QSAR) prediction models. It is defined as $r_m^2 = r^2 \times (1 - \sqrt{r^2 - r_0^2})$, where r^2 and r_0^2 are the squared correlation coefficients with and without intercept, respectively [29,30]. Our model achieves a value of $r_m^2 = 0.53$ on the test set.

Fig 4A and **4B** compare the performance of the full model to models that use only substrate or only enzyme information as inputs, applied to the BRENDA test dataset (which only contains previously unseen enzyme–substrate combinations). To predict the K_M value from only the enzyme UniRep vector, we again fitted a gradient boosting model, leading to $MSE = 1.01$ and $R^2 = 0.27$. To predict the K_M value from substrate information only, we chose the gradient boosting model with extended task-specific fingerprints as its inputs, which was used for the comparison with the other molecular fingerprints. (**Fig 2**).

Fig 4A and **4B** also compare the 3 models to the naïve approach of simply using the mean over all K_M values in the training set as a prediction for all K_M values in the test set, resulting in

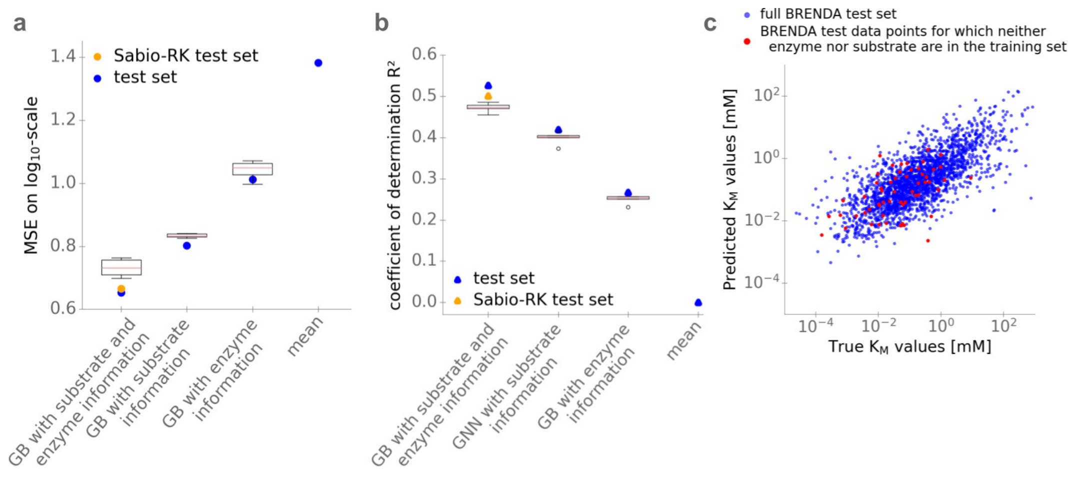


Fig 4. Performance of the optimized models. **(a)** *MSE.* **(b)** Coefficients of determination (R^2). Values in (a) and (b) are calculated using the gradient boosting model with different inputs: substrate and enzyme information; substrate information only (GNN); and enzyme information only (domain content). Boxplots summarize the results of the 5-fold cross-validations on the training set; blue dots show the results on the test set. For comparison, we also show results on the test set from a naïve model using the mean of the K_M values in the training set for all predictions. **(c)** Scatter plot of $\log_{10}$-transformed K_M values of the test set predicted with the gradient boosting model with substrate and enzyme information as inputs versus the experimental values downloaded from BRENDA. Red dots are for combinations where neither enzyme nor substrate were part of the training set. The data underlying the graphs shown in this figure can be found at https://github.com/AlexanderKroll/KM_prediction/tree/master/figures_data. GB, gradient boosting; GNN, graph neural network; MSE, mean squared error.

https://doi.org/10.1371/journal.pbio.3001402.g004

$MSE = 1.38$ and $R^2 = 0$. Fig 4C compares the values predicted using the full model with the experimental values of the test set obtained from BRENDA.

Predicting K_M for an independently acquired test dataset

Our model was trained and tested on data from BRENDA. To confirm its prediction power, it is desirable to test it on data from other sources. We thus created an additional, independent test set by obtaining the same type of information from the Sabio-RK database [31], keeping only entries that were not already included in the BRENDA dataset. This resulted in a second test set with 274 entries. The model trained on the BRENDA data achieves a very similar performance ($MSE = 0.67$, $R^2 = 0.49$) on the independent Sabio-RK test data (orange dots in S1 Fig).

Predicting K_M for enzymes and substrates not represented in the training data

Homologous enzymes that catalyze the same reaction tend to have broadly similar kinetic parameters. To test to what extent such similarities affect our results, we investigated how well our model performs for the 664 data points in the test set that have substrate–EC number combinations not found in the training set (violet dots in S1 Fig). The K_M predictions for these data points resulted in an $MSE = 0.79$ and $R^2 = 0.45$, compared to $MSE = 0.65$ and $R^2 = 0.53$ for the full test data.

It is conceivable that predictions are substantially better if the training set contains entries with the same substrate or with the same enzyme, even if not in the same combination. In practice, one may however want to also make predictions for combinations where the enzyme and/or the substrate are not represented in the training data at all. To test how our model performs in such cases, we separately analyzed those 57 entries in the test data where neither enzyme nor substrate occurred in the training data, resulting in *MSE* = 0.74 and R^2 = 0.26, compared to *MSE* = 0.65 and R^2 = 0.53 for the full test data (red points in Fig 4C). At least in part, the smaller R^2 value can be explained by the poor predictions for K_M values below 10^{-2} *mM* (see the residuals in panel **a** in S2 Fig). The training dataset contained few K_M values in this region (panel **b** in S2 Fig)—there may have simply been too little training data here for the challenging task of predicting K_M for unseen enzymes and substrates. In contrast, the model performs substantially better for unseen substrates and enzymes with K_M values between 10^{-2} and 10^0 *mM*, where much more training data were available. We conclude that given enough training data, the proposed model appears capable of predicting K_M values also for data points where substrate and/or enzyme are not in the training set.

Model performance increases linearly with training set size

The last analysis indicates that prediction performance may be strongly affected by the amount of relevant training data. Indeed, the training datasets employed for AI prediction tasks are typically vastly larger than those available for predicting K_M. To test if the size of the training set has a substantial, general effect on prediction quality, we trained the final gradient boosting model with different amounts of the available training samples. We excluded randomly data points from the original training set for this analysis, creating 6 different training sets with sizes ranging from about 4,500 to approximately 9,500 data points. Fig 5 shows that model performance—measured either in terms of *MSE* or R^2—increases approximately linearly with the size of the training set. This result indicates that our models are still far from overfitting and that increasing availability of data will allow more accurate predictions in the future.

K_M predictions for enzymatic reactions in genome-scale metabolic models

Above, we have described the development and evaluation of a pipeline for genome-scale, organism-independent prediction of K_M values. This pipeline and its parameterization can be used, for example, to obtain preliminary K_M estimates for enzyme–substrate combinations of interest or to parameterize kinetic models of enzymatic pathways or networks. To facilitate such applications, we predicted K_M values for all enzymes in 47 curated genome-scale metabolic models, (S1 Data), include models for *E. coli*, *S. cerevisiae*, *M. musculus*, and *H. sapiens*.

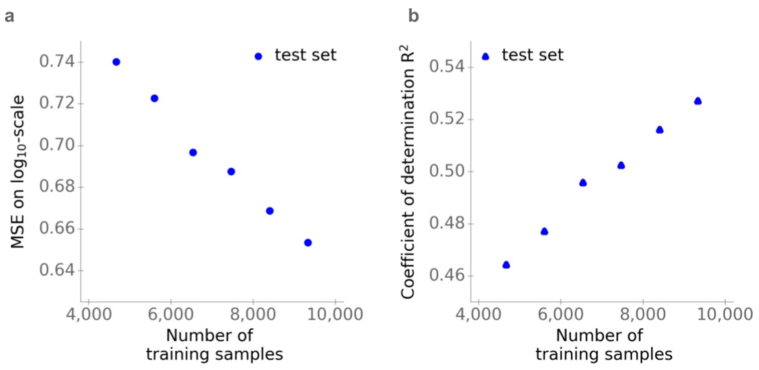


Fig 5. Effect of the training set size on model performance. (**a**) *MSE.* (**b**) Coefficients of determination (R^2). Values in (a) and (b) are calculated for the test sets, using the gradient boosting model with substrate and enzyme information as the input. The gradient boosting model is trained with different amounts of the available training samples. The data underlying the graphs shown in this figure can be found at https://github.com/AlexanderKroll/KM_prediction/tree/master/figures_data. MSE, mean squared error.

https://doi.org/10.1371/journal.pbio.3001402.g005

These models are for organisms from different domains, while the training and test data are dominated by bacteria. To test if this uneven training data distribution leads to biases, we divided our test set into subsets belonging to the domains Archaea, Bacteria, and Eukarya, calculating separate MSE and R^2 values for each domain. The test set contained 142 data points from Archaea, with $MSE = 0.71$ and $R^2 = 0.37$; 1,439 data points from Bacteria, with $MSE = 0.65$ and $R^2 = 0.51$; and 749 data points from Eukarya, with $MSE = 0.64$ and $R^2 = 0.56$. We therefore conclude that our model can predict K_M values for different domains approximately equally well.

The predictions for the genome-scale metabolic models in **S1 Data** are based on a machine learning model trained with all of the available data, including all data points from the test set. For 73% of the reactions across all 47 metabolic models, substrate and enzyme information were available, such that the full prediction model could be applied. For 15% only substrate information, for 10% only enzyme information, and for 2% neither substrate nor enzyme information were available. We treated situations with missing information as follows: If information on only one of the 2 molecules (enzyme or substrate) was available, we used the corresponding reduced prediction model (with either only UniRep vector or only extended GNN representation as input, respectively). If both substrate and enzyme information were missing, we predicted the K_M value as the geometric mean of all K_M values in our dataset.

Discussion

In conclusion, we found that Michaelis constants of enzyme–substrate pairs, K_M, can be predicted through artificial intelligence with a coefficient of determination of $R^2 = 0.53$: More than half of the variance in K_M values across enzymes and organisms can be predicted from deep numerical representations of enzyme amino acid sequence and substrate molecular structure. This performance is largely organism-independent and does not require that either enzyme or substrate are covered by the dataset used for training; the good performance was confirmed using a second, independent and nonoverlapping test set from Sabio-RK ($R^2 = 0.49$). To obtain this predictive performance, we used task-specific fingerprints of the substrate (GNN) optimized for the K_M prediction, as these appear to contain more information about K_M values than predefined molecular fingerprints based on expert-crafted transformations (ECFP, RDKit fingerprint, MACCS keys). The observed differences between GNNs and predefined fingerprints is in line with the results of a previous study on the prediction of chemical characteristics of small molecules [22].

Fig 4, which compares K_M predictions across different input feature sets, indicates that the relevant information contained in an enzyme's amino acid sequence may be less important for its evolved binding affinity to a natural substrate than the substrate's molecular structure: Predictions based only on substrate structures explain almost twice as much variance in K_M compared to predictions based only on enzyme representations. It is possible, though, that improved (possibly task-specific) enzyme representations will modify this picture in the future.

A direct comparison of the prediction quality of our model to the results of Yan and colleagues [7] would not be meaningful, as the scope of their model is very different from that of ours. Yan and colleagues trained a model specific to a single enzyme–substrate pair with only 36 data points, aiming to distinguish K_M values between different sequences of the same enzyme (beta-glucosidase) for the same substrate (cellobiose). However, the performance of our general model, with $MSE = 0.65$, compares favorably to that of the substrate-specific statistical models of Borger and colleagues [6], which resulted in an overall $MSE = 1.02$.

We compare our model to 2 different models for DTBA prediction, DeepDTA and SimBoost [10,16]. These two, which were trained and tested on the same 2 datasets, achieved r_m^2

values ranging from 0.63 to 0.67 on test sets. This compares to $r_m^2 = 0.53$ achieved for K_M predictions with our approach. It is generally difficult to compare prediction performance between models trained and tested on different datasets. Here, this difficulty is exacerbated by the different prediction targets (DTBA versus K_M). Crucially, the datasets used for DTBA and K_M prediction differ substantially with respect to their densities, i.e., the fraction of possible protein–ligand combinations covered by the training and test data. One of the datasets used for DTBA prediction encompasses experimental data for all possible drug–target combinations between 442 different proteins and 68 targets (442×68 = 30,056). The second dataset contains data for approximately 25% of all possible combinations between 229 proteins and 2,111 targets (118,254 out of 229×2,111 = 483, 419). In contrast, our K_M dataset features 7,001 different enzymes and 1,582 substrates but comprises only about 0.1% of their possible combinations (11,600 out of 7,001×1582 = 11,075, 582). Thus, our dataset is not only much smaller, but also has an extremely low coverage of possible protein–ligand combinations compared to the DTBA datasets used in [10,16]. As shown in **Fig 5**, the number of available training samples has a strong impact on model performance, and the same is likely true for the data density. Against this background, the performance of our K_M prediction model could be seen as being surprisingly good. **Fig 5** indicates that K_M predictions can be improved substantially once more training data become available.

To provide the model with information about the enzyme, we used statistical representations of the enzyme amino acid sequence. We showed that these features provide important enzyme-specific information for the prediction of K_M. It appears likely that predictions could be improved further by taking features of the enzyme active site into account—such as hydrophobicity, depth, or structural properties [5]—once such features become widely available [23]. Adding organism-specific information, such as the typical intracellular pH or temperature, may also increase model performance.

We wish to emphasize that our model is trained to predict K_M values for enzyme–substrate pairs that are known to interact as part of the natural cellular physiology, meaning that their affinity has evolved under natural selection. The model should thus be used with care when making predictions for enzyme interactions with other substrates, such as nonnatural compounds or substrates involved in moonlighting activities. In such cases, DTBA prediction models (with their higher data density) may be better suited, and estimates with our model should be regarded as a lower bound for K_M that might be reached under appropriate natural selection.

To put the performance of the current model into perspective, we consider the mean relative prediction error *MRPE* = 4.1, meaning that our predictions deviate from experimental estimates on average by 4.1-fold. This compares to a mean relative deviation of 3.4-fold between a single K_M measurement and the geometric mean of all other measurements for the same enzyme–substrate combination in the BRENDA dataset (the geometric means of enzyme–substrate combinations were used for training the models). Part of the high variability across values in BRENDA is due to varying assay conditions in the in vitro experiments [28]. Moreover, entries in BRENDA are not free from errors; on the order of 10% of the values in the database do not correspond to values in the original papers, e.g., due to errors in unit conversion [28].

Especially on the background of this variation, the performance of our enzyme–substrate specific K_M model appears remarkable. In contrast to previous approaches [6,7,13–16], the model requires no previous knowledge about measured K_M values for the considered substrate or enzyme. Furthermore, only one general purpose model is trained, and it is not necessary to obtain training data and to fit new models for individual substrates, enzyme groups, or organisms. Once the model has been fitted, it can provide genome-scale K_M predictions from existing features within minutes. We here provide such predictions for a broad set of model organisms, including mouse and human; these data can provide base estimates for unknown kinetic constants, e.g., to relate metabolomics data to cellular physiology, and can help to parameterize kinetic models of metabolism. Future work may develop similar prediction frameworks for enzyme turnover numbers (k_{cat}), which would facilitate the completion of such parameterizations.

Methods

Software and code availability

We implemented all code in Python [32]. We implemented the neural networks using the deep learning library TensorFlow [33] and Keras [34]. We fitted the gradient boosting models using the library XGBoost [35].

All datasets generated and the Python code used to produce the results (in Jupyter notebooks) are available from https://github.com/AlexanderKroll/KM_prediction. Two of the Jupyter notebooks contain all the necessary steps to download the data from BRENDA and Sabio-RK and to preprocess it. Execution of a second notebook performs training and validation of our final model. Two additional notebooks contain code to train the models with molecular fingerprints as inputs and to investigate the effect of the 2 additional features, *MW* and *LogP*, for the GNN.

Downloading and processing K_M values from BRENDA

We downloaded K_M values together with organism and substrate name, EC number, UniProt ID of the enzyme, and PubMed ID from the BRENDA database [25]. This resulted in a dataset with 156,387 entries. We mapped substrate names to KEGG Compound IDs via a synonym list from KEGG [26]. For all substrate names that could not be mapped to a KEGG Compound ID directly, we tried to map them first to PubChem Compound IDs via a synonym list from PubChem [36] and then mapped these IDs to KEGG Compound IDs using the web service of MBROLE [37]. We downloaded amino acid sequences for all data points via the UniProt mapping service [38] if the UniProt ID was available; otherwise, we downloaded the amino acid sequence from BRENDA via the organism name and EC number.

We then removed (i) all duplicates (i.e., entries with identical values for K_M, substrate, and amino acid sequence as another entry); (ii) all entries with non-wild-type enzymes (i.e., with a commentary field in BRENDA labeling it as mutant or recombinant); (iii) entries for nonbacterial organisms without an UniProt ID for the enzyme; and (iv) entries with substrate names that could not be mapped to a KEGG Compound ID. This resulted in a filtered set of 34,526 data points. Point (iii) was motivated by the expectation that isoenzymes are frequent in eukaryotes but rare in bacteria, such that organism name and EC number are sufficient to unambiguously identify an amino acid sequence in the vast majority of cases for bacteria but not for eukaryotes. If multiple $\log_{10}$-transformed K_M values existed for 1 substrate and 1 amino acid sequence, we took the geometric mean across these values. For 11,737 of these, we could find an entry for the EC number–substrate combination in the KEGG reaction database. Since we are only interested in K_M values for natural substrates, we only kept these data points [28]. We $\log_{10}$-transformed all K_M values in this dataset. We split the final dataset with 11,737 entries randomly into training data (80%) and test data (20%). We further split the training set into 5 subsets, which we used for 5-fold cross-validations for the hyperparameter optimization of the machine learning models. We used the test data to evaluate the final models after hyperparameter optimization.

To estimate the proportion of metabolic enzymes with K_M values measured in vitro for *E. coli*, we mapped the *E. coli* K_M values downloaded from BRENDA to reactions of the genome scale metabolic model *i*ML1515 [39], which comprises over 2,700 different reactions. To do this, we extracted all enzyme–substrate combinations from the *i*ML1515 model for which the model annotations listed an EC number for the enzyme and a KEGG Compound ID for the substrate, resulting in 2,656 enzyme–substrate combinations. For 795 of these combinations (i.e., 29.93%), we were able to find a K_M value in the BRENDA database.

Download and processing of K_M values from Sabio-RK

We downloaded K_M values together with the name of the organism, substrate name, EC number, UniProt ID of the enzyme, and PubMed ID from the Sabio-RK database. This resulted in a dataset with 8,375 entries. We processed this dataset in the same way as described above for the BRENDA dataset. We additionally removed all entries with a PubMed ID that was already present in the BRENDA dataset. This resulted in a final dataset with 274 entries, which we used as an additional test set for the final model for K_M prediction.

Calculation of predefined molecular fingerprints

We first represented each substrate through 3 different molecular fingerprints (ECFP, RDKit fingerprint, MACCS keys). For every substrate in the final dataset, we downloaded an MDL Molfile with 2D projections of its atoms and bonds from KEGG [26] via the KEGG Compound ID. We then used the package Chem from RDKit [19] with the Molfile as the input to calculate the 2,048-dimensional binary RDKit fingerprints [19], the 166-dimensional binary MACCS keys [20], and the 1,024-dimensional binary ECFPs [18] with a radius of 3.

Architecture of the fully connected neural network with molecular fingerprints

We used an FCNN to predict K_M values using only representations of the substrates as input features. We performed a 5-fold cross-validation on the training set for each of the 4 substrate representations (ECFP, RDKit fingerprints, MACCS keys, and task-specific fingerprints) for the hyperparameter optimization. The FCNN consisted of 2 hidden layers, and we used rectified linear units (*ReLUs*), which are defined as $ReLU(x) = max(x, 0)$, as activation functions in the hidden layers to introduce nonlinearity. We applied batch normalization [40] after each hidden layer. Additionally, we used *L2*-regularization in every layer to prevent overfitting. Adding dropout [41] did not improve the model performance. We optimized the model by minimizing the *MSE* with the stochastic gradient descent with Nesterov momentum as an optimizer. The hyperparameters regularization factor, learning rate, learning rate decay, dimension of hidden layers, batch size, number of training epochs, and momentum were optimized by performing a grid search. We selected the set of hyperparameters with the lowest mean *MSE* during cross-validation. The results of the cross-validations and best set of hyperparameters for each fingerprint are displayed in **S1 Table**.

Fitting of the gradient boosting models with molecular fingerprints

We used gradient boosting models to predict K_M values using only representations of the substrates as input features. As for the FCNNs, we performed a 5-fold cross-validation on the training set for each of the 4 substrate representations (ECFP, RDKit fingerprints, MACCS keys, and task-specific fingerprints) for hyperparameter optimization. We fitted the models using the gradient boosting library XGBoost [35] for Python. The hyperparameters regularization coefficients, learning rate, maximal tree depth, maximum delta step, number of training rounds, and minimum child weight were optimized by performing a grid search. We selected the set of hyperparameters with the lowest mean *MSE* during cross-validation. The results are displayed in **S2 Fig**.

Fitting of the elastic nets with molecular fingerprints

We used elastic nets to predict K_M values with representations of the substrates as input features. Elastic nets are linear regression model with additional *L1*- and *L2*-penalties for the coefficients of the model in order to apply regularization. We performed 5-fold cross-validations on the training set for all 4 substrate representations (ECFP, RDKit fingerprints, MACCS keys, and task-specific fingerprints) for hyperparameter optimization. During hyperparameter optimization, the coefficients for *L1*-regularization and *L2*-regularization were optimized by performing a grid search. The models were fitted using the machine learning library scikit-learn [42] for Python. The results of the hyperparameter optimizations are displayed in **S3 Table**.

Calculation of molecular weight (*MW*) and the octanol–water partition coefficient (LogP)

We calculated the additional 2 molecular features, *MW* and *LogP*, with the package Chem from RDKit [19], with the MDL Molfile of the substrate as the input.

Calculation of the input of the graph neural network

Graphs in GNNs are represented with tensors and matrices. To calculate the input matrices and tensors, we used the package Chem from RDKit [19] with MDL Molfiles of the substrates as inputs to calculate 8 features for very atom v (atomic number, number of bonds, charge, number of hydrogen bonds, mass, aromaticity, hybridization type, chirality) and 4 features for every bond between 2 atoms v and w (bond type, part of ring, stereo configuration, aromaticity). Converting these features (except for atom mass) into one-hot encoded vectors resulted in a feature vector with $F_b = 10$ dimensions for every bond and in a feature vector with $F_a = 32$ dimensions for every atom.

For a substrate with N atoms, we stored all bonds in an $N \times N$-dimensional adjacency matrix A, i.e., entry A_{vw} is equal to 1 if there is a bond between the 2 atoms v and w and 0 otherwise. We stored the bond features in a $(N \times N \times F_b)$-dimensional tensor E, where entry $E_{vw} \in \mathbb{R}^{F_b}$ contains the feature vector of the bond between atom v and atom w. Afterwards, we expanded tensor E by concatenating the feature vector of atom v to the feature vector E_{vw}. If there was no bond between the atoms v and w, i.e., $A_{vw} = 0$, we set all entries of E_{vw} to zero. We then used the resulting $(N \times N \times (F_a + F_b))$-dimensional tensor E, together with the adjacency matrix A, as the input of the GNN.

During training, the number of atoms N in a graph has to be restricted to a maximum. We set the maximum to 70, which allowed us to include most of the substrates in the training. After training, the GNN can process substrates of arbitrary sizes.

Architecture of the graph neural network

In addition to the predefined fingerprints, we also used a GNN to represent the substrate molecules. We first give a brief overview over such GNNs, before detailing our analysis.

As in the calculations of the ECFPs, a molecule is represented as a graph by interpreting the atoms as nodes and the chemical bonds as edges. Before a graph is processed by a GNN, feature vectors $\overrightarrow{x}_v$ for every node v and feature vectors $\overrightarrow{e}_{vw}$ for every edge between 2 nodes v and *w* are calculated. We calculated 8 features for every atom and 4 features for every bond of a substrate, including mass, charge, and type of atom as well as type of bond (see Methods, "Calculation of the input of the graph neural network"). The initial representations $\vec{x_v} = \vec{x_v}^{(0)}$ and $\overrightarrow{e_{vw}} = \overrightarrow{e_{vw}}^{(0)}$ are updated iteratively for a predefined number of steps *T* using the feature vectors of the neighboring nodes and edges (**Fig 1B**). During this process, the feature vectors are multiplied with matrices with trainable entries, which are fitted during the optimization of the GNN. After *k* iterations, each node representation $\vec{x_v}^{(k)}$ contains information about its k-hop neighborhood graph. After completing *T* iteration steps, all node representations are averaged to obtain a single vector $\vec{x}$, which represents the entire graph [43,44]. The vector $\vec{x}$ can then be used as an input of an FCNN to predict properties of the graph (the K_M value of the molecule in our case; **Fig 1**).

The described processing of a graph with a GNN can be divided into 2 phases. The first, message passing phase consists of the iteration process. The second, readout phase comprises the averaging of the node representations and the prediction of the target graph property [43]. During the training, both phases are optimized simultaneously. The vector $\vec{x}$ can thus be viewed as a task-specific fingerprint of the substrate. Since the model is trained end to end, the GNN learns to store all information necessary to predict K_M in this vector [44,45].

We use a variant of GNNs called directed message passing neural network (D-MPNN) [22,46]. In D-MPNNs, every edge is viewed as 2 directed edges pointing in opposite directions. During the iteration process (the message passing phase), feature vectors of nodes and edges are iteratively updated. To update them, feature vectors of neighboring nodes and edges are multiplied by matrices with learnable parameters and the results are summed. Then, an activation function, the *ReLU*, is applied to the resulting vector to introduce nonlinearities.

We set the number of iterations for updating the feature vector representations to $T = 2$. The dimension of the feature vectors during the message passing phase are set to $D = 50$. We apply batch normalization before every activation function. Additionally, we tried to apply dropout at the end of the message passing phase, but this does not improve model performance.

After the message passing phase, the readout phase starts, and feature vectors of all nodes and edges are pooled together using an order-invariant function to obtain a single vector $\vec{x} \in \mathbb{R}^D$, which is a representation of the input. The pooling is done using the element-wise mean of the feature vectors. We then concatenate $\vec{x}$ with the *MW* and the *LogP*, which are global molecular features that are correlated with the K_M value [28]. This results in an extended fingerprint $\vec{\hat{x}} = (\vec{x}^\top, MW, LogP)^\top \in \mathbb{R}^{D+2}$.

Afterwards, $\vec{\hat{x}}$ is used as the input of an FCNN with 2 layers with dimensions 32 and 16, again using *ReLUs* as activation functions. Batch normalization and L2-regularization are applied to the fully connected layers to avoid overfitting.

During training, the values of the matrices from the message passing phase and the parameters of the FCNN from the readout phase are fitted simultaneously. We trained the model by minimizing the *MSE* with the optimizer Adadelta [47] with a decaying learning rate (decay rate to $\rho = 0.95$), starting at 0.05 for 50 epochs. We used a batch size of 64, a regularization parameter $\lambda = 0.01$ for the parameters in the message passing phase, and a regularization parameter $\lambda = 1$ for the parameters in the readout phase. The hyperparameters regularization factor, learning rate, batch size, dimension of feature vectors *D*, and decay rate were optimized with a 5-fold cross-validation on the training set by performing a grid search. We selected the set of hyperparameters with the lowest mean *MSE* during cross-validation.

UniRep vectors

To obtain a 1,900-dimensional UniRep vector for every amino acid sequence in the dataset, we used Python code that is a simplified and modified version of the original code from the George Church group [24] and which contains the already trained UinRep model (available from https://github.com/EngqvistLab/UniRep50). The UniRep vectors were calculated from a file in FASTA format [48], which contained all amino acid sequences of our dataset.

Fitting of the gradient boosting model with substrate and enzyme information

We concatenated the task-specific substrate fingerprint $\overrightarrow{x} \in \mathbb{R}^{52}$ and the 1,900-dimensional UniRep vector with information about the enzyme's amino acid sequence. We used the resulting 1,952-dimensional vector as the input for a gradient boosting model for regression, which we trained to predict the K_M value. We set the maximal tree depth to 7, minimum child weight to 10.6, maximum delta step to 4.24, the learning rate to 0.012, the regularization coefficient λ to 3.8, and the regularization coefficient α to 3.1. We trained the model for 1,381 iterations. The hyperparameters regularization coefficients, learning rate, maximal tree depth, maximum delta step, number of training iterations, and minimum child weight were optimized by performing a grid search during a 5-fold cross-validation on the training set. We selected the set of hyperparameters with the lowest mean *MSE* during cross-validation.

Model comparison

To test if the differences in performance between the models with predefined fingerprints as input and the model with the task-specific fingerprint as input are statistically significant, we applied a one-sided Wilcoxon signed-rank test. The Wilcoxon signed-rank test tests the null hypothesis that the median of the absolute errors on the test set for predictions made with the model with task-specific fingerprints, $\bar{e}_1$, is greater or equal to the corresponding median for predictions made with a model with predefined fingerprints, $\bar{e}_2$ ($H_0 : \bar{e}_1 \geq \bar{e}_2$ versus $H_1 : \bar{e}_2 > \bar{e}_1$). We could reject H_0 ($p = 0.0022$ (ECFP), $p = 0.0515$ (RDKit), $p = 0.030$ (MACCS keys)), accepting the alternative hypothesis H_1.

Analogous to the described procedure, we tested if the difference in model performance between the GNNs with and without the 2 additional features, *MW* and *LogP*, is statistically significant. We could reject the null hypothesis H_0 that the median of the absolute errors on the test set for predictions made with the GNN with *MW* and *LogP* is greater or equal to the corresponding median for predictions made with the GNN without additional feature ($p = 0.0454$). To execute the tests, we used the Python library SciPy [49].

Prediction of K_M values for genome-scale models

We downloaded 46 genome-scale models from BiGG [50] and the genome-scale model yeast8 for *S. cerevisiae* [51]. We extracted all enzymatic reactions from these models and created 1 entry for every substrate in an enzymatic reaction. We extracted the KEGG Compound IDs for every substrate from the annotations of the model, if available; otherwise, we mapped the substrate names to KEGG Compound IDs via synonym lists from KEGG and PubChem in the same way as described for the substrate names in the BRENDA and Sabio-RK datasets. To obtain the enzyme information, we used the gene reaction rules, which contain the names of the involved genes. To obtain the amino acid sequence and the UniProt ID for every enzyme, we used the UniProt mapping service [38]. If multiple enzymes are given for one reaction, we made a prediction for all of the given enzymes. If an enzyme complex consisted of multiple genes, we tried to figure out which of the genes has a binding activity. Therefore, we downloaded for all of the associated UniProt IDs the GO annotations via QuickGO [52]. For every UniProt ID, we checked if a binding activity was stated in the annotations. If we found a binding activity for more than 1 UniProt ID or for none of the UniProt IDs in the enzyme complex, we did not use any enzyme information.

If enzyme and substrate information was available, we used the full model to predict K_M. If only substrate or only enzyme information was available, we used a gradient boosting model that only uses substrate or enzyme information as its input. If neither substrate nor enzyme information were available, we used the geometric mean over all K_M values in the BRENDA dataset as a prediction.

To train the gradient boosting model to predict K_M values, we used the whole BRENDA dataset for model training, including the test set.

Supporting information

S1 Table. Results of the hyperparameter optimizations of fully connected neural networks (FCNNs), which were trained to predict K_M from substrate information only. The hypeparameter optimizations were performed for each of 4 different fingerprints of the substrates with a 5-fold cross-validation on the training set.
(TIF)

S2 Table. Results of the hyperparameter optimizations of gradient boosting models, which were trained to predict K_M from substrate information only. The hypeparameter optimizations were performed for each of 4 different fingerprints of the substrates with a 5-fold cross-validation on the training set.
(TIF)

S3 Table. Results of the hyperparameter optimizations of elastic nets, which were trained to predict K_M from substrate information only. The hyperparameter optimizations were performed for each of 4 different fingerprints of the substrates with a 5-fold cross-validation on the training set.
(TIF)

S4 Table. Result of the hyperparameter optimization of a fully connected neural networks (FCNN), which was trained to predict K_M from substrate and enzyme information (GNN fingerprint and UniRep vector). The hypeparameter optimization was performed with a 5-fold cross-validation on the training set.
(TIF)

S5 Table. Result of the hyperparameter optimization of the gradient boosting model, which was trained to predict K_M from substrate and enzyme information (GNN fingerprint and UniRep vector). The hypeparameter optimization was performed with a 5-fold cross-validation on the training set.
(TIF)

S6 Table. Result of the hyperparameter optimization of an elastic net, which was trained to predict K_M from substrate and enzyme information (GNN fingerprint and UniRep vector). The hyperparameter optimization was performed with a 5-fold cross-validation on the training set.
(TIF)

S1 Fig. Scatter plot of log10-transformed K_M values predicted with the gradient boosting model with substrate and enzyme information as inputs versus the experimental values

downloaded from BRENDA and Sabio-RK. The scatter plot displays all data points of the Sabio-RK test set (orange) and all data points from the BRENDA test set with an EC number–substrate combination not present in the training set (violet). The data underlying the graphs shown in this figure can be found at https://github.com/AlexanderKroll/KM_prediction/tree/master/figures_data.
(TIF)

S2 Fig. (a) Scatter plot of measured K_M values and the absolute prediction errors of the BRENDA test data points for which neither the substrate nor the enzyme occurs in the training set. (b) Histogram with the distribution of the K_M values in the training set. The data underlying the graphs shown in this figure can be found at https://github.com/AlexanderKroll/KM_prediction/tree/master/figures_data.
(TIF)

S1 Data. Dataset in xlsx format containing complete K_M predictions for 47 genome-scale metabolic models, including those for *Homo sapiens*, *Mus musculus*, *Saccharomyces cerevisiae*, and *Escherichia coli*.
(XLSX)

Acknowledgments

We thank Hugo Dourado, Markus Kollmann, and Kyra Mooren for helpful discussions. Computational support and infrastructure was provided by the "Centre for Information and Media Technology (ZIM) at the University of Düsseldorf (Germany).

Author Contributions

Conceptualization: Alexander Kroll, David Heckmann, Martin J. Lercher.

Data curation: Alexander Kroll.

Formal analysis: Alexander Kroll.

Funding acquisition: Martin J. Lercher.

Investigation: Alexander Kroll.

Methodology: Alexander Kroll, David Heckmann.

Software: Alexander Kroll, Martin K. M. Engqvist.

Supervision: Martin J. Lercher.

Validation: Alexander Kroll.

Visualization: Alexander Kroll.

Writing – original draft: Alexander Kroll.

Writing – review & editing: Alexander Kroll, Martin K. M. Engqvist, David Heckmann, Martin J. Lercher.

References

1. Davidi D, Noor E, Liebermeister W, Bar-Even A, Flamholz A, Tummler K, et al. Global characterization of in vivo enzyme catalytic rates and their correspondence to in vitro k_{cat} measurements. Proc Natl Acad Sci. 2016; 113(12):3401±6. https://doi.org/10.1073/pnas.1514240113 PMID: 26951675

2. Khodayari A, Maranas CD. A genome-scale *Escherichia coli* kinetic metabolic model k-ecoli457 satisfying flux data for multiple mutant strains. Nat Commun. 2016; 7(1):13806. https://doi.org/10.1038/ncomms13806 PMID: 27996047

3. Saa PA, Nielsen LK. Formulation, construction and analysis of kinetic models of metabolism: A review of modelling frameworks. Biotechnol Adv. 2017; 35(8):981±1003. https://doi.org/10.1016/j.biotechadv.2017.09.005 PMID: 28916392

4. Strutz J, Martin J, Greene J, Broadbelt L, Tyo K. Metabolic kinetic modeling provides insight into complex biological questions, but hurdles remain. Curr Opin Biotechnol. 2019; 59:24±30. https://doi.org/10.1016/j.copbio.2019.02.005 PMID: 30851632

5. Heckmann D, Lloyd CJ, Mih N, Ha Y, Zielinski DC, Haiman ZB, et al. Machine learning applied to enzyme turnover numbers reveals protein structural correlates and improves metabolic models. Nat Commun. 2018; 9(1):5252. https://doi.org/10.1038/s41467-018-07652-6 PMID: 30531987

6. Borger S, Liebermeister W, Klipp E. Prediction of enzyme kinetic parameters based on statistical learning. Genome Inform. 2006; 17(1):80±7. PMID: 17503358

7. Yan SM, Shi DQ, Nong H, Wu G. Predicting K_M values of beta-glucosidases using cellobiose as substrate. Interdisciplinary Sciences: Computational Life Sciences. 2012; 4(1):46±53.

8. Thafar M, Raies AB, Albaradei S, Essack M, Bajic VB. Comparison study of computational prediction tools for drug-target binding affinities. Front Chem. 2019; 7:782. https://doi.org/10.3389/fchem.2019.00782 PMID: 31824921

9. Pahikkala T, Airola A, PietilaÈS, Shakyawar S, Szwajda A, Tang J, et al. Toward more realistic drug±target interaction predictions. Brief Bioinform. 2015; 16(2):325±37. https://doi.org/10.1093/bib/bbu010 PMID: 24723570

10. He T, Heidemeyer M, Ban F, Cherkasov A, Ester M. SimBoost: a read-across approach for predicting drug±target binding affinities using gradient boosting machines. J Chem. 2017; 9(1):1±14. https://doi.org/10.1186/s13321-017-0209-z PMID: 29086119

11. JimeÂnez J, Skalic M, Martinez-Rosell G, De Fabritiis G. K deep: protein±ligand absolute binding affinity prediction via 3d-convolutional neural networks. J Chem Inf Model. 2018; 58(2):287±96. https://doi.org/10.1021/acs.jcim.7b00650 PMID: 29309725

12. Trott O, Olson AJ. AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading, Journal of computational chemistry. 2010; 31(2):455±61. https://doi.org/10.1002/jcc.21334 PMID: 19499576

13. OÈztuÈrk H, Ozkirimli E, OÈzguÈr A. WideDTA: prediction of drug-target binding affinity [preprint]. arXiv. 2019:190204166.

14. Feng Q, Dueva E, Cherkasov A, Ester M. Padme: A deep learning-based framework for drug-target interaction prediction [preprint]. arXiv. 2018:180709741.

15. Karimi M, Wu D, Wang Z, Shen Y. DeepAffinity: interpretable deep learning of compound±protein affinity through unified recurrent and convolutional neural networks. Bioinformatics. 2019; 35(18):3329±38. https://doi.org/10.1093/bioinformatics/btz111 PMID: 30768156

16. OÈztuÈrk H, OÈzguÈr A, Ozkirimli E. DeepDTA: deep drug±target binding affinity prediction. Bioinformatics. 2018; 34(17):i821±9. https://doi.org/10.1093/bioinformatics/bty593 PMID: 30423097

17. Weininger D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. J Chem Inf Comput Sci. 1988; 28(1):31±6.

18. Rogers D, Hahn M. Extended-connectivity fingerprints. J Chem Inf Model. 2010; 50(5):742±54. https://doi.org/10.1021/ci100050t PMID: 20426451

19. Landrum G. RDKit: Open-source cheminformatics. 2006. Available from: http://www.rdkit.org.

20. Durant JL, Leland BA, Henry DR, Nourse JG. Reoptimization of MDL keys for use in drug discovery. J Chem Inf Comput Sci. 2002; 42(6):1273±80. https://doi.org/10.1021/ci010132r PMID: 12444722

21. Zhou J, Cui G, Zhang Z, Yang C, Liu Z, Wang L, et al. Graph neural networks: A review of methods and applications [preprint]. arXiv. 2018. p. arXiv:1812.08434.

22. Yang K, Swanson K, Jin W, Coley C, Eiden P, Gao H, et al. Analyzing learned molecular representations for property prediction. J Chem Inf Model. 2019; 59(8):3370±88. https://doi.org/10.1021/acs.jcim.9b00237 PMID: 31361484

23. Furnham N, Holliday GL, de Beer TA, Jacobsen JO, Pearson WR, Thornton JM. The Catalytic Site Atlas 2.0: cataloging catalytic sites and residues identified in enzymes. Nucleic Acids Res. 2014; 42(D1):D485±9. https://doi.org/10.1093/nar/gkt1243 PMID: 24319146

24. Alley EC, Khimulya G, Biswas S, AlQuraishi M, Church GM. Unified rational protein engineering with sequence-based deep representation learning. Nat Methods. 2019; 16(12):1315±22. https://doi.org/10.1038/s41592-019-0598-1 PMID: 31636460

25. Jeske L, Placzek S, Schomburg I, Chang A, Schomburg D. BRENDA in 2019: a European ELIXIR core data resource. Nucleic Acids Res. 2019; 47 (D1):D542±9. https://doi.org/10.1093/nar/gky1048 PMID: 30395242

26. Kanehisa M, Goto S. KEGG: Kyoto encyclopedia of genes and genomes. Nucleic Acids Res. 2000; 28 (1):27±30. https://doi.org/10.1093/nar/28.1.27 PMID: 10592173

27. Dalby A, Nourse JG, Hounshell WD, Gushurst AK, Grier DL, Leland BA, et al. Description of several chemical structure file formats used by computer programs developed at Molecular Design Limited. J Chem Inf Comput Sci. 1992; 32(3):244±55.

28. Bar-Even A, Noor E, Savir Y, Liebermeister W, Davidi D, Tawfik DS, et al. The moderately efficient enzyme: evolutionary and physicochemical trends shaping enzyme parameters. Biochemistry. 2011; 50(21):4402±10. https://doi.org/10.1021/bi2002289 PMID: 21506553

29. Pratim Roy P, Paul S, Mitra I, Roy K. On two novel parameters for validation of predictive QSAR models. Molecules. 2009; 14(5):1660±701. https://doi.org/10.3390/molecules14051660 PMID: 19471190

30. Roy K, Chakraborty P, Mitra I, Ojha PK, Kar S, Das RN. Some case studies on application of ªrm2º metrics for judging quality of quantitative structure±activity relationship predictions: emphasis on scaling of response data. J Comput Chem. 2013; 34(12):1071±82. https://doi.org/10.1002/jcc.23231 PMID: 23299630

31. Wittig U, Kania R, Golebiewski M, Rey M, Shi L, Jong L, et al. SABIO-RK±database for biochemical reaction kinetics. Nucleic Acids Res. 2012; 40(D1):D790±6. https://doi.org/10.1093/nar/gkr1046 PMID: 22102587

32. Van Rossum G, Drake FL. Python 3 Reference Manual. Scotts Valley, CA: CreateSpace; 2009.

33. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015. Available from: https://www.tensorflow.org/.

34. Chollet F. Keras. 2015. Available from: https://keras.io.

35. Chen T, Guestrin C. Xgboost: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 2016. p. 785±794.

36. Kim S, Chen J, Cheng T, Gindulyte A, He J, He S, et al. PubChem 2019 update: improved access to chemical data. Nucleic Acids Res. 2019; 47(D1):D1102±9. https://doi.org/10.1093/nar/gky1033 PMID: 30371825

37. Lôpez-Ibâñez J, Pazos F, Chagoyen M. MBROLE 2.0Ðfun ctional enrichment of chemical compounds. Nucleic Acids Res. 2016; 44(W1):W201±4. https://doi.org/10.1093/nar/gkw253 PMID: 27084944

38. Consortium TU. UniProt: The universal protein knowledgebase in 2021. Nucleic Acids Res. 2021; 49 (D1):D480±9. https://doi.org/10.1093/nar/gkaa1100 PMID: 33237286

39. Monk JM, Lloyd CJ, Brunk E, Mih N, Sastry A, King Z, et al. iML1515, a knowledgebase that computes *Escherichia coli* traits. Nat Biotechnol. 2017; 35(10):904±8. https://doi.org/10.1038/nbt.3956 PMID: 29020004

40. Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift [preprint]. arXiv. 2015. p. arXiv:1502.03167.

41. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res. 2014; 15(1):1929±58.

42. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine Learning in Python. J Mach Learn Res. 2011; 12:2825±30.

43. Gilmer J, Schoenholz SS, Riley PF, Vinyals O, Dahl GE. Neural message passing for quantum chemistry [preprint]. arXiv. 2017. p. arXiv:1704.01212.

44. Kearnes S, McCloskey K, Berndl M, Pande V, Riley P. Molecular graph convolutions: moving beyond fingerprints. J Comput Aided Mol Des. 2016; 30(8):595±608. https://doi.org/10.1007/s10822-016-9938-8 PMID: 27558503

45. Duvenaud DK, Maclaurin D, Iparraguirre J, Bombarell R, Hirzel T, Aspuru-Guzik A, et al. Convolutional networks on graphs for learning molecular fingerprints. Adv Neural Inf Process Syst. 2015:2224±32.

46. Dai H, Dai B, Song L. Discriminative embeddings of latent variable models for structured data. International Conference on Machine Learning; 2016. p. 2702±2711.

47. Zeiler MD. Adadelta: an adaptive learning rate method [preprint]. arXiv. 2012. p. arXiv:1212.5701.

48. Lipman DJ, Pearson WR. Rapid and sensitive protein similarity searches. Science. 1985; 227 (4693):1435±41. https://doi.org/10.1126/science.2983426 PMID: 2983426

49. Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nat Methods. 2020; 17:261±72. https://doi.org/10.1038/s41592-019-0686-2 PMID: 32015543

50. Norsigian CJ, Pusarla N, McConn JL, Yurkovich JT, DraÈger A, Palsson BO, et al. BiGG Models 2020: multi-strain genome-scale models and expansion across the phylogenetic tree. Nucleic Acids Res. 2020; 48(D1):D402±6. https://doi.org/10.1093/nar/gkz1054 PMID: 31696234

51. Lu H, Li F, SaÂnchez BJ, Zhu Z, Li G, Domenzain I, et al. A consensus S. cerevisiae metabolic model Yeast8 and its ecosystem for comprehensively probing cellular metabolism. Nat Commun. 2019; 10 (1):1±13. https://doi.org/10.1038/s41467-018-07882-8 PMID: 30602773

52. Binns D, Dimmer E, Huntley R, Barrell D, O'donovan C, Apweiler R. QuickGO: a web-based tool for Gene Ontology searching. Bioinformatics. 2009; 25(22):3045±6. https://doi.org/10.1093/bioinformatics/btp536 PMID: 19744993

3.2 MANUSCRIPT 2

Manuscript 2 is identical to the version of the paper as it is published on bioRxiv [45]. The manuscript is currently under consideration at Nature Communications.

Contributions to Manuscript 2

I designed the study together with Martin Lercher. Martin Lercher defined the overall task and suggested some additional analyses for the trained k_{cat} prediction model. I designed the rest of the study, including the choice of enzyme and reaction representations, selection of training data, and choice of machine learning algorithms. I implemented all software except for the calculation of reaction fluxes, which was executed by Nina Liebrand, and the calculation of the Codon Adaptation Index (CAI), which was implemented by Xiao-Pan Hu. I developed, implemented, and trained all machine learning models. I performed all data curation, analyses, visualizations, and I implemented a web server that allows an easy use of the trained prediction model. I wrote the original draft of the manuscript. The manuscript was reviewed and edited by Martin Lercher and me.

Turnover number predictions for kinetically uncharacterized enzymes using machine and deep learning

Alexander Kroll[1], **Xiao-Pan Hu**[1], **Nina A. Liebrand**[1], **Martin J. Lercher**[1]*

[1]Institute for Computer Science and Department of Biology, Heinrich Heine University, D-40225 Düsseldorf, Germany

*Correspondence to: Martin.Lercher@hhu.de

ABSTRACT

The turnover number k_{cat} , a measure of enzyme efficiency, is central to understanding cellular physiology and resource allocation. As experimental k_{cat} estimates are unavailable for the vast majority of enzymatic reactions, the development of accurate computational prediction methods is highly desirable. However, existing machine learning models are limited to a single, well-studied organism, or they provide inaccurate predictions except for enzymes that are highly similar to proteins in the training set. Here, we present TurNuP, a general and organism-independent model that successfully predicts turnover numbers for natural reactions of wild-type enzymes. We constructed model inputs by representing complete chemical reactions through difference fingerprints and by representing enzymes through a modified and re-trained Transformer Network model for protein sequences. TurNuP outperforms previous models and generalizes well even to enzymes that are not similar to proteins in the training set. Parameterizing metabolic models with TurNuP-predicted k_{cat} values leads to improved proteome allocation predictions. To provide a powerful and convenient tool for the study of molecular biochemistry and physiology, we implemented a TurNuP web server at https://turnup.cs.hhu.de.

Introduction

The turnover number k_{cat} is the maximal rate at which one activate site of an enzyme converts molecular substrates into products. k_{cat} is a central parameter for quantitative studies of enzymatic activities, and is of key importance for understanding cellular metabolism, physiology, and resource allocation. In particular, comprehensive sets of k_{cat} values are essential for metabolic models that consider the cost of producing or maintaining enzymes[1–9], a prerequisite for accurate simulations of cellular physiology and growth[10]. Currently, no high-throughput experimental assays exist for k_{cat} , and experiments are both time consuming and expensive. Thus, k_{cat} estimates are unavailable for most reactions; even for *Escherichia coli*, arguably the biochemically best-characterized organisms, *in vitro* k_{cat} is known for only $\sim 10\%$ of all enzyme-catalyzed reactions[11]. In genome-scale kinetic models of cellular metabolism, this issue is typically addressed by either sampling missing k_{cat} values or fitting them to large datasets[7,8,12,13]. However, these techniques typically result in inaccurate results, and fitted k_{cat} values bear little relationship to known *in vitro* estimates[7,12,13].

Recent advances in artificial intelligence have put the computational prediction of unknown k_{cat} values from *in vitro* training data into reach, and two recent publications have explored this possibility. Heckmann et al.[14] developed a k_{cat} prediction model for enzymes in *E. coli*. The model relies on detailed, expert-crafted input features such as enzyme active site properties, metabolite concentrations, experimental conditions, and reaction fluxes calculated through flux balance analysis (FBA)[15]. It achieved a coefficient of determination $R^2 \approx 0.34$ on an independent test set. However, the complete, detailed input information is only available for a small subset of enzymatic reactions even in *E. coli*, limiting the applicability of this approach. A deep learning model that requires less detailed input features, DLKcat, was recently developed by Li et al.[16]. DLKcat predicts k_{cat} using information about the enzyme's amino acid sequence and about one of the reaction's substrates, ignoring other reaction details such as products and co-substrates. In practical applications, k_{cat} predictions are most important when no experimental measurements for closely related enzymes are available, and hence general prediction models should generalize well to such cases. However, while DLKcat can in principle be applied to any enzymatic reaction, its predictions become misleading for enzymes not similar to those in the training set, as we demonstrate below.

Here, we present a general machine and deep learning approach for predicting *in vitro* k_{cat} values for natural reactions of wild-type enzymes. In contrast to previous approaches, we represent chemical reactions through numerical fingerprints that consider the complete set of substrates and products of a reaction. To capture the enzyme properties, we use fine-tuned state-of-the-art protein representations as additional model inputs (**Figure 1**). We created these enzyme representations using Transformer Networks, deep neural networks for sequence processing, which were trained with millions of protein sequences[17]. It has been shown for various prediction tasks that Transformer Networks outperform protein representations created with convolutional neural networks (CNNs)[18,19], which were used in previous models for predicting enzyme turnover numbers[16].

Our resulting Turnover Number Prediction model – TurNuP – outperforms both previous methods for predicting k_{cat}[14,16]. We show that TurNuP generalizes well even to enzymes with $< 40\%$ sequence identity to proteins in the training set. Using genome-scale, enzyme-constrained metabolic models for different yeast species[16], we demonstrate that parameterizations with TurNuP k_{cat} predictions lead to improved proteome allocation predictions. To facilitate widespread use of the TurNuP model, we not only provide a Python function for large-scale k_{cat} calculations by bioinformaticians, but we also built an easy-to-use web server that requires no specialized software (turnup.cs.hhu.de).

Results

Obtaining training and test data

We compiled a dataset that connects k_{cat} measurements with the corresponding enzyme sequences, reactant IDs, and reaction equations. The underlying data is derived from the three databases BRENDA[20], UniProt[21], and Sabio-RK[22]. Our aim was to build a turnover number prediction model for natural reactions of wild-type enzymes. We hypothesized that we do not have enough data to train a model to predict the catalytic effect of enzyme mutations or to predict the k_{cat} value of non-natural enzyme-reaction pairs, which have not been shaped by natural selection. Hence, we removed all data points with non-wild-type enzymes and all non-natural reactions (see Methods, "Data preprocessing"). We removed redundancy by deleting data that was identical to other data points in the set, and we excluded points with incomplete reaction or enzyme information. We also removed 55 outliers with unrealistically low or high measurements, i.e., reported k_{cat} values that are either very close to zero ($< 10^{-2.5}/s$) or that are unreasonably high ($> 10^5/s$)[23]. If multiple different k_{cat} values existed for the same enzyme-reaction pair, we took the geometric mean across these values.

This resulted in a final dataset with 4 271 data points, comprising 2 977 unique reactions and 2 827 unique enzymes (for more details on data preprocessing, see Methods). We $\log_{10}$-transformed all k_{cat}

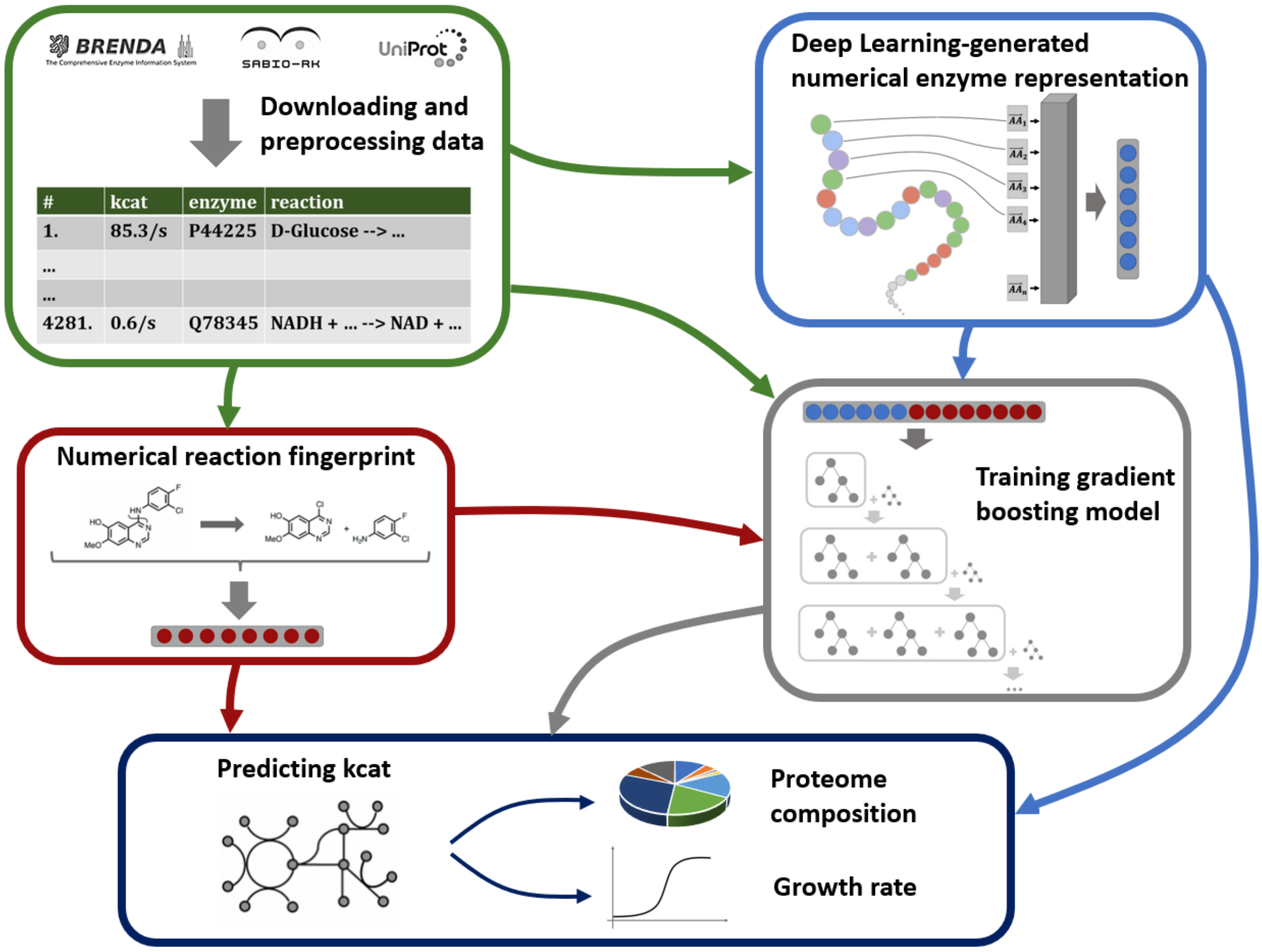

BRENDA
The Comprehensive Enzyme Information System
SABIO-RK
UniProt
Downloading and preprocessing data
# | kcat | enzyme | reaction
1. | 85.3/s | P44225 | D-Glucose --> ...
4281. | 0.6/s | Q78345 | NADH + ... --> NAD + ...
Deep Learning-generated numerical enzyme representation
Numerical reaction fingerprint
Training gradient boosting model
Predicting kcat
Proteome composition
Growth rate

Figure 1. Machine learning model to predict k_{cat} from numerical enzyme representations and reaction fingerprints. Experimentally measured k_{cat} values are downloaded from three different databases. Enzyme information is represented with numerical vectors obtained from natural language processing (NLP) models that use the linear amino acid sequence as their input. Chemical reactions are represented using integer vectors. Concatenated enzyme-reaction representations are used to train a gradient boosting model to predict k_{cat} . After training, the fitted model can be used to parameterize metabolic networks with k_{cat} values.

values to obtain a target variable with an approximately Gaussian distribution (**Figure** S1). We split the dataset into 80% training data and 20% test data in such a way that enzymes with the same amino acid sequence would not occur both in the training and in the test set. We further split the training set into 5 disjoint subsets to perform 5-fold cross validations (CVs) for hyperparameter optimization of our machine learning models. To challenge our models to learn to predict k_{cat} of enzymes without kinetically characterized close homologs, the cross validation sets were constructed such that no two subsets contained enzymes with identical amino acid sequences.

Numerical reaction fingerprints alone lead to reasonable k_{cat} predictions

The k_{cat} value of an enzyme-catalyzed reaction depends strongly on the catalyzing enzyme, but also on the chemical reaction itself. To integrate reaction information into our machine learning model, we used numerical reaction fingerprints. We compared the performance of two different types of such representations, structural and difference fingerprints (calculated with the Python package Chem from RDKit[24]).

To create structural reaction fingerprints, one first calculates for each substrate and each product a 1 638-dimensional binary molecular fingerprint, designed to encode structural information of small molecules. The bit-wise OR-function is then applied to all substrate fingerprints and separately to all product fingerprints, resulting in two 1 638-dimensional binary vectors with molecular information about the substrates and about the products, respectively. These two vectors are concatenated, providing a 3 276-dimensional binary vector with structural information about the reaction[24] (**Figure 2a**).

The calculation of difference reaction fingerprints starts with a different, 2 048-dimensional binary fingerprint for each substrate and each product. All substrate fingerprint vectors are summed to provide a single substrate vector, and all product fingerprint vectors are summed to provide a single product vector. This product fingerprint is then subtracted from the substrate fingerprint, resulting in a 2 048-dimensional reaction fingerprint with positive and negative integers[25] (**Figure 2b**).

To test how well the reaction fingerprints alone can predict the turnover numbers of enzyme-catalyzed reactions, we trained two gradient boosting models to predict k_{cat} , each with one of the reaction fingerprints as the only input. We performed a 5-fold CV with a random grid search for hyperparameter optimization for both models. After hyperparameter optimization, we chose the set of hyperparameters with the highest coefficient of determination R^2 across CV sets, and we re-trained each model with its best hyperparameters on the whole training set. On the test set, the resulting model with structural reaction fingerprints as its inputs achieves a coefficient of determination $R^2 = 0.31$, a mean squared error $MSE = 0.99$, and a Pearson correlation coefficient $r = 0.56$ on the test set. The model with difference reaction fingerprints achieves slightly improved results, with $R^2 = 0.34$, $MSE = 0.95$, and $r = 0.60$ on the test set (**Figure 3**). Thus, a model based on chemical reaction information alone can already predict about a third of the variation in k_{cat} across enzyme-catalyzed reactions.

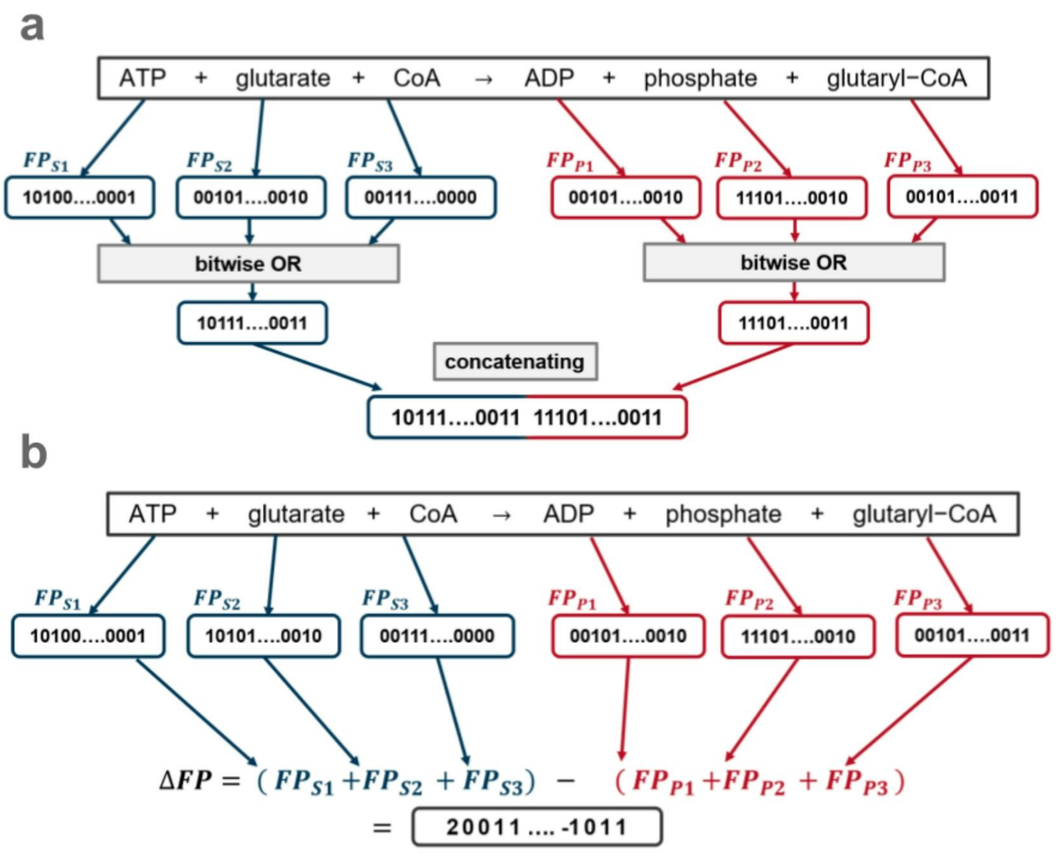


Figure 2. Calculation of reaction fingerprints for an exemplary reaction. (a) Structural reaction fingerprints. Binary molecular fingerprints are calculated for each substrate and each product. The bitwise OR-function is applied to all substrates and also to all products. The resulting substrate and the resulting product vector are then concatenated. **(b)** Difference reaction fingerprints. Binary molecular fingerprints are calculated for each substrate and each product. All substrate fingerprint vectors are summed, and the same is done for all product fingerprint vectors. To create the difference fingerprint, the resulting product vector is subtracted from the substrate vector.

To test if the better performance of difference fingerprints is statistically significant, we used a two-sided Wilcoxon signed-rank test that compared the absolute errors of the two models on the test set, resulting in $p = 0.0089$. Hence, using difference reaction fingerprints leads to statistically significant improvements, and we chose the difference reaction fingerprints to represent the catalyzed chemical reactions in the further analyses.

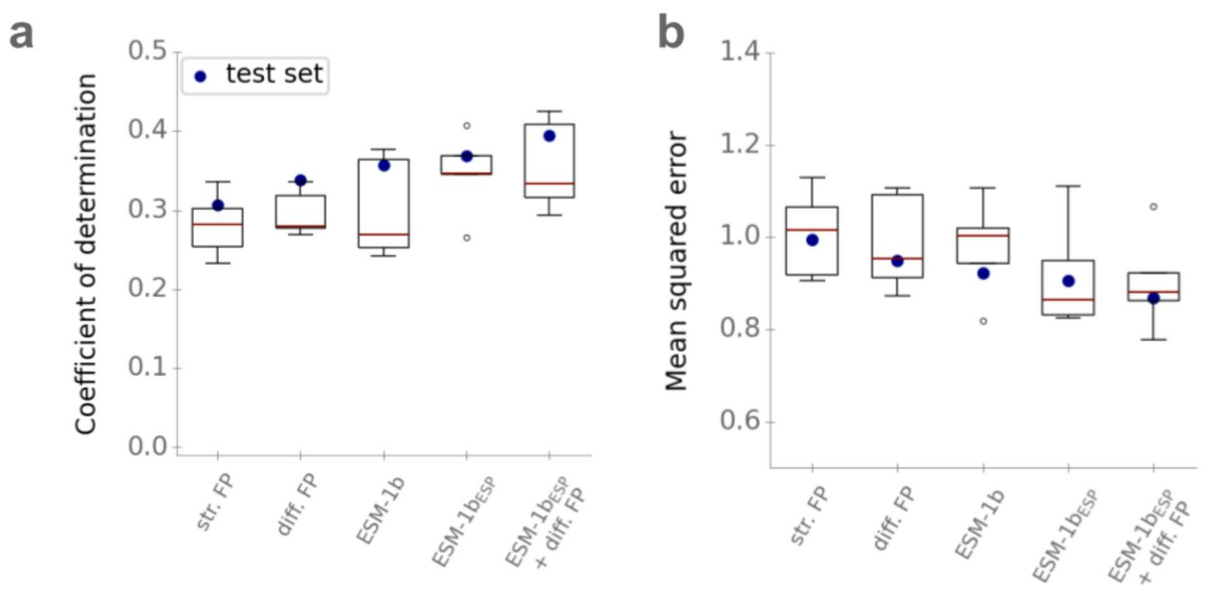


Figure 3. Using enzyme and reaction information combined leads to improved k_{cat} predictions. (a) Coefficients of determination R^2 for models with different inputs. **(b)** Mean squared errors (*MSE*) on $\log_{10}$-scale. Boxplots summarize the results of the 5-fold CVs on the training set with the best set of hyperparameters; blue dots show the results on the test set using the optimized models trained on the whole training set. Model performances are plotted for the models with structural reaction fingerprints (str. FP), difference reaction fingerprints (diff. FP), *ESM-1b* vectors (*ESM-1b*), task-specific *ESM-1b* vectors (*ESM-1b$_{ESP}$*), and with enzyme and reaction information (*ESM-1b$_{ESP}$* + diff. FP).

Numerical enzyme representations alone lead to reasonable k_{cat} predictions

The turnover number k_{cat} of an enzyme-catalyzed reaction is highly dependent on the catalyzing enzyme. It can vary by orders of magnitude even between isoenzymes that catalyze the same reaction but differ in amino acid sequence[26]. To account for this dependence when predicting k_{cat} , it is crucial to create meaningful enzyme representations as inputs to machine learning models. In recent years, deep learning architectures that were originally developed for natural language processing (NLP) tasks, such as translating a sentence from one language into another, have been applied successfully to the creation of numerical protein representations from amino acid sequences[17,27]. When applied to natural languages, NLP models typically represent all words in a sentence through numerical vectors that encode information about the words' contents and positions. When applying NLP models to protein sequences, proteins replace sentences and amino acids replace words.

The current state-of-the-art architecture for NLP tasks is a Transformer Network[28], which can, in contrast to previous methods, process all words of a sequence with arbitrary length simultaneously. The Facebook AI Research team trained such a Transformer Network, called *ESM-1b*, with a dataset of ~ 27 million protein sequences from the UniRef50 dataset[29] to create 1280-dimensional numerical protein vectors. The *ESM-1b* model was trained in a self-supervised fashion, i.e., 10-15 % of the amino acids in a sequence were masked at random, and the model was trained to predict the identity of the masked amino acids. It has been shown that the resulting representations contain rich information about the structure and the function of the proteins[17,30,31]. Using the pre-trained *ESM-1b* model[17], we calculated these 1280-dimensional representations for all enzymes in our dataset, in the following referred to as *ESM-1b* vectors.

In a previous project[30], we created a fine-tuned and task-specific version of the *ESM-1b* model that led to improved predictions for the substrate scope of enzymes, a problem for which abundant training data exists. Such comprehensive data is required to re-train the *ESM-1b* model, but is not available for k_{cat} , and we were thus unable to create a version specific to the task of predicting k_{cat} . However, we speculated that the *ESM-1b* vectors fine-tuned previously for the prediction of enzyme-substrate pairs might also improve k_{cat} predictions. To test this hypothesis, we used our previously published model, ESP[30], to calculate fine-tuned representations for all enzymes in our dataset. In the following, we will refer to these representations as $\textit{ESM-1b}_{ESP}$ vectors.

We tested how well models that use enzyme information alone can predict turnover numbers. We trained a gradient boosting model[32] that used either the *ESM-1b* or $\textit{ESM-1b}_{ESP}$ vectors to predict the k_{cat} value of enzyme-catalyzed reactions, without using any additional information on the reaction or on substrates or products. Gradient boosting models consist of many decision trees that are built iteratively during the training process. In the first iteration, a single decision tree is built that tries to predict the correct k_{cat} for all data points in the training set. In all following iterations, a new decision tree is built in order to reduce the errors that have been made by the already existing trees. After training, many different decision trees exist that ideally focus on different aspects of the input features and that try to predict the correct outcome as an ensemble[33].

To optimize the hyperparameters of the gradient boosting models, we again performed 5-fold cross validations (CV) with a random grid search on the training set. Afterwards, we re-trained each model with its best hyperparameters on the whole training set. On the test set, the model with *ESM-1b* vectors as its input achieves a coefficient of determination $R^2 = 0.36$, a mean squared error $MSE = 0.92$, and a

Pearson correlation coefficient $r = 0.60$ (**Figure 3**). The model with *ESM-1b$_{ESP}$* vectors achieves slightly improved performance, with $R^2 = 0.37$, $MSE = 0.91$, and $r = 0.61$ on the test set (**Figure 3**). Thus, a model based on enzyme information alone leads to slightly better k_{cat} predictions than a model based only on information on the catalyzed chemical reaction.

We had hypothesized that task-specific *ESM-1b$_{ESP}$* vectors would lead to improved results compared to the original *ESM-1b* vectors. Indeed, the model with *ESM-1b$_{ESP}$* input vectors led to slightly improved R^2, MSE, and Pearson r on the test set and achieved better results during CV (**Figure 3**). To test if this performance difference is statistically significant, we used a one-sided Wilcoxon signed-rank test that compared the absolute errors made by both models on the test set, resulting in $p = 0.41$. Although the difference in absolute errors is not statistically significant at the commonly used 5% level, $p < 0.5$ indicates that rejecting the null hypothesis (*ESM-1b$_{ESP}$* vectors do not lead to superior results) is more likely than not to improve the model. Thus, we chose to represent enzymes through *ESM-1b$_{ESP}$* vectors in the following.

A joint model with enzyme and reaction information leads to improved k_{cat} predictions

To train a Turnover Number Prediction model (TurNuP) with enzyme and reaction information, we concatenated the *ESM-1b$_{ESP}$* vector and the difference reaction fingerprint for every data point in our dataset. We used this resulting vector as the input for a gradient boosting model. As before, we performed a 5-fold CV with a random grid search for hyperparameter optimization, trained the model with the best set of hyperparameters on the whole training set, and validated it on the test set. The final TurNuP model achieves a coefficient of determination $R^2 = 0.40$, a mean squared error $MSE = 0.86$, and a Pearson correlation coefficient $r = 0.63$ on the test set (**Figures 3** and **4**).

Using enzyme and reaction information combined in one model improves performance compared to using only enzyme or only reaction information (**Figure 3**). To compare these differences statistically, we used a one-sided Wilcoxon signed-rank test, testing if the the absolute errors on the test set for the joint model are lower than for the models with either only enzyme or only reaction information. These tests showed that the differences are statistically significant at the 5% level, with $p = 0.043$ (difference fingerprint) and $p = 0.0046$ (*ESM-1b$_{ESP}$*). However, the improvement for the joint model is relatively small, indicating that the information stored in the reaction fingerprints and in the enzyme representations are overlapping. This overlap is not surprising, as the enzyme sequence contains information about the catalyzed reaction[30]; conversely, given that enzymes evolve on fitness landscapes shaped by the catalyzed

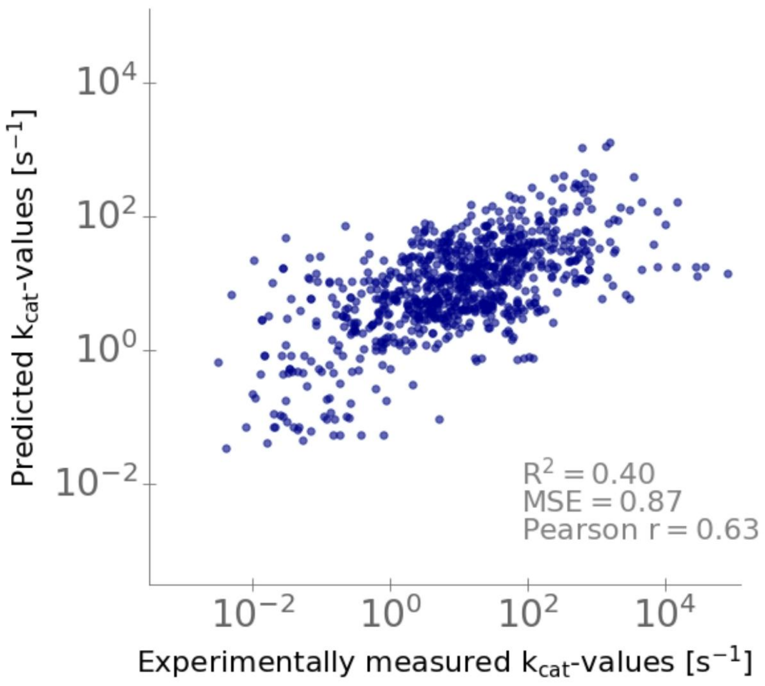


Figure 4. Comparison of predicted and experimentally measured k_{cat} values. k_{cat} values predicted with the complete TurNuP model, plotted against the corresponding experimental measurements. Each dot is one data point from the test set.

reactions, the chemical reaction likely also contains information about the type of catalyzing enzyme.

TurNuP provides meaningful predictions even if no close homologs with known k_{cat} exist

In our study on predicting the substrate scope of enzymes[30], we found that prediction performance depends strongly on the sequence similarity between a target enzyme and enzymes in the training set, consistent with the widely held belief that enzymes are more likely to be functionally similar if they have more similar sequences [34]. We hence examined the performance of TurNuP for enzyme sets that differed in their maximal similarity to proteins in the training set. We partitioned the enzymes in the test set into four subsets with 0-40%, 40-80%, 80-99%, and 99-100% maximal sequence identity to enzymes in the test set, respectively. We calculated TurNuP's coefficient of determination for all four categories (**Figure 5a**, black points). As expected, prediction performance decreases with increasing distance of the enzyme's amino acid sequence to proteins in the training set. While TurNuP's coefficient of determination is $R^2 = 0.66$ for 99-100% sequence identity, it decreases to $R^2 = 0.28$ for enzymes with a maximal sequence identity below 40%.

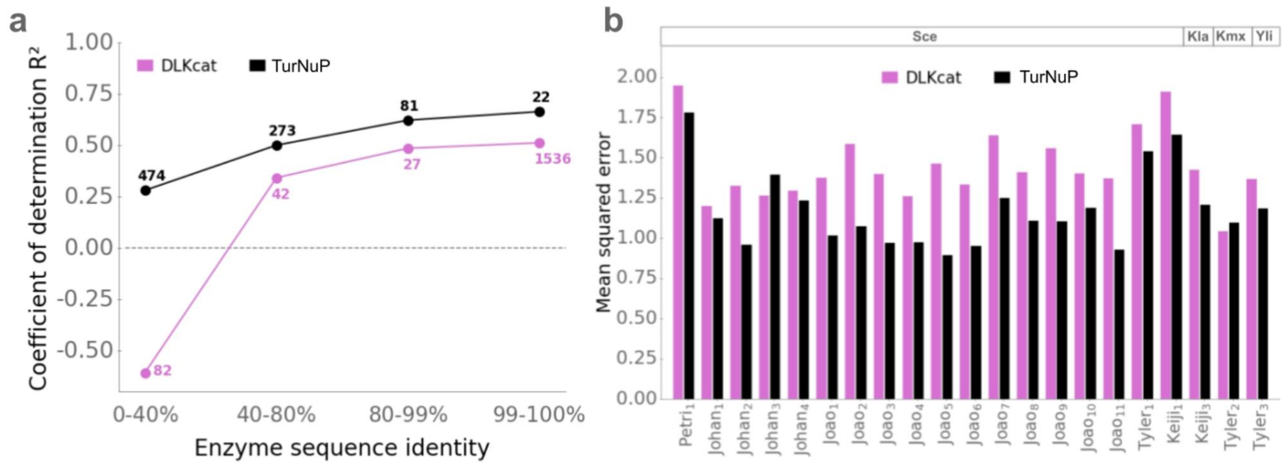


Figure 5. **Predictions are more accurate for enzymes more similar to proteins in the training set, and TurNuP predictions are more accurate compared to an existing deep learning model. (a)** Coefficients of determination R^2 for the test sets for our TurNuP model (black) and the previously published DLKcat model[16] (magenta) for different levels of maximal enzyme sequence identity compared to enzymes in the training set. Numbers next to points show how many data points of this category are in the test set. The horizontal dashed line corresponds to a model that predicts the same mean k_{cat} value for all test data points. **(b)** Mean squared errors (MSE) for the prediction of absolute proteome data compared to experimental data. Proteome predictions were achieved with enzyme-constrained genome-scale models, parameterized with k_{cat} values predicted with TurNuP (black) or with the DLKcat model (magenta). Proteome data was predicted for four different yeast species (Sce, *Saccharomyces cerevisiae*; Kla, *Kluyveromyces lactis*; Kmx, *Kluyveromyces marxianus*; Yli, *Yarrowia lipolytica*) in 21 different culture conditions (for details, see Methods).

A simple, straight-forward, and often used alternative method to predict approximate k_{cat} values is to simply average over the k_{cat} values of the most similar enzymes. Such simple averages are expected to work well in cases where kinetically characterized homologs with highly similar amino acid sequences exist; in contrast, they are unlikely to provide good estimates if no close homologs with known k_{cat} exist. As expected, for enzymes in the test set with close homologs in the training set (99-100% max. identity), the geometric mean across the three most similar enzymes in the test set leads to reasonable estimates, with $R^2 = 0.21$ ($N = 22$). In contrast, averaging over the three most similar enzymes leads to a dismal $R^2 = 0.02$ ($N = 474$) if no close homologs exist in the training data (0-40% max. identity).

These results demonstrate that any sophisticated prediction model for turnover numbers will be most relevant for enzymes for which no close homologs with known k_{cat} exist. As expected, TurNuP predictions are statistically significantly better than those provided by simple averages, across the complete test set ($R^2 = 0.40$ vs. $R^2 = 0.24$, $N = 851$, $p = 0.00040$ from one-sided Wilcoxon signed-rank test) as well as for the similarity classes (0-40% max. identity: $p = 2.2 \times 10^{-7}$, $N = 22$; 99-100% max. identity: $p = 0.0054$, $N = 474$).

TurNuP outperforms previous models for predicting k_{cat}

Heckmann et al.[14] trained and validated a machine learning model for the prediction of k_{cat} values for *E. coli*. As enzyme-related input features, their model used enzyme molecular weight and global structural disorder, as well as several molecular details of the active site: number of residues, solvent access, depth, hydrophobicity, secondary structure, and exposure. Additional input features were reaction flux, number of substrates, the dissociation constant K_M, EC number, substrate and product concentration, thermodynamic efficiency, and the pH value and temperature at which k_{cat} was measured *in vitro*. Out of this large set of features, the most important input was found to be the reaction flux, which was calculated by performing parsimonious flux balance analyses (pFBA)[35,36]. The total number of training and validation data points was limited to 215, as Heckmann et al.[14] only considered reactions from *E. coli*, and as many input features are not available for most enzymes – the least widely available features were information about the enzymes' active site, and the pH and temperature of the *in vitro* experiment. The model achieved a coefficient of determination $R^2 \approx 0.34$ on a test set. In comparison, our general model, which can be applied to enzymes from any organism and which does not require any enzyme information beyond the linear amino acid sequence, achieves an improved coefficient of determination of $R^2 = 0.40$ on a test set.

The DLKcat model by Li et al.[16] examined the same problem addressed here, the prediction of k_{cat} values across the space of all possible enzymatic reactions. Therefore, we undertook a more in-depth comparison to DLKcat. We partitioned the enzymes in the DLKcat test set according to their maximal sequence identity to proteins in the DLKcat training set, analogous to the classification of enzymes in our own test set. **Figure 5a** shows that TurNuP (black) achieves substantially higher coefficients of determination than DLKcat (magenta) for all categories of enzyme sequence identity. We tested if the differences in model performance are statistically significant using one-sided Wilcoxon–Mann–Whitney tests; for each subset, the test compares the distributions of absolute errors of the two models. For three of the four subsets, the improvements are statistically significant, with p-values of $p = 1.5 \times 10^{-10}$ (0-40%), $p = 0.0039$ (40-80%), and $p = 0.0025$ (80-99%). With $p = 0.080$, the p-value for the forth subset (99-100%) is slightly higher than the customary significance level of 5%; this might be related to the small sample size ($N = 22$) of the TurNuP test set in this category, caused by our decision to make sure that amino acid sequences in the test set are distinct from those in the training set.

We also compared the performance of DLKcat to the simple strategy of taking the geometric mean across the k_{cat} values of the three most similar enzymes in the training set. Across all data points in the test set, k_{cat} predictions by DLKcat ($R^2 = 0.45$, $N = 1\,687$) were only marginally better than those provided by simple averages ($R^2 = 0.44$; $p = 0.0033$ from one-sided Wilcoxon signed-rank test). This surprising similarity in prediction quality between DLKcat and simple k_{cat} averages is likely related to the strong sequence similarities between DLKcat's training and test sets. 68% ($N = 1\,142$) of the enzymes in the test set are also included in the training set (100% max. sequence identity), and a further 23% ($N = 394$) are at least 99% identical to amino acid sequences in the training set (but not 100% identical). In contrast to TurNuP, DLKcat was not challenged during its training to predict k_{cat} values for enzymes with dissimilar amino acid sequences[16]. As a consequence, using the geometric mean across k_{cat} values of similar enzymes is a major improvement in cases where no close homologs with known k_{cat} exist (0-40% max. identity). While simple averages achieve $R^2 = 0.11$ in this case, DLKcat predictions lead to $R^2 = -0.61$ ($p = 0.008$, two-sided Wilcoxon signed-rank test). The strongly negative coefficient of determination shows that for enzymes without kinetically characterized close homologs in the training data, DLKcat predictions are substantially worse than a trivial model that "predicts" the same mean k_{cat} value independent of the enzyme and the reaction. In sum, using DLKcat appears to be no improvement over the simple approach of calculating an average k_{cat} value across the most similar enzymes with available turnover numbers.

Although TurNuP performs better than DLKcat in each of the four categories of enzyme sequence identities (**Figure 5a**), DLKcat achieves a higher R^2 value ($R^2 = 0.44$) on its overall test set, compared to the TurNuP model on its overall test set ($R^2 = 0.40$). This counter-intuitive observation is an example of Simpson's paradox. It is caused by the differential distribution of data points across categories in the DLKcat and TurNuP test sets. As shown by the numbers above/below points in **Figure 5a**, 91% of the data points in the DLKcat test set fall into the 99-100% identity class, while the majority of data points in the TurNuP test set (56%) have less than 40% sequence identity to any enzymes in the corresponding training set and are hence much harder to predict.

Li et al.[16] designed a pipeline to use predicted k_{cat} values for the parameterization of enzyme-constrained genome-scale metabolic models, with the goal of predicting the proteome allocation patterns of yeast species. They compared the resulting proteome predictions to absolute proteomics measurements for four yeast species in 21 different environments. We employed this pipeline to test if k_{cat} values calculated with the TurNuP model lead to improved proteome predictions. In 19 out of 21 environment-species combinations, our k_{cat} values led to improved predictions ($p = 0.00010$, one-sided binomial test). The mean squared errors between measured and predicted protein abundances improved on average by $\sim 18\%$ when using TurNuP (**Figure 5b**).

Using additional input features does not improve model performance

TurNuP employs very general input features, using only the enzyme's linear amino acid sequence and information on the reaction's substrates and products. However, it is unclear if these features cover all important aspects for predicting k_{cat} . To test if we can improve prediction quality, we examined three potential additional input features: Michaelis constants K_{M}, Codon Adaptation Indices (CAIs), and reaction fluxes.

The Michaelis constant K_{M} is defined as the substrate concentration at which an enzyme works at half of its maximal catalytic rate; hence, K_{M} quantifies the affinity of an enzyme for its substrate. It has been shown that k_{cat} is correlated with the enzyme's Michaelis constant(s) of the reaction's substrate(s)[23]. To utilize this correlation for the prediction of k_{cat} , we determined K_{M} values for all enzyme-substrate combinations in our dataset. Where available, we extracted suitable K_{M} values from the BRENDA database[20] ($\sim 7\%$ of the enzyme-substrate pairs in our dataset); for all other cases we applied a machine learning model that uses numerical representations of the substrate and the enzyme as its input to predict K_{M}[37]. For reactions with multiple substrates, we took the geometric mean of all K_{M} values to obtain a single K_{M} value for every data point. To calculate how much variance of k_{cat} can be explained by K_{M}, we fitted a linear regression model to the training set, with the $\log_{10}$-transformed K_{M} value as the only input. The linear regression model achieves a coefficient of determination $R^2 = 0.11$, a mean squared error $MSE = 1.28$, and a Pearson correlation coefficient $r = 0.34$ on the test set (**Figure S2**). We thus considered K_{M} a promising candidate for improving the TurNuP predictions.

The second additional input feature, the Codon Adaptation Index (CAI), quantifies the synonymous codon usage bias of protein-coding genes. It is widely used as an indicator of gene expression and protein levels, with highly expressed genes typically using more 'preferred' codons than less highly expressed genes[38]. The CAI is a value between 0 and 1 that describes the similarity of synonymous codons frequencies between a given gene and a set of highly expressed genes, where values close to 1 indicate nearly optimal codon usage, typically associated with a high expression level in evolutionarily relevant environments. We calculated CAI for all enzymes in our dataset originating from *E. coli*. We fitted a linear regression model to the corresponding 237 data points in the training set, with CAI as the only input feature. We validated the model on 66 test data points (**Figure** S3). The model achieved a coefficient of determination $R^2 = 0.012$, a $MSE = 1.31$, and a Pearson correlation coefficient $r = 0.12$ on the test set, indicating that CAI cannot explain much of the variance of k_{cat} values. Hence, we did not consider CAI a promising candidate for improving the TurNuP predictions, and we did not calculate CAI for other organisms beyond *E. coli*.

The most important input feature in the k_{cat} prediction model established by Heckmann et al. for reactions in *E. coli*[14] was an estimate of the reaction flux, calculated using parsimonious flux balance analysis (pFBA)[35,36] across a broad range of nutrient conditions. For 108 metabolic genome scale models from the BiGG database[39], we calculated fluxes in a similar way as Heckmann et al. (Methods). For further analyses, we selected the six BiGG models of distinct species that showed the highest Pearson correlation between predicted fluxes and measured k_{cat} values in the training set. We mapped the calculated fluxes to k_{cat} values from our dataset. In cases where no metabolic genome scale model was available for an organism, we mapped the flux of an identical reaction but from a different organism to the data point. If we were not able to find the identical reaction in the BiGG database, we selected the most similar one using a similarity score (see Methods). To calculate how much variance of the k_{cat} values can be explained by the calculated fluxes, we fitted a linear regression model to the training set, with the $\log_{10}$-transformed fluxes as the only input. The fitted model achieves a coefficient of determination $R^2 = 0.021$, a $MSE = 1.40$, and a Pearson correlation coefficient $r = 0.15$ on the test set (**Figure** S4). Thus, we found no evidence for a high predictive power of fluxes beyond *E. coli*; however, as fluxes were the most important predictor for k_{cat} in Ref.[14], we still retained them as a potential additional input feature for TurNuP.

To test if adding K_{M} and reaction flux as input features improves model performance, we trained a new model. As the model input, we created a concatenated vector comprised of the enzyme *ESM-1b*$_{ESP}$ vector, the difference reaction fingerprint, the reaction flux, and the Michaelis constant K_{M} for every data point. For a gradient boosting model, we then performed a 5-fold CV with a random grid search for hyperparameter optimization. Afterwards, we trained the model with the best set of hyperparameters on the complete training set. On the test set, this model achieves a coefficient of determination $R^2 = 0.39$, a $MSE = 0.87$, and a Pearson correlation coefficient $r = 0.63$. Thus, model performance did not improve compared to the model without the additional input features flux and K_{M}.

The TurNuP web server provides an easy acces to the prediction model

We implemented a web server that facilitates an easy use of the TurNuP model without requiring programming skills or the installation of specialized software. It is available at https://turnup.cs.hhu.de. As input, the web server requires an enzyme amino acid sequence and representations of all substrates and all products; the latter can be provided either as SMILES strings, KEGG Compound IDs, or InChI strings. Users can enter a single enzymatic reaction into an online form, or upload a CSV file with multiple reactions. Since TurNuP was trained only with natural reactions of wild-type enzymes, we recommend to use the web server only for such enzyme-reaction pairs.

Discussion

Predicting the turnover number of enzyme-catalyzed reactions is a complex task, and the available datasets for model training are small and noisy. For example, Bar-Even et al.[23] found that that up to 20% of the entries in BRENDA differ from the entries in the reference papers, probably caused by copying errors and erroneous replacements of units. Even aside from such obvious errors, the variance of k_{cat} measurements for the same enzyme-reaction pairs between different studies can be high. We found an average deviation of 5.7-fold (mean deviation on $\log_{10}$-scale = 0.75) between two k_{cat} measurements for the same enzyme-reaction pair. This variance is likely not only due to errors in the databases, but also to different experimental procedures or varying assay conditions, such as temperature and pH value. When comparing a single measurement to the geometric mean of all other measurements for the same enzyme-reaction pair, we found an average deviation of 3.3-fold (mean deviation on $\log_{10}$-scale = 0.52). This compares to an average deviation of 5.1-fold (mean deviation on $\log_{10}$-scale = 0.71) of predicted k_{cat} values with our TurNuP model compared to the geometric mean of all available measurements for this enzyme-reaction pair. These numbers indicate that in practice, using predictions calculated with the TurNuP model may lead to similar deviations and error rates compared to performing experimental measurements.

Although the accuracy of TurNuP's predictions is not very different from that of experimental estimates, model accuracy can still be improved. On the one hand, we trained and validated the model with a total of only 4 271 data points, which is rather small for a machine learning model with high-dimensional input vectors. Once more high-quality training data becomes available, model performance will most likely improve. On the other hand, k_{cat} values can differ widely if measured under different experimental conditions such as varying pH and temperature. However, as information about the experimental conditions is mostly unavailable in databases for enzyme kinetic parameters, we were not able to include these conditions as an input to our prediction model. Manually extracting this information from research papers has the potential to further improve accuracy of the prediction models and to create models that are capable of accounting for different experimental conditions. Moreover, as we have shown above, the high variability of experimental estimates for the same enzyme-reaction pairs indicates a lot of noise across the measured k_{cat} values. Better predictions will become possible in the future if experimental variation will be reduced through improved technologies.

TurNuP achieves superior performance compared to previous methods for predicting k_{cat} . Its coefficient of determination ($R^2 = 0.4$) is higher than than of Heckmann et al. ($R^2 \approx 0.34$)[14], who trained an organism-specific prediction model with very detailed and expert-crafted input features, including enzyme active site properties, metabolite concentrations, reaction fluxes, and experimental conditions. TurNuP also outperforms the most recent method for predicting k_{cat} , the DLKcat model[16] (**Figure 5**). One reasons for TurNuP's superior performance might be the use of state-of-the-art enzyme representations compared to convolutional neural networks (CNNs) and the use of representations for the whole chemical reactions instead of using only information on one of the substrates.

An additional important reason might be a careful preprocessing of the k_{cat} dataset. We excluded all data points with mutated enzymes and non-natural reactions, because we are mainly interested in predicting turnover numbers for natural reactions of wild-type enzymes, and we hypothesized that we do not have enough training data to teach our model to predict the catalytic effect of enzyme mutations or to predict the k_{cat} value of non-natural enzyme-reaction pairs. When including mutated enzymes, one has to be very careful when splitting the dataset into training and test set, as one may easily end up with many nearly identical enzymes in both sets. 91% of the enzymes in the DLKcat test set have a maximal sequence identity between 99 and 100% compared to the enzymes in the training set. It is likely that the same issue arose in the validation set used for hyperparameter optimization; such a structure of training and validation sets makes it difficult to train a model that generalizes well to enzymes not highly similar to those in the training set. Indeed, we showed that DLKcat does not produce meaningful predictions for enzymes with a maximal sequence identity lower than 40% compared to the enzymes in the training set, and total model performance on the test set is not meaningfully better than calculating k_{cat} averages across the most similar enzymes in the training set. In comparison, less than 3% of the enzymes in the TurNuP test set have amino acid sequences that are >99% identical to those of homologs in the training data. Moreover, TurNuP generalizes well even to enzymes that are not highly similar to enzymes in the training set, and provides a major improvement over simple k_{cat} averages.

To achieve these results, we used general input features: the *ESM-1b$_{ESP}$* vector[17], a fine-tuned, state-of-the-art numerical representation of the enzyme, calculated from its amino acid sequence; and a reaction fingerprint that integrates structural information about all substrates and products[24], which allowed us to create input vectors of fixed length even for varying numbers of reactants. Surprisingly, we found that the *ESM-1b$_{ESP}$* vectors work very well if used as the only input feature (**Figure 3**). The reason for that is neither that enzymes with high fluxes or with high expression levels have different enzyme representations, because these features by themselves do not predict much variance of the k_{cat} values (see Results, ”Using additional input features does not improve model performance”).

It is at first sight surprising that the reaction fluxes estimated with pFBA do not explain much of the variance of k_{cat} , while they were found to be the best predictor in the model developed by Heckmann et al.[14]. When calculating genome-scale reaction fluxes for different organisms, for many data points, we obtained fluxes that were zero or close to zero. In contrast, Heckmann et al. focused on a small dataset that mostly consisted of well-studied, central reactions in *E. coli*. Those reactions typically have fluxes substantially different from zero at least in some of the simulated conditions. It appears likely that this biased construction of a small dataset in Ref.[14] is responsible for the high correlation observed between reaction fluxes and k_{cat} by Heckmann et al..

Computational estimates of k_{cat} values are highly relevant for the functional and kinetic study of individual enzymes[40], and TurNuP can provide a first estimate of k_{cat} before performing labor-intensive experiments. Another major use case of TurNuP is the prediction of k_{cat} values for genome-scale metabolic models. We found that our predictions can be used successfully to improve proteome allocation predictions (**Figure 5b**). In future work, k_{cat} predictions with TurNuP can be combined with an existing approach for predicting Michaelis constants (K_{M})[37]. This would facilitate full parameterizations of non-linear enzyme kinetics in genome-scale metabolic models, a powerful tool for gaining fundamental insights into cellular physiology[9,41].

Methods

Software and code availability

All software was coded in Python[42]. We created the enzyme representations using the deep learning library PyTorch[43]. We fitted the gradient boosting models using the library XGBoost[32]. We used the web framework Django[44] to implement the TurNuP web server. The code used to generate the results of this paper, in the form of Jupyter notebooks, as well as all datasets, are available from `https://github.com/AlexanderKroll/Kcat_prediction`.

Downloading k_{cat} data

We used data from three different databases, Sabio-RK, UniProt, and BRENDA, to create a k_{cat} dataset for model training and validation. We downloaded 3 971 k_{cat} values for wild-type enzymes together with UniProt IDs and reaction information from Sabio-RK[22]. We tried to map all metabolites involved in the reactions to unique identifiers using either a KEGG reaction ID[45], if available, or using the metabolite names and the PubChem synonym database[46]. We removed all data points for which we could not map all substrates and all products to an ID. This resulted in a dataset with 2 830 data points for 289 different enzymes.

We downloaded 5 664 k_{cat} values for wild-type enzymes together with UniProt IDs and CHEBI reaction IDs from UniProt via the UniProt mapping service[21]. We mapped the metabolites of all reactions to unique IDs using CHEBI reaction IDs[47]. We removed data points, if we could not map all metabolites of a reaction to an ID. This resulted in a dataset with 1 738 k_{cat} values for 1 017 different enzymes.

We downloaded 14 165 turnover numbers for wild-type enzymes with protein information and substrate names from BRENDA[20]. Most of the k_{cat} values in BRENDA are not assigned with a unique reaction equation and the entered k_{cat} values are known to be prone to errors[23]. To overcome these issues, we manually checked for more than half of all points if the stated k_{cat} value is identical to the value from the original paper and we assigned a unique reaction equation to all manually checked data points. After removing those data points with incomplete reaction information and non unique enzyme IDs, 8 267 data points were left for 3 149 different enzymes.

Data preprocessing

We merged all three k_{cat} datasets from BRENDA, Sabio-RK, and UniProt, which resulted in a dataset with 12 835 data points. We removed 1 050 duplicated data points from this data set. To obtain protein sequences for all enzymes, we used the UniProt mapping service[21] to map all UniProt IDs to amino acid sequences. We used the Python package Bioservices[48] to map all metabolites to InChI strings[49]. If multiple k_{cat} values existed for the same enzyme-reaction combination, we took the geometric mean across these values. For the calculation of the geometric mean, we wanted to ignore those values that were likely obtained under non-optimal conditions. Thus, we excluded k_{cat} values smaller than 1% compared to the maximal k_{cat} value for the same enzyme-reaction combination. Calculating the geometric mean resulted in a dataset with 7 496 entries.

The BRENDA, UniProt, and Sabio-RK databases contain many k_{cat} values that were measured for secondary, non-natural reactions of enzymes. As we are only interested in measurements for the natural reaction of an enzyme, we excluded k_{cat} values if another measurement existed for the same enzyme but for a different reaction with a k_{cat} value that was more than ten times higher. To further exclude data points that were measured under non-optimal conditions or for non-natural reactions of the enzyme, we excluded data points if we could find a measurement for the same reaction or the same EC number that was more than 100 times higher. The described procedures led to the removal of 3 092 data points.

We calculated reaction fingerprints and enzyme representations for all enzyme-reaction pairs (see below) and removed all 26 data points, where either the reaction fingerprint or the enzyme representation could not be calculated.

To exclude data points with possibly wrongly assigned reaction equations, we removed those 52 data points where the sum of molecular weights of substrates did not match the sum of molecular weights of the products. We removed another 55 data points because their k_{cat} values are outliers (i.e., values below $10^{-2.5}/s$ or higher than $10^5/s$). This resulted in a final dataset with 4 271 data points.

Splitting the dataset into training and test set

We randomly split the dataset into 80% training data and 20% test data. We made sure that the same enzyme would not occur in the training and the test set. We further split the training set into 5 disjoint subsets for a 5-fold cross-validation (CV) to perform hyperparameter optimizations of the machine learning models. In order to achieve a model that generalizes well during CV, we created these 5 subsets also in such a way that the same enzyme did not occur in two different subsets.

Calculating enzyme representations

To create the *ESM-1b* model[17], the Facebook AI research (FAIR) team trained a Transformer Network[28] with 33 hidden layers and a hidden layer size of 1 280 using $\sim$ 27 million protein amino acid sequences from the UniRef50 dataset[29]. To process a protein sequence, the type and position of every amino acid in a sequence is encoded in a 1 280-dimensional numerical vector. All amino acid representations of a sequence are simultaneously applied to the *ESM-1b* model and updated for 33 time steps using the attention mechanism[28]. The attention mechanism allows to use all representations as an input when updating a single amino acid representation. The attention mechanism selectively chooses only relevant input when calculating an update of a representation. To train the *ESM-1b* model, randomly $10-15\%$ of the amino acids in a sequence are masked. The model is then trained to predict the type of the masked amino acids. After training, a single representation for the whole model can be created by calculating the element-wise mean of all amino acid representations after they were updated for 33 times.

We used the trained *ESM-1b* model and the code provided on the GitHub repository of the FAIR team[17], to calculate a 1 280-dimensional numerical representation for every enzyme in our dataset. As the *ESM-1b* model can only process amino acid sequences up to 1 024 amino acids, we only used the first 1 024 amino acids for those sequences that were too long.

To calculate the fine-tuned enzyme representations that were originally created for the task of predicting the substrate scope of enzymes[30], the $\textit{ESM-1b}_{ESP}$ vectors, we used code and models provided on the following GitHub repository: `https://github.com/AlexanderKroll/ESP`.

Calculating reaction fingerprints

To calculate difference and structural reaction fingerprints, we first represented all reactions in our dataset using the language SMARTS[50]. SMARTS can be used to describe patterns of small molecules and of chemical reactions. To calculate the reaction fingerprints, we used functions from the RDKit[24] package Chem with the reaction SMARTS as the input.

Structural reaction fingerprints are created by first calculating 1638-dimensional binary molecular fingerprints (ExplicitBitVect) for all substrates and products. Then, the bitwise OR-function is separately applied to all substrate fingerprints and to all product fingerprints, which results in two 1638-dimensional binary vectors with information about the substrates and about the products, respectively. Finally, both vectors are concatenated, which results in a 3276-dimensional binary vector with structural information about the reaction. We used the RDKit function *Chem.rdChemReactions.CreateStructuralFingerprintForReaction* to calculate the fingerprints.

To calculate difference reaction fingerprints, first, a 2048-dimensional binary atom-pair fingerprint (AtompairFP) for each substrate and each product is calculated. Then, the fingerprints for all substrates and also for all products are element-wise summed. The resulting fingerprint for the products is then subtracted from the fingerprint for the substrates, which results in a 2048-dimensional reaction fingerprints with positive and negative integers. To calculate these fingerprints, we used the RDKit function *Chem.rdChemReactions.CreateDifferenceFingerprintForReaction*.

Hyperparameter optimization for gradient boosting models

To perform hyperparameter optimizations for all gradient boosting models, we split the training set into five disjoint subsets with approximately equal sizes to perform 5-fold cross-validations (CVs). We performed a random grid search for the hyperparameters learning rate, regularization coefficients α and λ, maximal tree depth, maximum delta step, number of training iterations, and minimum child weight using the Python package hyperopt[51]. Afterwards, we chose the set of hyperparameters that led to the highest mean coefficient of determination R^2 during CV.

Comparison of k_{cat} predictions between the DLKcat model and TurNuP

We used code provided on a GitHub repository by Li et al.[16] to reproduce the DLKcat model and to make predictions for their test set. We divided both, the DLKcat test set and our test set, into four different subsets according to the protein sequence identity compared to the amino acid sequences in the training sets. To achieve this, we calculated for every test sequence the maximal pairwise sequence identity compared to all sequences in the training set using the Needleman-Wunsch algorithm from the software package EMBOSS[52]. We used the coefficient of determination R^2 to compare the results of TurNuP with the results of the DLKcat model.

Predicting protein abundances using predicted k_{cat} values

Li et al.[16] developed a Bayesian pipeline to use predicted k_{cat} values for enzyme-constrained genome-scale metabolic models to predict the proteome of yeast species. We used Matlab code provided on a GitHub repository by Li et al.[16] to follow the same pipeline for k_{cat} values predicted with TurNuP. The predicted proteome allocations were compared to measured proteome data for four different species in 21 different cultural conditions. The measured proteome data was taken from six different publications[53–58].

Statistical tests for model comparison

To test if the difference in model performance between the TurNuP model with enzyme and reaction information compared to the models with either only enzyme or reaction information is statistically significant, we applied a one-sided Wilcoxon signed-rank test implemented in the Python package SciPy[59]. We tested the null hypothesis that the median of the absolute errors on the test set for predictions made with TurNuP, $\bar{e}_1$, is greater or equal to the corresponding median for predictions made with a model with only reaction or only enzyme information, $\bar{e}_2$ ($H_0 : \bar{e}_1 \geq \bar{e}_2$ vs. $H_1 : \bar{e}_2 > \bar{e}_1$). We could reject H_0 ($p = 0.0003$ (structural fingerprint), $p = 0.0427$ (difference fingerprint), $p = 0.0046$ (*ESM-1b$_{ESP}$*)), accepting the alternative hypothesis H_1.

We also tested if the differences in model performance between TurNuP and the DLKcat model are statistically significant for all subsets of the test set with different enzyme sequence identity levels. We used the non-parametric one-sided Wilcoxon–Mann–Whitney test implemented in the Python package SciPy[59] to test the null hypothesis that the prediction errors for the two models are equally distributed. We could reject the null hypothesis for three subsets at the 5% level with p-values of $p = 1.5 \times 10^{-10}$ ($0 - 40\%$), $p = 0.0039$ ($40 - 80\%$), and $p = 0.0025$ ($80 - 99\%$), while the p-value for the forth subset ($99 - 100\%$) was slightly above the 5% level ($p = 0.080$).

Calculating reaction fluxes

We calculated reaction fluxes for all 108 genome-scale metabolic models (GEMs) from the BiGG database[39]. We selected those six GEMs for different organisms that showed the highest correlation between calculated fluxes through parsimonious flux balance analyses (pFBA) and k_{cat} values in our dataset. We selected the following six models: iECO111_1330 (*Escherichia coli*), iEK1008 (*Mycobacterium tuberculosis*), iHN637 (*Clostridium ljungdahlii*), iIT341 (*Helicobacter pylori*), iSbBS512_1146 (*Shigella boydii*), and iJN1463 (*Pseudomonas putida*).

We calculated the reaction fluxes similar to the approach by Heckmann et al.[14] for *E. coli*. For each of the six GEMs, we simulated 10 000 minimal growth sustaining environments through pFBA[36] using the Python package COBRApy[60]. Afterwards, we calculated for every reaction the mean of all non-zero fluxes among all simulations. In all of the 10 000 simulations, first a growth sustaining environments was created with a growth rate higher than 0.1 $\left[h^{-1}\right]$ and oxygen uptake was allowed with a probability of 50% for aerobic organisms. To convert the medium into a minimal media, each metabolite of the medium was removed if growth was sustained without it. If we could not obtain a non-zero flux for a reaction in all simulations, we repeated the described procedure with a flux variability analysis (FVA)[61] instead of a pFBA. If we could not obtain a non-zero-flux for a reaction either via pFBA or via FVA, we replaced the reaction flux with the mean of all non-zero fluxes. Python code for calculating the fluxes is available on the following GitHub repository: `https://github.com/Nina181/kcat_flux_relationship`.

Mapping data points to BiGG reaction IDs

We created a list with reactions from six different metabolic genome-scale models from the BiGG database[39] (iECO111_1330, iEK1008, iHN637, iIT341, iSbBS512_1146, iJN1463). To create this list, we downloaded a json-files for each model and we extracted all substrate names and IDs (MetaNetX or KEGG), product names and IDs, and BiGG reaction IDs. We discarded all reactions with an incomplete list of substrate or product IDs. If only a MetaNetX ID and no KEGG ID was available for a metabolite, we downloaded an InChI string[49] for the metabolite using the MetaNetX database[62]. Next, we calculated structural reaction fingerprints for all extracted BiGG reactions using the KEGG IDs and InChI strings of the substrates and products (for details see above, ”Calculating reaction fingerprints”).

To map data points from our data set to BiGG reactions IDs, we calculated a pairwise similarity score between all reactions in our dataset and all reactions from the 6 extracted BiGG models. To calculate the similarity score, we used the Python function *TanimotoSimilarity* from the RDKit package DataStructs[24] with structural reaction fingerprints as the input. This resulted in a similarity score between 0 (no similarity) and 1 (very high similarity) for all pairs of reactions. We mapped every data point in our dataset to the BiGG reaction with the highest similarity score.

Calculating Michaelis constants

To calculate the Michaelis constants K_M for all enzyme catalyzed reactions in our dataset, we created a list with all enzyme-substrate pairs. We used the BRENDA database[20] to map enzyme-substrate pairs to K_M values via the enzymes' amino acid sequences and via a molecular fingerprint of the substrate, called ECFP vector[63]. We were able to map a K_M value to $\sim 7\%$ of 8 984 enzyme-substrate pairs.

If we could not find a value for an enzyme-substrate pair in the BRENDA database, we predicted K_M using a machine learning model[37]. The K_M prediction model uses a graph neural network (GNN)[64,65] to create a 50-dimensional task-specific fingerprint of the substrate. These fingerprints are used together with a 1900-dimensioanl enzyme representation, called UniRep vector[27], as the input for a gradient boosted decision tree model[32] to predict the K_M value for an enzyme-substrate pair. For reactions with multiple substrates, we took the geometric mean of K_M values to create a single K_M value for every data point.

Calculating the Codon Adaptation Index

The codon adaptation index (CAI) for *E. coli* was calculated according to the original definition[66], considering ribosomal protein genes as the highly expressed genes. The sequences of ribosomal protein genes were retrieved from genome annotation of *E. coli* (NC_000913.3 from RefSeq[67]).

Acknowledgements

We thank David Heckmann for helpful discussions. Computational support and infrastructure was provided by the “Centre for Information and Media Technology” (ZIM) at the University of Düsseldorf (Germany). This work was funded by the Deutsche Forschungsgemeinschaft (DFG,German Research Foundation) through CRC 1310, and, under Germany’s Excellence Strategy, through grant EXC 2048/1 (Project ID: 390686111).

Conflict of interest

The authors declare that they have no conflicts of interest.

Author contributions

AK designed the dataset and models and performed all other analyses. NAL implemented the calculation of the reaction fluxes. XPH calculated the Codon Adaptation Index (CAI) for genes from *E. coli*. MJL conceived of and supervised the study, and acquired funding. AK wrote the initial manuscript, which was edited by AK and MJL.

References

1. Sánchez, B. J. *et al.* Improving the phenotype predictions of a yeast genome-scale metabolic model by incorporating enzymatic constraints. *Mol. Syst. Biol.* **13,** 935 (2017).
2. Beg, Q. K. *et al.* Intracellular crowding defines the mode and sequence of substrate uptake by Escherichia coli and constrains its metabolic activity. *PNAS* **104,** 12663–12668 (2007).
3. Lerman, J. A. *et al.* In silico method for modelling metabolism and gene product expression at genome scale. *Nat. Commun.* **3,** 1–10 (2012).
4. O'brien, E. J., Lerman, J. A., Chang, R. L., Hyduke, D. R. & Palsson, B. Ø. Genome-scale models of metabolism and gene expression extend and refine growth phenotype prediction. *Mol. Syst. Biol.* **9,** 693 (2013).
5. Yang, L., Yurkovich, J. T., King, Z. A. & Palsson, B. O. Modeling the multi-scale mechanisms of macromolecular resource allocation. *Curr. Opin. Microbiol.* **45,** 8–15 (2018).
6. Thiele, I. *et al.* Multiscale Modeling of Metabolism and Macromolecular Synthesis in E. coli and Its Application to the Evolution of Codon Usage. *PLOS ONE* **7,** 1–18 (2012).
7. Khodayari, A. & Maranas, C. D. A genome-scale Escherichia coli kinetic metabolic model k-ecoli457 satisfying flux data for multiple mutant strains. *Nat. Commun.* **7,** 1–12 (2016).
8. Ebrahim, A. *et al.* Multi-omic data integration enables discovery of hidden biological regularities. *Nat. Commun.* **7,** 1–9 (2016).
9. Dourado, H. & Lercher, M. J. An analytical theory of balanced cellular growth. *Nat. Commun.* **11,** 1–14 (2020).
10. Dourado, H., Liebermeister, W., Ebenhöh, O. & Lercher, M. J. Growth Mechanics: General principles of optimal cellular resource allocation in balanced growth. *bioRxiv.* doi:10.1101/2022.10.27.514082 (2022).
11. Davidi, D. *et al.* Global characterization of in vivo enzyme catalytic rates and their correspondence to in vitro kcat measurements. *PNAS* **113,** 3401–3406 (2016).
12. Saa, P. A. & Nielsen, L. K. Formulation, construction and analysis of kinetic models of metabolism: A review of modelling frameworks. *Biotechnol. adv.* **35,** 981–1003 (2017).

13. Strutz, J., Martin, J., Greene, J., Broadbelt, L. & Tyo, K. Metabolic kinetic modeling provides insight into complex biological questions, but hurdles remain. *Curr. Opin. Biotechnol* **59,** 24–30 (2019).

14. Heckmann, D. *et al.* Machine learning applied to enzyme turnover numbers reveals protein structural correlates and improves metabolic models. *Nat. Commun.* **9,** 1–10 (2018).

15. Orth, J. D., Thiele, I. & Palsson, B. Ø. What is flux balance analysis? *Nat. Biotechnol.* **28,** 245–248 (2010).

16. Li, F. *et al.* Deep learning-based kcat prediction enables improved enzyme-constrained model reconstruction. *Nat. Catal.,* 1–11 (2022).

17. Rives, A. *et al.* Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *PNAS* **118,** e2016239118 (2021).

18. Rao, R. *et al.* Evaluating protein transfer learning with TAPE. *Adv Neural Inf Process Syst.* **32,** 9686–9698 (2019).

19. Detlefsen, N. S., Hauberg, S. & Boomsma, W. Learning meaningful representations of protein sequences. *Nat. Commun.* **13,** 1–12 (2022).

20. Chang, A. *et al.* BRENDA, the ELIXIR core data resource in 2021: new developments and updates. *Nucleic Acids Res.* **49,** D498–D508 (2021).

21. UniProt Consortium. UniProt: a worldwide hub of protein knowledge. *Nucleic Acids Res.* **47,** D506–D515 (2019).

22. Wittig, U., Rey, M., Weidemann, A., Kania, R. & Müller, W. SABIO-RK: an updated resource for manually curated biochemical reaction kinetics. *Nucleic Acids Res.* **46,** D656–D660 (2018).

23. Bar-Even, A. *et al.* The moderately efficient enzyme: evolutionary and physicochemical trends shaping enzyme parameters. *Biochemistry* **50,** 4402–4410 (2011).

24. Landrum, G. *et al.* Rdkit: Open-source cheminformatics software, 2016. *URL http://www. rdkit. org/, https://github. com/rdkit/rdkit* **149,** 150 (2016).

25. Hu, Q.-N. *et al.* Assignment of EC numbers to enzymatic reactions with reaction difference fingerprints. *PLOS ONE* **7,** 1–6 (2012).

26. Smallbone, K. *et al.* A model of yeast glycolysis based on a consistent kinetic characterisation of all its enzymes. *FEBS Lett.* **587,** 2832–2841 (2013).

27. Alley, E. C., Khimulya, G., Biswas, S., AlQuraishi, M. & Church, G. M. Unified rational protein engineering with sequence-based deep representation learning. *Nat. Methods* **16,** 1315–1322 (2019).

28. Vaswani, A. *et al. Attention is all you need* in *Advances in neural information processing systems* (Curran Associates, Inc., 2017), 5998–6008.

29. Suzek, B. E. *et al.* UniRef clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics* **31,** 926–932 (2015).

30. Kroll, A., Ranjan, S., Engqvist, M. K. M. & Lercher, M. J. The substrate scopes of enzymes: a general prediction model based on machine and deep learning. *bioRxiv.* doi:10.1101/2022.05.24.493213 (2022).

31. Goldman, S., Das, R., Yang, K. K. & Coley, C. W. Machine learning modeling of family wide enzyme-substrate specificity screens. *PLoS Comput. Biol.* **18,** 1–20 (2022).

32. Chen, T. & Guestrin, C. *Xgboost: A scalable tree boosting system* in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (2016), 785–794.

33. Friedman, J. H. *The elements of statistical learning: Data mining, inference, and prediction* (SpringerOpen, 2017).

34. Tian, W. & Skolnick, J. How well is enzyme function conserved as a function of pairwise sequence identity? *J. Mol. Biol.* **333,** 863–882 (2003).

35. Holzhütter, H. G. The principle of flux minimization and its application to estimate stationary fluxes in metabolic networks. *European Journal of Biochemistry* **271,** 2905–2922. doi:10.1111/j.1432-1033.2004.04213.x (2004).

36. Lewis, N. E. *et al.* Omic data from evolved E. coli are consistent with computed optimal growth from genome-scale models. *Mol. Syst. Biol.* **6,** 390 (2010).

37. Kroll, A., Engqvist, M. K. M., Heckmann, D. & Lercher, M. J. Deep learning allows genome-scale prediction of Michaelis constants from structural features. *PLoS Biol.* **19,** 1–21 (2021).

38. Sharp, P. M. & Li, W.-H. The codon adaptation index-a measure of directional synonymous codon usage bias, and its potential applications. *Nucleic Acids Res.* **15,** 1281–1295 (1987).

39. King, Z. A. *et al.* BiGG Models: A platform for integrating, standardizing and sharing genome-scale models. *Nucleic Acids Res.* **44,** D515–D522 (2016).

40. McDonald, A. G. & Tipton, K. F. Parameter Reliability and Understanding Enzyme Function. *Molecules* **27,** 263 (2022).

41. Wilken, S. E. *et al.* Interrogating the effect of enzyme kinetics on metabolism using differentiable constraint-based models. *Metabolic Engineering* **74,** 72–82 (2022).

42. Van Rossum, G. & Drake, F. L. *Python 3 Reference Manual* (CreateSpace, 2009).

43. Paszke, A. *et al.* Pytorch: An imperative style, high-performance deep learning library. *Adv. Neur. In.* **32,** 8026–8037 (2019).

44. Django Software Foundation. *Django* version 2.2. May 5, 2019.

45. Kanehisa, M. & Goto, S. KEGG: kyoto encyclopedia of genes and genomes. *Nucleic Acids Res.* **28,** 27–30 (2000).

46. Kim, S. *et al.* PubChem in 2021: new data content and improved web interfaces. *Nucleic Acids Res.* **49,** D1388–D1395 (2021).

47. Hastings, J. *et al.* ChEBI in 2016: Improved services and an expanding collection of metabolites. *Nucleic Acids Res.* **44,** D1214–D1219 (2016).

48. Cokelaer, T., Pultz, D., Harder, L. M., Serra-Musach, J. & Saez-Rodriguez, J. BioServices: a common Python package to access biological Web Services programmatically. *Bioinformatics* **29,** 3241–3242 (2013).

49. Heller, S., McNaught, A., Stein, S., Tchekhovskoi, D. & Pletnev, I. InChI-the worldwide chemical structure identifier standard. *J. Cheminf.* **5,** 1–9 (2013).

50. Sayle, R. *1st-class SMARTS patterns* in *EuroMUG 97* (1997).

51. Bergstra, J., Yamins, D. & Cox, D. *Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures* in *Proceedings of the 30th International Conference on Machine Learning* (eds Dasgupta, S. & McAllester, D.) **28** (PMLR, Atlanta, Georgia, USA, 17–19 Jun 2013), 115–123.

52. Rice, P., Longden, I. & Bleasby, A. EMBOSS: the European molecular biology open software suite. *Trends Genet.* **16,** 276–277 (2000).

53. Lahtvee, P.-J. *et al.* Absolute quantification of protein and mRNA abundances demonstrate variability in gene-specific translation efficiency in yeast. *Cell Syst.* **4,** 495–504 (2017).

54. Björkeroth, J. *et al.* Proteome reallocation from amino acid biosynthesis to ribosomes enables yeast to grow faster in rich media. *PNAS* **117,** 21804–21812 (2020).

55. Paulo, J. A., O'Connell, J. D., Gaun, A. & Gygi, S. P. Proteome-wide quantitative multiplexed profiling of protein expression: carbon-source dependency in Saccharomyces cerevisiae. *Mol. Biol. Cell* **26,** 4063–4074 (2015).

56. Paulo, J. A. *et al.* Quantitative mass spectrometry-based multiplexing compares the abundance of 5000 S. cerevisiae proteins across 10 carbon sources. *J. Proteomics* **148,** 85–93 (2016).

57. Doughty, T. W. *et al.* Stress-induced expression is enriched for evolutionarily young genes in diverse budding yeasts. *Nat. Commun.* **11,** 1–9 (2020).

58. Kito, K. *et al.* Yeast interspecies comparative proteomics reveals divergence in expression profiles and provides insights into proteome resource allocation and evolutionary roles of gene duplication. *Mol. Cell. Proteomics* **15,** 218–235 (2016).

59. Virtanen, P. *et al.* SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nat. Methods* **17,** 261–272 (2020).

60. Ebrahim, A., Lerman, J. A., Palsson, B. O. & Hyduke, D. R. COBRApy: constraints-based reconstruction and analysis for python. *BMC Syst. Biol.* **7,** 1–6 (2013).

61. Mahadevan, R. & Schilling, C. H. The effects of alternate optimal solutions in constraint-based genome-scale metabolic models. *Metab. Eng.* **5,** 264–276 (2003).

62. Moretti, S., Tran, V. D. T., Mehl, F., Ibberson, M. & Pagni, M. MetaNetX/MNXref: unified namespace for metabolites and biochemical reactions in the context of metabolic models. *Nucleic Acids Res.* **49,** D570–D574 (2021).

63. Rogers, D. & Hahn, M. Extended-connectivity fingerprints. *J. Chem. Inf. Model.* **50,** 742–754 (2010).

64. Zhou, J. *et al.* Graph neural networks: A review of methods and applications. *AI Open* **1,** 57–81 (2020).

65. Yang, K. *et al.* Analyzing learned molecular representations for property prediction. *J. Chem. Inf. Model.* **59,** 3370–3388 (2019).

66. Sharp, P. M. & Li, W.-H. The codon adaptation index-a measure of directional synonymous codon usage bias, and its potential applications. *Nucleic Acids Res.* **15,** 1281–1295 (1987).

67. O'Leary, N. A. *et al.* Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation. *Nucleic Acids Res.* **44,** D733–D745 (2016).

Supporting Figures S1-S4

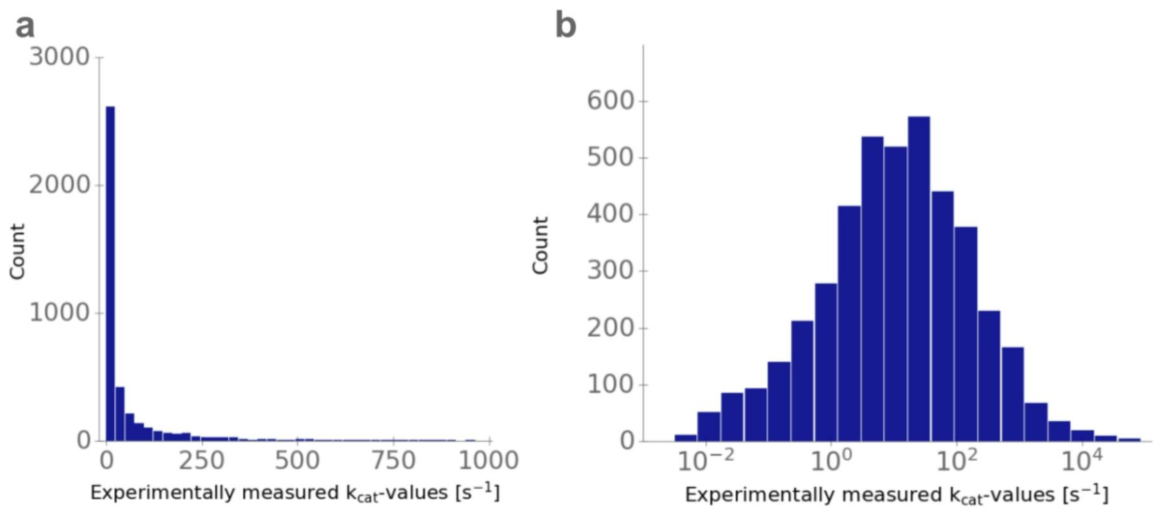


Figure S1. Turnover numbers are approximately log-normally distributed. Histograms of **(a)** untransformed k_{cat} values and **(b)** log_{10}-transformed k_{cat} values.

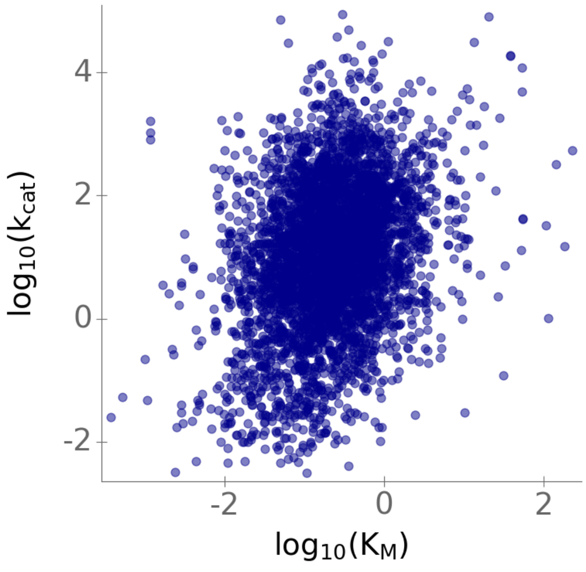


Figure S2. Michaelis constants K_M compared to k_{cat} values. We obtained either experimentally measured K_M values from BRENDA or predicted K_M values for every reaction and we plotted these values against the corresponding k_{cat} values on a $\log_{10}$-scale. The plot contains 4271 data points from our training and test set.

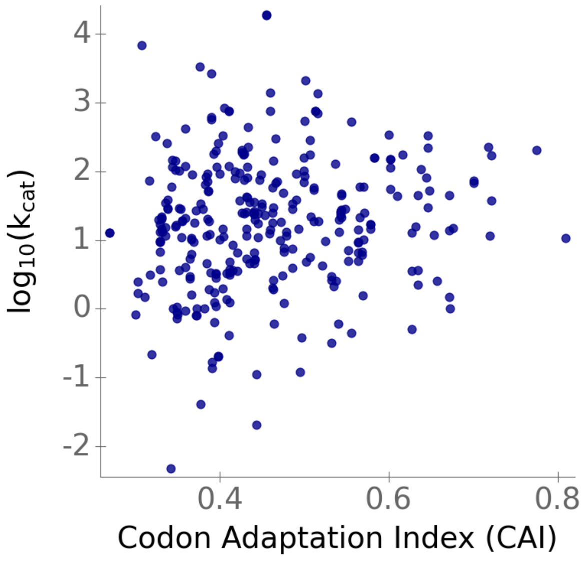


Figure S3. Codon Adaptation Index (CAI) for enzymes from *E. coli* compared to their k_{cat} values. We calculated the CAI for genes from *E. coli* and plotted it against their $\log_{10}$-transformed k_{cat} value. The plot contains 303 data points from our training and test set.

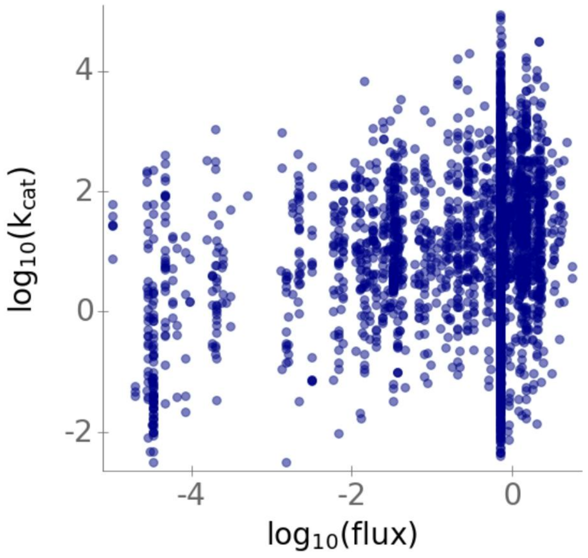


Figure S4. Predicted reaction fluxes compared to k_{cat} values. We obtained either predicted reaction fluxes via parsimonious flux balance analysis (pFBA) or flux variability analysis (FVA) and we plotted these values against the corresponding k_{cat} values on a $\log_{10}$-scale. The plot contains 4271 data points from our training and test set.

3.3 MANUSCRIPT 3

Manuscript 3 is a revised version of the paper that is published on bioRxiv [76]. The manuscript is currently under revision at Nature Communications.

Contributions to Manuscript 3

I designed the study together with Martin Lercher and Martin Engqvist. Martin Lercher and Martin Engqvist defined the overall task and suggested additional analyses for the trained enzyme-substrate pair prediction model. I created the dataset with experimentally validated enzyme-substrate pairs, and I designed the model for creating task-specific enzyme representations, the *ESM-1b*$_{\mathrm{ts}}$ model, which was trained and implemented by Sahasra Ranjan. I developed, implemented, and trained all other machine learning models, including the graph neural network for creating task-specific small molecule representations and the gradient boosting model for final model training. Moreover, I developed the procedure for sampling negative data points from unlabeled data. I performed all data curation, analyses, visualizations, and I implemented a web server that allows an easy use of the prediction model. I wrote the original draft of the manuscript. The manuscript was reviewed and edited by Martin Lercher, Martin Engqvist, and me.

The substrate scopes of enzymes: a general prediction model based on machine and deep learning

Alexander Kroll[1], **Sahasra Ranjan**[3], **Martin K. M. Engqvist**[2,4], **Martin J. Lercher**[1,*]

[1] Institute for Computer Science and Department of Biology, Heinrich Heine University, D-40225 Düsseldorf, Germany

[2] Department of Biology and Bioengineering, Chalmers University of Technology, SE-412 96 Gothenburg, Sweden

[3] Department of Computer Science and Engineering, Indian Institute of Technology Bombay, Powai, Mumbai 400076, India

[4] Current address: EnginZyme AB, Tomtebodevägen 6, 17165 Stockholm, Sweden

[*] Correspondence to: Martin.Lercher@hhu.de

ABSTRACT

For a comprehensive understanding of metabolism, it is necessary to know all potential substrates for each enzyme encoded in an organism's genome. However, for most proteins annotated as enzymes, it is unknown which primary and/or secondary reactions they catalyze, as experimental characterizations are time-consuming and costly. Machine learning predictions could provide an efficient alternative, but are hampered by a lack of information regarding enzyme non-substrates, as available training data comprises mainly positive examples. Here, we present ESP, a general machine learning model for the prediction of enzyme-substrate pairs, with an accuracy of over 91% on independent and diverse test data. This accuracy was achieved by representing enzymes through a modified transformer model with a trained, task-specific token, and by augmenting the positive training data by randomly sampling small molecules and assigning them as non-substrates. ESP can be applied successfully across widely different enzymes and a broad range of metabolites included in the training data. It outperforms recently published models designed for individual, well-studied enzyme families, which use much more detailed input data. We implemented a user-friendly web server to predict the substrate scope of arbitrary enzymes, which may support not only basic science, but also the development of pharmaceuticals and bioengineering processes.

Introduction

Enzymes evolved to efficiently catalyze one or more specific chemical reactions, increasing reaction rates up to over a million-fold over the spontaneous rates[1]. In addition, most enzymes are promiscuous, i.e., they catalyze further, physiologically irrelevant or even harmful reactions[2–4]. Accordingly, a comprehensive mapping of enzyme-substrate relationships plays a crucial role in pharmaceutical research and bio-engineering, e.g., for the production of drugs, chemicals, food, and biofuels[5–7].

Unfortunately, it is both expensive and time-consuming to determine experimentally which reactions are catalyzed by a given enzyme. There is thus a huge imbalance between the number of proteins predicted to be enzymes and the experimental knowledge about their substrate scopes. While the UniProt database[8] contains entries for over 36 million different enzymes, more than 99% of these lack high-quality annotations of the catalyzed reactions. Efforts are underway to develop high-throughput methods for the experimental determination of enzyme-substrate relationships, but these are still in their infancy[9–11]. Furthermore, even high-throughput methods cannot deal with the vast search space of all possible small molecule substrates, but require the experimenter to choose a small subset for testing.

Our goal in this study was to develop a single machine learning model capable of predicting enzyme-substrate relationships across all proteins, thereby providing a tool that helps to focus experimental efforts on enzyme-small molecule pairs likely to be biologically relevant. Developing such a model faces two major challenges. First, a numerical representation of each enzyme that is maximally informative for the downstream prediction task must be obtained[12]. To be as broadly applicable as possible, these representations should be based solely on the enzymes' primary sequence and not require additional features, such as binding site characteristics. Second, public enzyme databases only list positive instances, i.e., molecules with which enzymes display measurable activity (substrates)[13]. For training a prediction model, an automated strategy for obtaining suitable negative, non-binding enzyme-small molecule instances must thus be devised.

Existing machine learning approaches for predicting enzyme-substrate pairs were either developed specifically for small enzyme families for which unusually comprehensive training datasets are available[13–17], or they are only capable of connecting substrates with EC classes but not with specific enzymes. For example, Mou et al.[14] developed models to predict the substrates of bacterial nitrilases, using input features based on the 3D-structures and active sites of the enzymes. They trained various machine learning models based on experimental evidence for all possible enzyme-small molecule combinations within the models' prediction scope ($N = 240$). Yang et al.[15] followed a similar approach, predicting the substrate scope of plant glycosyltransferases among a pre-defined set of small molecules. They trained a decision tree-based model with a dataset covering almost all possible combinations of enzymes and relevant small molecules. Pertusi et al.[13] trained four different support vectors machines (SVMs), each for a specific enzyme. As input features, their models only use information about the (potential) substrates, as well as non-substrates manually extracted from the literature; no explicit information about the enzymes was used. Roettig et al.[16] and Chevrette et al.[17] predicted the substrate scopes of small enzyme families, training machine learning models with structural information relating to the enzymes' active sites. Finally, Visani et al.[18] implemented a general machine learning model for predicting suitable EC classes for a given substrate. To train this model, all EC classes that are not associated with a certain substrate were used as negative data points, which resulted in a low average positive to negative ratio of 0.0032. Visani et al. did not use any enzyme information beyond the EC class as model input, and therefore the model cannot distinguish between different enzymes assigned to the same EC class.

All these previous models can either not be applied to individual enzymes, or they aim to predict substrates for only a single enzyme or enzyme family. Those models that make predictions for specific enzymes rely on very dense experimental training data, i.e., experimental results for all or almost all potential enzyme-substrate pairs. However, for the vast majority of enzyme families, such extensive training data is not available. As yet, there have been no published attempts to formulate and train a general model that can be applied to predict substrates for specific enzymes across widely different enzyme families. Deep learning models have been used to predict enzyme functions by either predicting their assignment to EC classes[19–21], or by predicting functional domains within the protein sequence[22]. However, different enzymes sharing the same domain architecture or assigned to the same EC class can have highly diverse substrate scopes[23]. Directly predicting specific substrates for enzymes goes an important step beyond those previous methods and can help to predict enzyme function more specifically and more precisely.

Prior work related to the prediction of enzyme-substrate pairs are the prediction of drug-target binding affinities (DTBAs) and of Michaelis-Menten constants, K_{M} and k_{cat}. State-of-the-art approaches in this domain are feature-based, i.e., numerical representations of the protein and the substrate molecule are used as input to machine learning models[24–28]. As numerical descriptions of the substrate molecule, these approaches use SMILES representations[29], expert-crafted fingerprints[30], or fingerprints created with graph neural networks[31,32]. Proteins are usually encoded numerically through deep learning-based representations of the amino acid sequences[33–35]. However, these approaches cannot be transferred one-to-one to the problem of predicting enzyme-substrate pairs. The K_M and k_{cat} prediction models are exclusively trained with positive enzyme-substrate pairs and therefore cannot classify molecules as substrates or non-substrates[27,28]. Many of the proteins used to train the DTBA prediction models have no enzymatic functions; even if they do, the molecules used for training are mostly not naturally occurring potential substrates, and thus there has been no natural selection for or against binding. In contrast, the binding between enzymes and substrates evolved under natural selection. It appears likely that this evolutionary relationship influences our ability to predict enzyme-substrate pairs, and DTBA models are thus not expected to perform well at this task.

Here, we go beyond the current state-of-the-art by creating maximally informative protein representations, using a customized, task-specific version of the *ESM-1b* transformer model[33]. The model contains an extra 1 280-dimensional token, which was trained end-to-end to store enzyme-related information salient to the downstream prediction task. This general approach was first introduced for natural language processing[36], but has not yet been applied to protein feature prediction. We created negative training examples using data augmentation, by randomly sampling small molecules similar to the substrates in experimentally confirmed enzyme-substrate pairs. Importantly, we sampled all negative data points from a limited set of metabolites, the set of $\sim$ 1 400 substrates that occur among all experimentally confirmed enzyme-substrate pairs of our dataset. Thus, we do not sample from the space of all possible alternative reactants similar to the true substrates, but only consider small molecules likely to occur in at least some biological cells. While many enzymes are rather promiscuous[2–4], it is likely that most of the potential secondary substrates are not contained in this restricted set for any given enzyme, and hence the chance of sampling false negative data points was likely small. We numerically represented all small molecules with task-specific fingerprints that we created with graph neural networks (GNNs)[37–39]. A gradient-boosted decision tree model was trained on the combined protein and small molecule representations for a high-quality dataset with $\sim$ 18 000 very diverse, experimentally confirmed positive enzyme-substrate pairs (**Figure 1**).

The resulting Enzyme Substrate Prediction model – ESP – achieves high prediction accuracy for those $\sim$ 1 400 substrates that have been part of our training set and outperforms previously published enzyme family-specific prediction models. Thus, our work demonstrates how augmented datasets and enzyme representations re-trained for a specific task can be used to overcome challenges in predicting enzyme-substrate relationships. While model performance decreases when ESP is applied to small

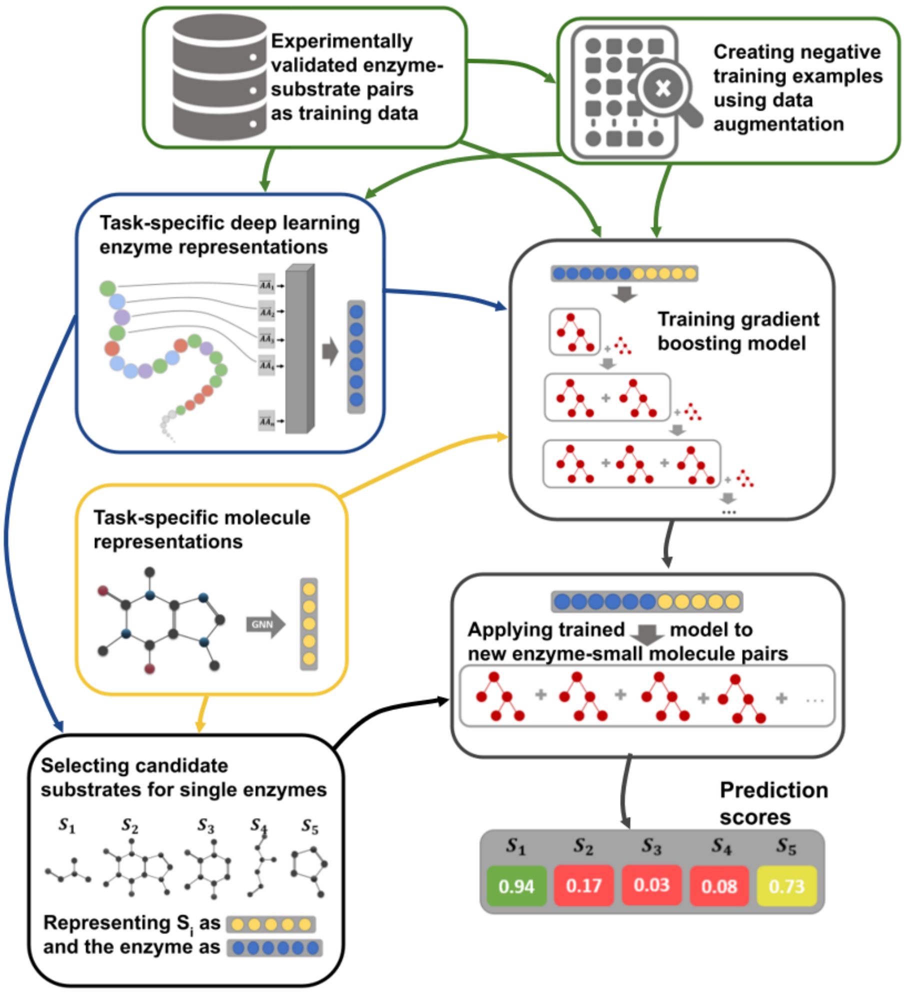

Experimentally validated enzyme-substrate pairs as training data
Creating negative training examples using data augmentation
Task-specific deep learning enzyme representations
Training gradient boosting model
Task-specific molecule representations
GNN
Applying trained model to new enzyme-small molecule pairs
Selecting candidate substrates for single enzymes
S_1 S_2 S_3 S_4 S_5
Representing S_i as
and the enzyme as
Prediction scores
S_1 0.94
S_2 0.17
S_3 0.03
S_4 0.08
S_5 0.73

Figure 1. Model overview. Experimentally validated enzyme-substrate pairs and sampled negative enzyme-small metabolite pairs are numerically represented with task-specific enzyme and small molecule representations. Concatenated enzyme-small molecule representations are used to train a gradient boosting model. After training, the fitted model can be used to predict promising candidate substrates for enzymes.

molecules that have not been part of the training set, many use cases will aim to connect known substrates to unknown enzymes. For example, a specific user might be interested in finding new enzymes that bind to substrates involved in the metabolism of malate. To identify promising enzyme-substrate pairs for further experimental study, ESP could be applied to the metabolites involved in malate metabolism in combination with enzymes with incomplete functional characterizations.

Results

Obtaining training and test data

We created a dataset with experimentally confirmed enzyme-substrate pairs using the GO annotation database for UniProt IDs[40] (Methods, “Creating a database with enzyme-substrate pairs”). For training our machine learning models, we extracted 18 351 enzyme-substrate pairs with experimental evidence for binding, comprised of 12 156 unique enzymes and 1 379 unique metabolites. We also extracted 274 030 enzyme-substrate pairs with phylogenetically inferred evidence, i.e., these enzymes are evolutionarily closely related to enzymes associated with the same reactions. These ”guilt by association” assignments are much less reliable than direct experimental evidence, and we only used them during pre-training to create task-specific enzyme representations – numerical vectors aimed at capturing information relevant to the prediction task from the enzyme amino acid sequences. Our validations demonstrate that using phylogenetically inferred functions for the construction of appropriate enzyme representations has a positive effect on the prediction of experimentally confirmed enzyme-substrate pairs (see below, ”Representing enzymes through a modified state-of-the-art deep learning architecture”).

There is no systematic information on negative enzyme-small molecule pairs, i.e., pairs where the molecule is not a substrate of the enzyme. We hypothesized that such negative data points could be created artificially through random sampling, which is a common strategy in classification tasks that lack negative training data[41]. To challenge our model to learn to distinguish similar binding and non-binding reactants, we sampled negative training data only from enzyme-small molecule pairs where the small molecule is structurally similar to a known true substrate. However, we only considered small molecules included among the experimentally confirmed enzyme-substrate pairs in our dataset. Among such a limited and biased subset, enzymes are quite specific catalysts, and therefore most of the potential secondary substrates are not included for the majority of enzymes. Thus, we assumed that the frequency of incorrectly created negative labels is sufficiently low to not adversely affect model performance. This assumption was confirmed by the high model accuracy on independent test data, as detailed below.

To select putatively non-binding small molecules that are structurally similar to the known substrates, we used a similarity score based on molecular fingerprints, with values ranging from 0 (no similarity) to 1 (identity; see Methods, “Sampling negative data points”). For every positive enzyme-substrate pair, we sampled three molecules with similarity scores between 0.75 and 0.95 to the actual substrate of the enzyme, and used them to construct negative enzyme-molecule pairs. We opted for creating more negative data points than we have positive data points, as this not only provided us with more data, but it also more closely reflects the true distribution of positive and negative data points compared to a balanced distribution.

Our final dataset comprises 69 365 entries. We split this data into a training set (80%) and a test set (20%). In many machine learning domains, it is standard practice to split the data into training and test set completely at random. However, when dealing with protein sequences, this strategy often leads to test sets with amino acid sequences that are almost identical to those of proteins in the training set. Such close homologs often share the same function[42], and the assessment of model performance could thus be overly optimistic. It is therefore common practice to split such datasets into training, validation, and test sets based on protein sequences similarities[43]. Here, we made sure that no enzyme in the test set has a sequence identity higher than 80% compared to any enzyme in the training set. To show that despite this sequence-based partitioning, enzymes from the training and test sets follow the same distribution, we used dimensionality reduction to map all enzymes to a two-dimensional subspace and plotted the corresponding data points (**Supplementary Fig. 1**). To evaluate how well our final model performs for different levels of enzyme similarities, we divided the test set further into three subsets with maximal sequence identities between 0-40%, 40-60%, and 60-80% compared to all enzymes in the training set.

Representing small molecules as numerical vectors

Extended-connectivity fingerprints (ECFPs) are expert-crafted binary representations for small molecules. The molecules are represented as graphs, with atoms interpreted as nodes and chemical bonds as edges. For the numerical encoding, one classifies bond types and calculates feature vectors with information about every atom (types, masses, valences, atomic numbers, atom charges, and number of attached hydrogen atoms)[30]. Afterwards, these identifiers are updated for a fixed number of steps by iteratively applying predefined functions to summarize aspects of neighboring atoms and bonds. After the iteration process, all identifiers are converted into a single binary vector with structural information about the molecule. The number of iterations and the dimension of the fingerprint can be chosen freely. We set them to the default values of 3 and 1 024, respectively. For comparison, we also created 512- and 2 048-dimensional ECFPs, but these led to slightly inferior predictions (**Supplementary Fig. 2**). Using ECFPs can lead to identical representations for structurally very similar molecules, e.g., for some molecules that differ only by the length of a chain of carbon atoms. In our dataset, 182 out of 1 379 different molecules shared an identical fingerprint with a structurally similar molecule.

As an alternative to expert-crafted fingerprints such as ECFPs, neural networks can be used to learn how to map graph representations of small molecules to numerical vectors. Such networks are referred to as graph neural networks (GNNs)[37–39]. We trained a GNN for the binary task of predicting if a small molecule is a substrate for a given enzyme. While training for this task, the GNN is challenged to store all information about the small molecule that is relevant for solving the prediction task in a single numerical vector. After training, we extracted these 100-dimensional task-specific vectors for all small molecules in our dataset. It has been observed that pre-training GNNs for a related task can significantly improve model performance[44,45]. Thus, we first pre-trained a GNN for the related task of predicting the Michaelis constants K_M of enzyme-substrate pairs (see Methods, “Calculating task-specific fingerprints for the small molecules using graph neural networks”). As shown below (see “Successful prediction of enzyme-substrate pairs by using combined enzyme and small molecule representations”), pre-training indeed improved prediction performance significantly. In contrast to ECFPs, GNN-generated fingerprints lead to much fewer cases of identical representations for different molecules. In our dataset, identical fingerprints occurred for 42 out of 1 379 molecules.

Representing enzymes through a modified state-of-the-art deep learning architecture

The *ESM-1b* model is a state-of-the-art transformer network[46], trained with ∼27 million proteins from the UniRef50 dataset[47] in a self-supervised fashion[33]. This model takes an amino acid sequence as its input and puts out a numerical representation of the sequence; these representations are often referred to as protein embeddings. During training of *ESM-1b*, ∼ 15% of the amino acids in a protein’s sequence are randomly masked and the model is trained to predict the identity of the masked amino acids (**Figure 2a**). This training procedure forces the model to store both local and global information about the protein sequence in one 1 280-dimensional representation vector for each individual amino acid. In order to create a single fixed-length numerical representation of the whole protein, one typically calculates the element-wise mean across all amino acid representations[33,34,48]. We refer to these protein representations as *ESM-1b* vectors.

However, simply taking the element-wise mean results in information loss and does not consider the task for which the representations shall be used, which can lead to subpar performance[12]. To overcome these issues, we created task-specific enzyme representations optimized for the prediction of enzyme-substrate pairs. We slightly modified the architecture of the *ESM-1b* model, adding one additional 1 280-dimensional token to represent the complete enzyme, intended to capture information salient to the

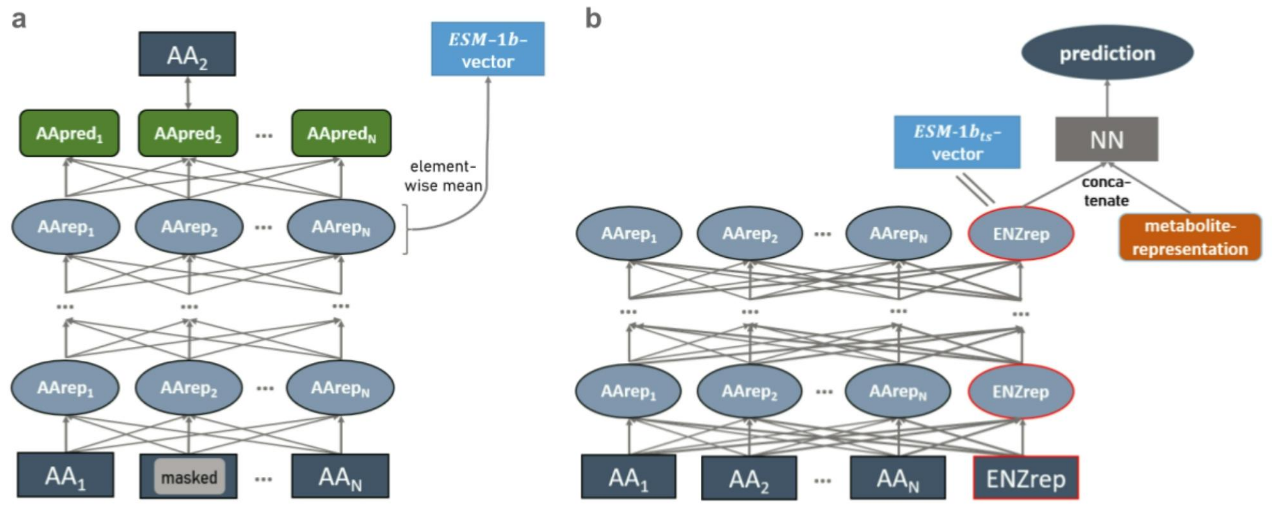


Figure 2. A task-specific enzyme representation developed from the *ESM-1b* model. (a) *ESM-1b* model. Amino acids of a protein sequence are represented with numerical vectors and passed through a transformer network. Some amino acid representations are masked. All representations are iteratively updated 33 times, using information about neighboring and distant amino acids. The *ESM-1b* model is trained to predict the masked amino acids. *ESM-1b* vectors are calculated by taking the element-wise mean of all representations in the last layer. **(b)** Modified *ESM-1b* model. An additional representation for the whole enzyme is added to the amino acid representations. After updating all representations 33 times, the enzyme representation is concatenated with a small molecule representation. The network is trained to predict whether the small molecule is a substrate for the given enzyme. After training, the *ESM-1b*$_{ts}$ vector is extracted as the enzyme representation before adding the small molecule representation.

downstream prediction task (**Figure 2b**). This whole-enzyme representation was updated in the same way as the regular *ESM-1b* amino acid representations.

After a predefined number of update steps, the enzyme representation was concatenated with the small molecule ECFP-vector. The combined vector was used as the input for a fully connected neural network (FCNN), which was then trained end-to-end to predict whether the small molecule is a substrate for the enzyme. This approach facilitates the construction of a single, optimized, task-specific representation. The *ESM-1b* model contains many parameters and thus requires substantial training data. Therefore, in the pre-training that produces the task-specific enzyme representations, we added phylogenetically inferred evidence to our training set; this resulted in a total of $\sim 287\,000$ data points used for training the task-specific enzyme representation. After training, we used the network to extract the 1 280-dimensional

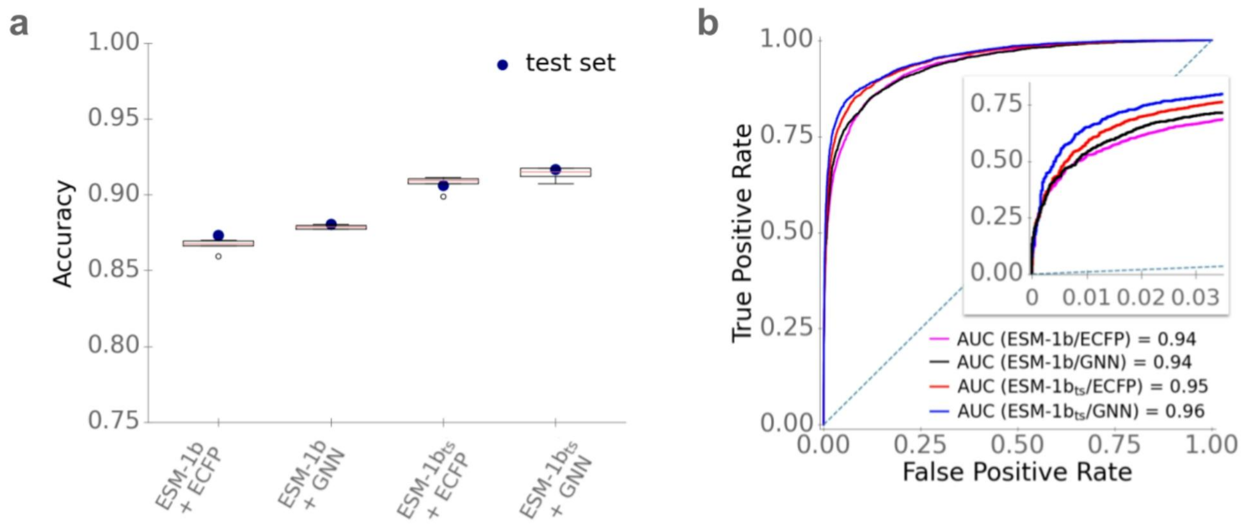


Figure 3. Optimized models provide accurate predictions of enzyme-substrate pairs. (a) Accuracies. Boxplots summarize the results of the 5-fold CV on the training set with the best sets of hyperparameters. Blue dots display the accuracies on the test set, using the optimized models trained on the whole training set. **(b)** ROC curves for the test set. The dotted line displays the ROC curve expected for a completely random model.

task-specific representations for all enzymes in our dataset. In the following, these representations are called *ESM-1b$_{ts}$* vectors.

Successful prediction of enzyme-substrate pairs by using combined enzyme and small molecule representations

To compare the performances of the different enzyme representations (*ESM-1b* and *ESM-1b$_{ts}$* vectors) and of the two small molecule representations (ECFPs and GNN-generated fingerprints), we estimated prediction quality on our test set when using machine learning models with each of the four combinations of enzyme and small molecule representations. In each case, we concatenated one of the two 1 280-dimensional enzyme representations with one of the two small molecule representations to create a single input vector for every enzyme-small molecule pair. We used these inputs to train gradient boosted decision tree models[49] for the binary classification task of predicting whether the small molecule is a substrate for the enzyme.

We performed hyperparameter optimizations for all four models, including the parameters learning rate, depth of trees, number of iterations, and regularization coefficients. For this, we performed a random grid search with a 5-fold cross-validation (CV) on the training set. To challenge the model to learn to predict the substrate scope of enzymes not included in the training data, we made sure that each enzyme occurred in only one of the five subsets used for cross-validation (Methods, "Hyperparameter optimization of the gradient boosting models"). To account for the higher number of negative compared to positive training data, we also included a weight parameter that lowered the influence of the negative data points. The results of the cross-validations are displayed as boxplots in **Figure 3a**. The best sets of hyperparameters are listed in **Supplementary Table 1.** After hyperparameter optimization, the models were trained with the best set of hyperparameters on the whole training set and were validated on our independent test set, which had not been used for model training or hyperparameter selection. It is noteworthy that for some input combinations, the accuracies on the test set are higher than the accuracies achieved during cross-validation (**Figure 3a**). This improved performance on the test set may result from the fact that before validation on the test set, models are trained with approximately 11 000 more samples than before each cross-validation; the number of training samples has a substantial influence on model performance (see below, "Model performance increases with increased training set size").

Commonly used metrics to measure the performance of binary classification models are accuracy, ROC-AUC score, and Matthews correlation coefficient (MCC). Accuracy is simply the fraction of correctly predicted data points among the test data. The ROC-AUC score is a value between 0 and 1 that summarizes how well a classifier is able to distinguish between the positive and negative classes, where a value of 0.5 would result from a model that randomly assigns class labels, and a value of 1 corresponds to perfect predictions. The MCC is a correlation coefficient for binary data, comparable to the Pearson correlation coefficient for continuous data; it takes values between -1 and +1, where 0 would result from a model that randomly assigns class labels, and +1 indicates perfect agreement.

As shown in **Figure 3** and **Table 1**, models with task-specific enzyme and/or small molecule representations performed better than those with generic representations. The best-performing model combined the fine-tuned $ESM\text{-}1b_{ts}$ enzyme representations with the GNN-generated small molecule fingerprints, achieving an accuracy of 91.5%, a ROC-AUC score of 0.956, and an MCC of 0.78. The difference between the two best models ($ESM\text{-}1b_{ts}$ + GNN vs. $ESM\text{-}1b_{ts}$ + ECFP) is statistically highly significant (McNemar's test: $p < 10^{-5}$). For the final ESP model, we thus chose to represent enzymes with $ESM\text{-}1b_{ts}$ vectors and small molecules with GNN-generated, task-specific fingerprints.

To compare the gradient boosting model to alternative machine learning models, we also trained a logistic regression model and a random forest model for the task of predicting enzyme-substrate pairs

Table 1. Prediction performance on the test set for all four combinations of enzyme and small molecule representations.

	ROC-AUC score	**Accuracy**	**MCC**
ESM-1b + ECFP	0.937	87.2%	0.69
ESM-1b$_{ts}$ + ECFP	0.950	90.5%	0.75
ESM-1b + GNN	0.940	88.8%	0.72
ESM-1b$_{ts}$ + GNN	0.956	91.5%	0.78

from the combined *ESM-1b$_{ts}$* and GNN vectors. However, these models performed worse compared to the gradient boosting model (**Supplementary Table 2**).

The GNN used to represent small molecules in the best-performing model was pre-trained for the task of predicting the Michaelis constants K_M of enzyme-substrate pairs. To test if this pre-training improved the predictions, we also tested model performance for fingerprints that were created with a GNN that was not pre-trained. Using a pre-trained GNN indeed led to better model performance (**Supplementary Table 3**; $p < 10^{-7}$ from McNemar's test).

The results summarized in **Table 1** demonstrate that re-training and fine-tuning the *ESM-1b* model can significantly improve model performance. This finding contrasts previous observations that fine-tuning protein representations can negatively influence model performance and can lead to worse results compared to using the original *ESM-1b* model[12,50]. To achieve the improved enzyme representations, we added an extra token for the whole enzyme, and we trained the model to store all relevant information for the prediction task in this token. To investigate the importance of the added token for the observed superior performance, we alternatively re-trained the *ESM-1b* without such an extra token. Our results show that using the extra token indeed improves model performance (**Supplementary Table 4**; $p = 0.040$ from McNemar's test).

Good predictions even for enzymes with low sequence identity to training data

It appears likely that prediction quality is best for enzymes that are highly similar to enzymes in the training set, and decreases for enzymes that are increasingly dissimilar to the enzymes used for training. How strong is that dependence? To answer this question, we first calculated the maximal enzyme sequence identity compared to the enzymes in the training set for all 2 291 enzymes in the test set. Next, we split the

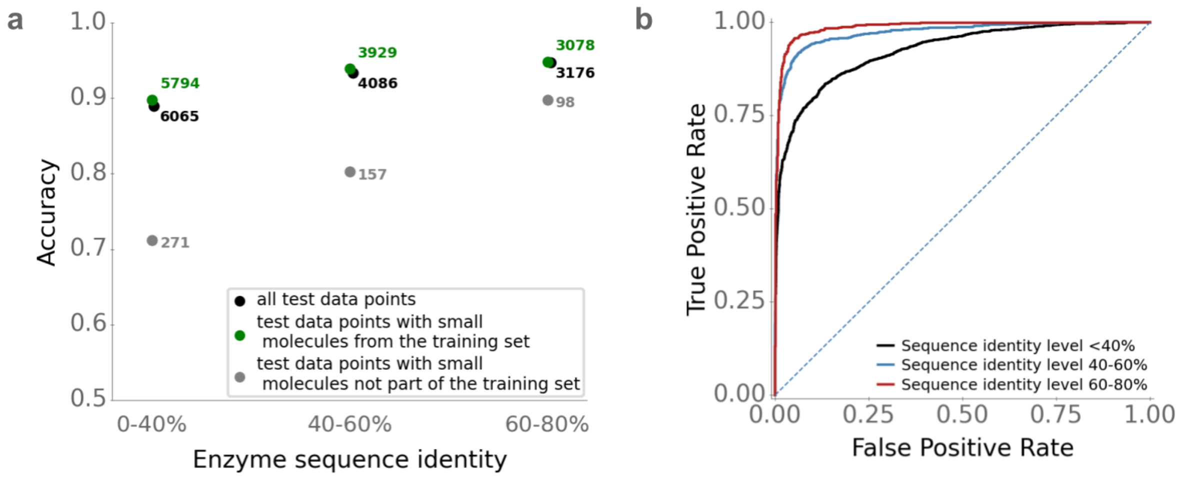


Figure 4. Accurate predictions even for enzymes with distinct sequence similarity compared to enzymes in the training data. We divided the test set into subsets with different levels of enzyme sequence identity compared to enzymes in the training set. **(a)** ESP accuracies, calculated separately for enzyme-small molecule pairs where the small molecule occurred in the training set and where it did not occur in the training set. **(b)** ESP ROC curves. The dotted line displays the ROC curve expected for a completely random model.

test set into three subgroups: data points with enzymes with a maximal sequence identity to training data between 0 and 40%, between 40% and 60%, and between 60% and 80%.

For data points with high sequence identity levels (60-80%), the ESP model is highly accurate, with an accuracy of 95%, ROC-AUC score of 0.99, and MCC of 0.88 (**Figure 4**). ESP still performs very well for data points with intermediate sequence identity levels (40-60%), achieving an accuracy of 93%, ROC-AUC score 0.97, and MCC 0.83. Even for enzymes with low sequence identity to training data ($0-40\%$), the ESP model achieves good results and classifies 89% of the data points correctly, with ROC-AUC score 0.93 and MCC 0.72. Thus, while using more similar enzymes during training improves the prediction quality, very good prediction accuracy can still be achieved for enzymes that are only distantly related to those in the training set. The observed differences were statistically significant for sequence identities 0-40% versus 40-60% (Mann–Whitney U test: $p < 10^{-23}$), but not for 40-60% versus 60-80% ($p = 0.14$).

Low model performance for unseen small molecules

In the previous subsection, we showed that model performance is highest for enzymes that are similar to proteins in the training set. Similarly, it appears likely that the model performs better when making predictions for small molecules that are also in the training set. To test this hypothesis, we divided the test set into data points with small molecules that occurred in the training set ($N = 13\,459$) and those with small molecules that did not occur in the training set ($N = 530$).

The ESP model does not perform well for data points with small molecules not present in the training set. When considering only enzyme-small molecules pairs with small molecules not represented in the training set and an enzyme sequence identity level of 0-40% compared to the training data, ESP achieves an accuracy of 71%, ROC-AUC score 0.59, and MCC 0.15. At an enzyme sequence identity level of 40-60%, accuracy improves to 83%, with ROC-AUC score 0.78, and MCC 0.25 for unseen small molecules. At high enzyme sequence identity levels of 60-80%, the accuracy reaches 90%, with ROC-AUC score 0.71, and MCC 0.27. Thus, for unseen small molecules, even a very moderate model performance requires that proteins similar to the enzyme ($> 40\%$ identity) are present in the training set. We again found the differences to be statistically significant for 0-40% versus 40-60% (Mann–Whitney U test: $p < 10^{-20}$), but not for 40-60% versus 60-80% ($p = 0.226$).

For those test data points with small molecules not present in the training set, we wondered if a high similarity of the small molecule compared to at least one substrate in the training set leads to improved predictions, analogous to what we observed for enzymes with higher sequence identities. For each small molecules not present in the training set, we calculated the maximal pairwise similarity score compared to all substrates in the training set. We could not find any evidence that a higher maximal similarity score leads to better model performance (**Supplementary Fig. 3**). Hence, we conclude that ESP only achieves high accuracies for new enzyme-small molecule pairs if the small molecule was present among the $\sim$1 400 substrates of our training set.

How many training data points with identical substrates are needed to achieve good model performance? For every small molecule in the test set, we counted how many times the same molecule occurs as an experimentally confirmed substrate in the training set. **Supplementary Fig. 4** shows that having as few as two positive training data points for a given small molecule leads to good accuracy when pairing the same small molecule with other enzymes.

Model performance increases with increased training set size

The previous subsections suggest that a bigger training set with a more diverse set of enzymes and small molecules should lead to improved performance. However, using more data does not guarantee an improved model performance. For example, there could be a limitation in the model architecture

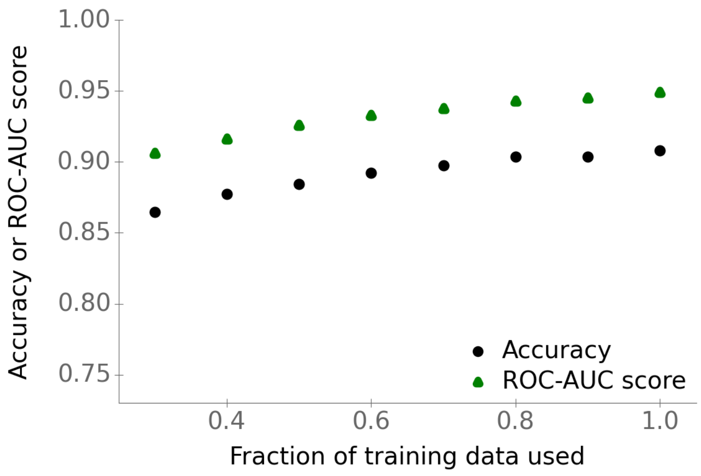


Figure 5. Model performance increases with training set size. Points show accuracies and ROC-AUC scores for the test set versus the fraction of the available training data used for training the gradient boosting model.

that prevents the model from better fitting the data. To test how our model performs with different amounts of training data and to analyze if more data is expected to lead to higher generalizability, we trained the gradient boosting model with different training set sizes, ranging from 30% to 100% of the available training data. **Figure** 5 shows that accuracy and ROC-AUC score indeed increase with increasing training set size (Spearman rank correlations, accuracy: $\rho^2 = 0.95$, $p < 10^{-4}$; ROC-AUC score: $\rho^2 = 1.0$, $p < 10^{-15}$). Thus, collecting more and more diverse data – for example, through targeted additional experiments – will likely lead to further model improvements.

ESP can express uncertainty for data points with low prediction accuracy

Internally, our trained classification model does not simply output the positive or negative class as a prediction. Instead, it outputs a prediction score between 0 and 1, which can be interpreted as a measurement of the probability for a data point to belong to the positive class. So far, we assigned all predictions with a score ≥ 0.5 to the positive class, and all predictions below 0.5 to the negative class. To provide a more detailed view of prediction accuracies, **Figure** 6 displays the distributions of the true (blue) and false (red) predictions for our test set across prediction scores.

Most true predictions have a score either close to 0 or close to 1, i.e., the ESP model is very confident about these predictions. In contrast, false predictions are distributed much more evenly across prediction scores. Approximately 4% of prediction scores for our test data fall between 0.4 and 0.6. The model

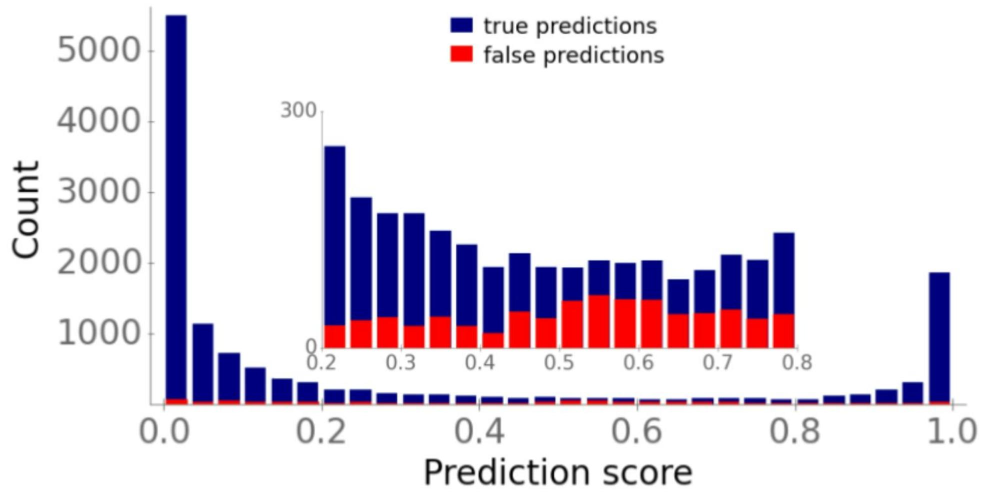


Figure 6. Prediction scores around 0.5 indicate model uncertainty. Stacked histogram bars display the prediction score distributions of true predictions (blue) and false predictions (red). The inset shows a blow-up of the interval $[0.2, 0.8]$.

seems to be uncertain for these data points: for this subset, predictions are only barely better than random guesses, with an accuracy of 59%, ROC-AUC score 0.60, and MCC 0.17 (**Figure** 6, inset). Thus, when applied in practice, prediction scores between 0.4 and 0.6 should be considered uncertain and should not be assigned to one of the two classes.

ESP outperforms two recently published models for predicting the substrate scope of enzymes

We compared ESP with two recently published models for predicting the substrate scopes of specific enzyme families. ESP has been trained with much more data points compared to the previously published models; conversely, these previous models used much more detailed input information. Thus, a fair, direct comparison of model architectures is impossible. Instead, we analyzed if our model, which is capable of making use of large amounts of freely available data, can lead to better prediction accuracies than much more targeted approaches that necessarily work on smaller datasets.

Mou et al.[14] trained four different machine learning models (logistic regression, random forest, gradient-boosted decision trees, and support vector machines) to predict substrates of bacterial nitrilases. For model training and validation, they used a dataset with all possible combinations of 12 enzymes and 20 small molecules ($N = 240$), randomly split into 80% training data and 20% test data. We added all training data from Ref.[14] to our training set and validated the updated ESP model on the corresponding test data, which had no overlap with our training data. Mou et al.[14] achieved an accuracy of 82% and a ROC-AUC score of 0.90 on the test set. ESP achieves better results, with an accuracy of 87.5%, ROC-AUC score 0.94, and MCC 0.75. This improvement is particularly striking given that Mou et al.[14] used knowledge about the enzymes' 3D structures and binding sites, while we only use a representation of the linear amino acid sequences.

Yang et al.[15] published a decision tree-based model, GT-Predict, for predicting the substrate scope of glycosyltransferases of plants. As a training set, they used 2 847 data points with 59 different small molecules and 53 different enzymes from *Arabidopsis thaliana*, i.e., the data covered 90.7% of all possible enzyme-small molecule combinations. These authors used two independent test sets to validate the model, one dataset with 266 data points with enzymes from *Avena strigosa* and another dataset with 380 data points with enzymes from *Lycium barbarum*. On those two test sets, GT-Predict achieves accuracies of 79.0% and 78.8%, respectively, and MCCs of 0.338 and 0.319, respectively. We added the training set from Ref.[15] to our training set. The test sets from *Avena strigosa* and *Lycium barbarum* had no overlap with our training data. For these two sets, we achieved similar accuracies as Yang et al. (78.2% in both cases), but substanitally improved MCCs: 0.484 for *Avena strigosa* and and 0.517 for *Lycium barbarum* (ROC-AUC scores were 0.80 and 0.84, respectively). As the test datasets used by Yang et al.[15] are imbalanced, with a proportion of 18-31% of positive data points, the MCC is a more meaningful score compared to the accuracy[51]; we hence conclude that ESP outperforms GT-Predict. Beyond benchmarking the performance of ESP, the above comparisons of our model predictions to two (almost) complete experimental datasets also indicate that ESP is indeed capable of predicting the full substrate scope of enzymes.

We also tested model performances for the test sets by Mou et al.[14] and Yang et al.[15] without adding any new training data to ESP. Only $\sim 5\%$ and $\sim 8\%$ of the small molecules in these test sets did already occur in our training set. As we showed above that performance drops massively if the model is applied to unseen small molecules (**Figure 4a**), we did not expect good model performances. Indeed, for all three test sets, accuracies are below 68%, ROC-AUC scores are below 0.59, and MCCs are below 0.12 (**Supplementary Table 5**).

The ESP web server facilitates an easy use of the prediction model

We implemented a web server that allows an easy use of ESP without requiring programming skills or the installation of specialized software. It is available at `https://esp.cs.hhu.de`. As input, the web server requires an enzyme amino acid sequence and a representation of a small molecule (either as a

SMILES string, KEGG Compound ID, or InChI string). Users can either enter a single enzyme-small molecule pair into an online form, or upload a CSV file with multiple such pairs. In addition to the prediction score, the ESP web server reports how often the entered metabolite was present as a true substrate in our training set. Since we have shown that model performance drops substantially when the model is applied to small molecules not used during training, we recommend to use the prediction tool only for those small molecules represented in our training dataset. We uploaded a full list with all small molecules from the training set to the web server homepage, listing how often each one is present among the positive data points.

Discussion

We presented a general approach for predicting the substrate scope of enzymes; ESP achieves an accuracy of over 91% on an independent test set with enzymes that share at most 80% sequence identity with any enzyme used for training. Notably, the model performs with an accuracy of 89% even for enzymes with very low sequence identity ($< 40\%$) to proteins in the training set. This performance seems remarkable, as it is believed that enzymes often evolve different substrate specificities or even different functions if sequence identity falls below 40%[42].

To achieve these results, we used very general input features: a task-specific fingerprint of the small molecule, constructed with a graph neural network (GNN) from a graph representing structural information, and a numerical representation of the enzyme calculated from its amino acid sequence. We showed that creating task-specific enzyme representations leads to significant improvements compared to non-task-specific enzyme representations (**Figure 3**). Moreover, our results clearly show that a carefully devised strategy of randomly sampling negative enzyme-molecule pairs can be an effective and viable approach. Future refinements of this approach might boost model performance further. For example, when creating negative data points for confirmed enzyme-substrate pairs, a tighter decision boundary might result from preferentially choosing structurally similar substrates of highly different enzymes. On the other hand, the sets of true substrates of highly similar enzymes often overlap, and excluding known substrates of highly similar enzymes could avoid creating some false negative data points.

An additional avenue towards potential model improvements could be to test new model architectures. In this study, we trained two separate models for creating task-specific enzyme and small molecule representations. Future work could investigate if the pre-training of the enzyme representation and the small molecule representation could be performed jointly in a single model, thereby creating matched, task-specific enzyme and small molecule representations simultaneously.

Despite the structural similarities of ESP to state-of-the-art models for predicting drug–target binding affinities (DTBAs) and for predicting Michaelis-Menten constants of enzyme-substrate pairs[24–28], the performances of these models are not comparable, as we trained ESP for a binary classification task, whereas the other models address regression tasks. Instead, we compared our approach to two recently published models for predicting enzyme-substrate pairs[14,15]. These two models used very specific input features, such as an enzyme's active site properties and physicochemical properties of the metabolite, and were designed and trained for only a single enzyme family. Our general ESP model – which can be trained on much larger datasets – achieves superior results, despite learning and extracting all relevant information for this task from much less detailed, general input representations. The application of ESP to the dataset from Mou et al.[14] also demonstrated that our model can successfully distinguish between similar potential substrates for the same enzyme, as it achieved good results when it was applied to different nitriles for bacterial nitrilases.

One limitation of ESP is that model performance drops substantially for small molecules that did not occur in the training set. However, the current version of ESP can still be applied successfully to a broad range of almost 1 400 different small molecules present in our dataset. Once more training data becomes available, model performance will very likely improve further (**Figure 5**). Mining other biochemical databases – such as BRENDA[52], Sabio-RK[53], and UniProt[8] – for new and non-overlapping data might be a low-cost way to expand the number of different small molecules in the dataset. Adding as few as two additional positive training data points for new molecules will typically lead to accurate predictions (**Supplementary Fig. 4**).

The recent development of AlphaFold[54] and RoseTTAFold[55] facilitates predictions of the 3D structure for any protein with known amino acid sequence. Future work may also include input features extracted from such predicted enzyme structures. Our high-quality dataset with many positive and negative enzyme-small metabolite pairs, which is available on GitHub, might be a promising starting point to explore the utility of such features.

A main use case for the ESP model will be the prediction of possible substrate candidates for single enzymes. In contrast, ESP will likely not lead to satisfactory results when used to predict all enzyme-substrate pairs in a genome scale metabolic model. This problem results from the trade-off between the True Positive Rate (TPR) and the False Postive Rate (FPR) for different classification thresholds (**Figure 3b**). For example, choosing a classification threshold with a TPR of $\sim 80\%$ leads to a FPR of $\sim 5\%$. If we consider a genome scale model with approximately 2 000 enzymes and 2 000 metabolites, then there exist $\sim 4 \times 10^6$ possible enzyme-small molecule pairs, of which only about 6 000 will be true enzyme-substrate pairs. A TPR of 80% would lead to the successful detection of 4 800 true pairs. At the same time, an FPR of 5% would lead to an additional $\sim$ 200 000 false predictions.

If, on the other hand, ESP is applied to a set of pre-selected candidate substrates for a single enzyme, a false positive rate of 5% can be acceptable. If we choose 200 molecules as substrate candidates, where one of these 200 is a true substrate for the enzyme, an FPR of 5 % means that the model predicts only $\sim$ 10 molecules falsely as a substrate, and there is an 80% chance that the true substrate is labeled correctly. This could help to bring down the experimental burden – and associated costs – of biochemical assays to levels where laboratory tests become tractable.

Methods

Software

All software was coded in Python[56]. We implemented and trained the neural networks using the deep learning library PyTorch[57]. We fitted the gradient boosting models using the library XGBoost[49].

Creating a database with enzyme-substrate pairs

To create a database with positive enzyme-substrate pairs, we searched the Gene Ontology (GO) annotation database for UniProt IDs[40] for experimentally confirmed annotations of the catalytic activity of enzymes. A GO annotation consists of a GO Term that is assigned to a UniProt ID, which is an identifier for proteins. GO Terms can contain information about the biological processes, molecular functions, and cellular components in which proteins are involved[58]. We first created a list with all 6 587 catalytic GO Terms containing information about enzyme-catalyzed reactions. For each of these GO Terms, we extracted identifiers for the substrates involved in the reaction. If the GO Term definition stated that the reaction is reversible, we treated all reactants (including products) as substrates; if a reaction was labeled as irreversible, we only extracted the reactants annotated as substrates. For this purpose, we used a RHEA reaction ID[59] from the GO Term, which was available for 4 086 out of 6 587 GO Terms. If no RHEA reaction ID was listed for the GO Term, we extracted the substrate names via text mining from the GO Term definition. Substrate names were then mapped to KEGG and ChEBI identifiers via the synonym database from KEGG[60], or, if no entry in KEGG was found, the PubChem synonym database[61]. We discarded all 824 catalytic GO Terms for which we could not map at least one substrate to an identifier.

Entries in the GO annotation database have different levels of evidence: experimental, phylogenetically-inferred, computational analysis, author statement, curator statement, and electronic evidence. For training our final model, we were interested only in in entries with catalytic GO Terms based on experimental evidence. From these, we removed 6 219 enzyme-substrate pairs with water, oxygen, and ions, as these small substrates did not lead to unique representations (see below). We extracted protein and substrate IDs for the remaining 18 351 enzyme-substrate pairs with experimental evidence. 15 051 of these pairs resulted from a GO Term that was associated with a RHEA reaction ID, the rest were created via text mining of GO Term definitions. These data points are combinations of 12 156 unique enzymes and 1 379 unique substrates.

Before training our models for predicting enzyme-substrate pairs, we pre-trained the *ESM-1b* protein representations to capture information relevant to enzyme-substrate binding. Due to the high dimensionality of the protein representations, much more data than the 18 351 enzyme-substrate pairs with experimental evidence was required for this task. Only for this pre-training, we thus additionally extracted protein and substrate IDs for 274 030 entries with catalytic GO Terms and phylogenetically inferred evidence (this set excludes 98 384 entries with water, oxygen, and ions as substrates). 200 634 of these enzyme-substrate pairs resulted from a GO Term associated with a RHEA reaction ID, the rest were constructed via text mining of GO Term definitions. These additional data points based on phylogenetic evidence are combinations of 198 259 unique enzymes and 661 unique substrates.

It might be surprising that although we found many more enzyme-substrate pairs with phylogenetically inferred evidence compared to data points with experimental evidence, the number of unique substrates is much smaller. To investigate if we can see a systematic difference between both groups, we plotted the distribution of the first digit of EC classes among the enzymes of both classes. However, no substantial difference was evident except for an over-representation of EC6 (ligases) in the data with phylogenetic evidence (**Supplementary Fig. 5**). Hence, we assume that the data structure of phylogenetically inferred data points is not an important issue for the calculation of enzyme representations.

We downloaded all enzyme amino acid sequences via the UniProt mapping service[8].

Sampling negative data points

For every positive enzyme-substrate pair in our dataset, we created three negative data points for the same enzyme by randomly sampling small molecules. The distinction between true and false substrates is harder for small molecules that are similar to the true, known substrates. To challenge our model to learn this distinction, we restricted our sampling of negative data points to small molecules similar to the true substrate. For this purpose, we first calculated the pairwise similarity of all small molecules in our dataset with the function FingerprintSimilarity from the RDKit package DataStructs[62]. This function uses molecular fingerprints of the molecules as its input and computes values between zero (no similarity) and one (high similarity). If possible, we sampled small molecules with a similarity score between 0.7 and 0.95. If we did not find such molecules, we reduced the lower bound in steps of 0.2 until enough small molecules could be sampled. We had to reduce the lower bound in $\sim 19\%$ of enzyme-substrate pairs. We did not simply choose the three most similar compounds as negative data points, because if a substrate appears multiple times in our dataset, this would have led to selecting always the same three small molecules as non-substrates. Instead, we randomly picked three molecules from within the selected similarity range. During this sampling process, we took the distribution of the small molecules among the positive data points into account, i.e., molecules that occur more frequently as substrates among the positive data points also appear more frequently among the negative data points. To achieve this, we excluded small molecules from the sampling process if these molecules were already sampled enough times (i.e., three times their total occurrence in the set of positive enzyme-substrate pairs).

Splitting the dataset into training and test sets

Before we split the dataset into training and test sets, we clustered all sequences by amino acid sequence identity using the CD-HIT algorithm[63]. The clusters were created in such a way that two sequences from different clusters do not have a pairwise sequence identity higher than 80%. We used these clusters to split the dataset randomly into 80% training data and 20% test data using a sequence identity cutoff of 80%, i.e., every enzyme in the test set has a maximal sequence identity of 80% compared to any enzyme in the training set. This was achieved by placing all sequences from one cluster either into the training or the test set. To analyze the ESP performance for different sequence identity levels, we further split the test set into subsets with maximal sequence identity to enzymes in the training set of 0-40%, 40-60%, and 60-80% using the CD-HIT algorithm[63].

Calculating extended-connectivity fingerprints for the small molecules

All small molecules in our final datasets were either assigned to a KEGG ID or ChEBI (Chemical Entities of Biological Interest) ID. For all small molecules with a KEGG ID, we downloaded an MDL Molfile with 2D projections of its atoms and bonds from KEGG[60]. If no MDL Molfile could be obtained in this way, we instead downloaded the International Chemical Idenitifier (InChI) string via the mapping service of MetaCyc[64], if a ChEBI ID was available. We then used the package Chem from RDKit[62] with the MDL Molfiles or InChI strings as the input to calculate the 1 024-dimensional binary ECFPs[30] with a radius (number of iterations) of 3. We also calculated 512 and 2 048-dimensional ECFPs to investigate if these lead to better model performance than 1 024-dimensional ECFPs.

Calculating task-specific fingerprints for the small molecules using graph neural networks

In addition to the pre-defined ECFPs, we also used a graph neural network (GNN) to calculate task-specific numerical representations for the small molecules. GNNs are neural networks that can take graphs as their input[37–39]. A molecule can be represented as a graph by interpreting the atoms and bonds of the molecule as nodes and edges, respectively.

We trained and implemented a variant of GNNs called Directed Message Passing Neural Network (D-MPNN)[32], using the Python package PyTorch[57]. To provide the GNN with information about the small molecules, we calculated feature vectors for every bond and every atom in all molecules[27]. For every atom, these features comprise the atomic number, number of bonds, charge, number of hydrogen bonds, mass, aromaticity, hybridization type, and chirality; for every bond, these features comprise bond type, part of ring, stereo configuration, and aromaticity. To input this information into a GNN, the graphs and the feature vectors are encoded with tensors and matrices. While a graph is processed by a GNN, all atom feature vectors are iteratively updated for a pre-defined number of steps by using information of neighboring bond and atom feature vectors. Afterwards, all atom feature vectors are pooled together by applying the element-wise mean to obtain a single graph representation. The dimension D of the updated atom feature vectors and of the final graph representation can be freely chosen; we chose $D = 100$.

This small molecule representation was then concatenated with a small representation of an enzyme; we chose to use a small enzyme representation instead of the full *ESM-1b* vector to keep the input dimension of the machine learning model used for learning the task-specific small molecule representation low. To compute the small enzyme representation, we performed principal component analysis (PCA)[65] on the *ESM-1b* vectors (see below) and selected the first 50 principal components. The concatenated enzyme-small molecule vector was used as the input for a fully connected neural network (FCNN) with two hidden layers of size 100 and 32, which was trained for predicting whether the small molecule is a substrate for the enzyme. We trained the whole model (the GNN including the FCNN) end-to-end. Thereby, the model was challenged to store task-specific and meaningful information in the graph representations. After training, we extracted a graph representation for every small molecule in our training set, which was then used as input for the complete enzyme-substrate pair prediction model. For more details regarding training and implementation, see our GitHub repository.

We performed a pre-training of the described GNN by training it for the related task of predicting the Michaelis constants K_M of enzyme-substrate pairs. As for the task of identifying potential enzyme-substrate pairs, the prediction of K_M is dependent on the interaction between enzymes and small molecules, and hence, this pre-training task challenged the GNN to learn interactions between an enzyme and a substrate. To train the model for the K_M prediction, we used a dataset that was previously constructed for a K_M prediction model[27]. After pre-training, we fine-tuned the GNN by training it for the task of predicting enzyme-substrate pairs, i.e., we used all parameters that were learned during the pre-training task as initial parameters for the GNN that was fine-tuned.

Calculating enzyme representations

We used the *ESM-1b* model[33] to calculate 1 280-dimensional numerical representations of the enzymes. The *ESM-1b* model is a transformer network[46] that takes amino acid sequences as its input and produces numerical representations of the sequences. First, every amino acid in a sequence is converted into a 1 280-dimensional representation, which encodes the type of the amino acid and its position in the sequence. Afterwards, every representation is updated iteratively for 33 update steps by using information about the representation itself as well as about all other representations of the sequence using the attention mechanism[66]. The attention mechanism allows the model to selectively focus only on relevant amino acid representations to make updates[66]. During training, $\sim 15\%$ of the amino acids in a sequence are masked at random, and the model is trained to predict the type of the masked amino acids. The *ESM-1b* model has been trained with ~ 27 million proteins from the UniRef50 dataset[47]. To create a single representation for the whole enzyme, *ESM-1b* calculates the element-wise mean of all updated amino acids representations in a sequence[33]. We created these representations for all enzymes in our dataset using the code and the trained *ESM-1b* model provided by the Facebook AI Research team on GitHub.

Modifying the *ESM-1b* model to create task-specific enzyme representations

To create task-specific enzyme representations for our task of predicting enzyme-substrate pairs, we modified the *ESM-1b* model. For every input sequence, in addition to the representations of all individual amino acids, we added a token that represents the whole enzyme. This enzyme representation is updated in the same way as the amino acid representations. The parameters of this modified model are initialized with the parameters of the trained *ESM-1b* model, setting the additional enzyme token initially to the element-wise mean of the amino acid representations. After the last update layer of the model, i.e., after 33 update steps, we take the 1 280-dimensional representation of the whole enzyme and concatenate it with a representation for a metabolite, the 1 024-dimensional ECFP vector (see above).

This concatenated vector is then used as the input for a fully-connected neural network (FCNN) with two hidden layers of size 256 and 32. The whole model was trained end-to-end for the binary classification task of predicting whether the added metabolite is a substrate for the given enzyme. This training procedure challenged the model to store all necessary enzyme information for the prediction task in the enzyme representation. After training the modified model, we extracted the updated and task-specific representations, the $ESM\text{-}1b_{ts}$ vectors, for all enzymes in our dataset.

We implemented and trained this model using the Python package PyTorch[57]. We trained the model with the extended dataset of 287 386 enzyme-substrate pairs with phylogenetically inferred or experimental evidence for 2 epochs on 6 NVIDA DGX A100s, each with 40GB RAM. Training the model for more epochs did not lead to improved results. Because of the immense computational power and long training times, it was not possible to perform a systematic hyperparameter optimization. We chose hyperparameters after trying a few selected hyperparameter settings with values similar to the ones that were used for training the original *ESM-1b* model.

Fine-tuning the *ESM-1b* model without an additional enzyme token

To investigate the effect on model performance of adding a token for the whole enzyme to the *ESM-1b* model, we also re-trained the model without such an extra token. Instead, we calculated the element-wise mean of all amino acid representations after the last update layer of the model, as is done in the original *ESM-1b* model. We concatenated the resulting 1280-dimensional vector with a representation for a metabolite, the 1 024-dimensional ECFP vector. As for the model described above, this concatenated vector is then used as the input for a fully-connected neural network (FCNN) with two hidden layers of size 256 and 32. The whole model was trained end-to-end for the binary classification task of predicting whether the added metabolite is a substrate for the given enzyme. The training procedure of this model was identical to the model with an additional token for the whole enzyme (see above).

Hyperparameter optimization of the gradient boosting models

To find the best hyperparameters for the gradient boosting models, we performed 5-fold cross-validations (CVs). To ensure a high diversity between all folds, we created the five folds in such a way that the same enzyme would not occur in two different folds. We used the Python package hyperopt[67] to perform a random grid search for the following hyperparameters: learning rate, maximum tree depth, lambda and alpha coefficients for regularization, maximum delta step, minimum child weight, number of training epochs, and weight for negative data points. The last hyperparameter was added because our dataset is imbalanced; this parameter allows the model to assign a lower weight to the negative data points during training. To ensure that our model is indeed not assigning too many samples to the over-represented negative class, we used a custom loss function that contains the False Negative Rate, FNR, and the False Positive Rate, FPR. Our loss function, $2 \times FNR^2 + FPR^{1.3}$, penalizes data points that are mistakenly assigned to the negative class stronger than data points that are mistakenly assigned to the positive class. After hyperparameter optimization, we chose the set of hyperparameters with the lowest mean loss during CV. We used the python package xgboost[49] for training the gradient boosting models.

Displaying the results of cross-validations with boxplots

We used boxplots to display the results of the 5-fold cross-validations, which we performed to find the best set of hyperparameters. We used a $2\times$ interquartile range for the whiskers, the boxes extend from the lower to upper quartile values, and the red horizontal lines are displaying the median of the data points.

Training of additional machine learning models

To compare the performance of the gradient boosting model to additional machine learning models, we also trained a logistic regression model and a random forest model for the same prediction task. To find the best hyperparameters for the models, we again performed 5-fold CVs on the training set. For the random forest model, the hyperparameter optimized was the number of estimators, and for the logistic regression model we searched for the best penalty function and coefficient of regularization strength. We used the python package scikit-learn[68] for training both models.

Validating our model on two additional test sets

We compared the performance of ESP with two published models for predicting the substrate scope of single enzyme families. One of these models is a machine learning model developed by Mou et al. to predict the substrates of 12 different bacterial nitrilases[14]. Their dataset consists of 240 data points, where each of the 12 nitriliases was tested with the same 20 small molecules. This dataset was randomly split by Mou et al. into 80% training data and 20 % test data[14]. We added all training data to our training set. After re-training, we validated our model performance on the test set from Ref.[14].

The second model that we compared to ESP is a decision tree-based model, called GT-predict, for predicting the substrate scope of glycosyltransferases of plants[15]. As a training set, Yang et al.[15] used 2 847 data points with 59 different small molecules and 53 different enzymes from *Arabidopsis thaliana*. They used two independent test sets to validate model performance: one dataset with 266 data points comprising 7 enzymes from *Avena strigose* and 38 different small molecules, and a second dataset with 380 data points comprising 10 enzymes from *Lycium barbarum* and 38 different small molecules. We added all training data to our training set. After re-training, we validated ESP model performance on both test sets from Ref.[15].

Analyzing the effect of training set size

To analyze the effect of different training set sizes, we created eight different subsets of our training set, with sizes ranging from 30% to 100% of the original training set size. To create these subsets, we first generated an enzyme list containing all enzymes of the training set in random order. To create the subsets, we extracted all training data points with enzymes that occur in the first 30%, 40%, ..., 100% of the generated enzyme list. Afterwards, we re-trained our model on all different subsets of the training set and validated each version on our full test set.

Statistical tests for model comparison

We tested if the difference in model performance between the two models with *ESM-1b*$_{ts}$ and ECFP vectors compared to the model with *ESM-1b*$_{ts}$ vectors and GNN-generated fingerprints is statistically significant. For this purpose, we used McNemar's test[69] (implemented in the Python package Statsmodels[70]), testing the null hypothesis that both models have a similar proportion of errors on our test set. We could reject the null hypothesis ($p < 10^{-9}$), concluding that combining *ESM-1b*$_{ts}$ vectors with GNN-generated fingerprints leads to a statistically significant improvement over a combination with ECFP vectors. We performed the same test to show that a model with fingerprints created with a pre-trained GNN achieves improved results compared to a model with fingerprints created with a not pre-trained GNN ($p < 10^{-7}$). Moreover, we used McNemar's test to show that the model with *ESM-1b*$_{ts}$ vectors and GNN-generated fingerprints achieves significantly improved performance compared to the model with *ESM-1b* and ECFP vectors as the input ($p < 10^{-37}$) and also compared to the model with *ESM-1b* and GNN-generated fingerprints ($p < 10^{-19}$). Furthermore, we used the same test to show that the task-specific enzyme representations, the *ESM-1b*$_{ts}$ vectors, that were created by fine-tuning the *ESM-1b* model with an extra token for the whole enzyme achieved improved performance compared to task-specific enzyme representations that resulted from fine-tuning the *ESM-1b* model without such an extra token ($p = 0.040$).

We also tested if the differences in model performance between the three different splits of our test set with different enzyme sequence identity levels (0-40%, 40-60%, and 60-80%) are statistically significant. Here, we used the non-parametric two-sided Mann–Whitney *U* test implemented in the Python package SciPy[71] to test the null hypothesis that the prediction errors for the different splits are equally distributed.

Acknowledgements

We thank Veronica Maurino for insightful discussions. Computational support and infrastructure was provided by the "Centre for Information and Media Technology" (ZIM) at the University of Düsseldorf (Germany). We acknowledge financial support to MJL by the German Research Foundation (DFG) through CRC 1310, and, under Germany's Excellence Strategy, through EXC2048/1 (Project ID:390686111), as well as through a grant by the Volkswagen Foundation under the ”Life” initiative.

Conflict of interest

The authors declare that they have no conflicts of interest.

Author contributions

SR implemented and trained the task-specific *ESM-1b*$_{ts}$ model. AK designed the dataset and models and performed all other analyses. MKME conceived of the study. MJL supervised the study and acquired funding. AK, MKME, and MJL interpreted the results and wrote the manuscript.

Data availability.

All datasets that were created and used to produce the results of this study are publicly available only at `https://github.com/AlexanderKroll/ESP`.

Code availability.

The Python code used to generate all results is publicly available only at `https://github.com/AlexanderKroll/ESP`.

References

1. Cooper, G. M., Hausman, R. E. & Hausman, R. E. *The cell: a molecular approach* (ASM press Washington, DC, 2007).
2. Copley, S. D. Shining a light on enzyme promiscuity. *Curr. Opin. Struct. Biol.* **47,** 167–175 (2017).
3. Tawfik, O. K. & S, D. Enzyme promiscuity: a mechanistic and evolutionary perspective. *Annu. Rev. Biochem.* **79,** 471–505 (2010).
4. Nobeli, I., Favia, A. D. & Thornton, J. M. Protein promiscuity and its implications for biotechnology. *Nat. Biotechnol.* **27,** 157–167 (2009).
5. Adrio, J. L. & Demain, A. L. Microbial enzymes: tools for biotechnological processes. *Biomolecules* **4,** 117–139 (2014).
6. Wang, S. *et al.* Engineering a Synthetic Pathway for Gentisate in Pseudomonas Chlororaphis P3. *Front. Bioeng. Biotechnol.* **8,** 1588 (2021).
7. Wu, M.-C., Law, B., Wilkinson, B. & Micklefield, J. Bioengineering natural product biosynthetic pathways for therapeutic applications. *Curr. Opin. Biotechnol.* **23,** 931–940 (2012).
8. UniProt: the universal protein knowledgebase in 2021. *Nucleic Acids Res.* **49,** D480–D489 (2021).
9. Rembeza, E., Boverio, A., Fraaije, M. W. & Engqvist, M. K. Discovery of Two Novel Oxidases Using a High-Throughput Activity Screen. *ChemBioChem* **23,** e202100510 (2022).
10. Longwell, C. K., Labanieh, L. & Cochran, J. R. High-throughput screening technologies for enzyme engineering. *Curr. Opin. Biotechnol.* **48,** 196–202 (2017).
11. Black, G. W. *et al.* A high-throughput screening method for determining the substrate scope of nitrilases. *Chem. Commun.* **51,** 2660–2662 (2015).
12. Detlefsen, N. S., Hauberg, S. & Boomsma, W. Learning meaningful representations of protein sequences. *Nat. Commun.* **13,** 1914 (Apr. 2022).
13. Pertusi, D. A. *et al.* Predicting novel substrates for enzymes with minimal experimental effort with active learning. *Metab. Eng.* **44,** 171–181 (2017).
14. Mou, Z. *et al.* Machine learning-based prediction of enzyme substrate scope: Application to bacterial nitrilases. *Proteins Struct. Funct. Bioinf* **89,** 336–347 (2021).

15. Yang, M. *et al.* Functional and informatics analysis enables glycosyltransferase activity prediction. *Nat. Chem. Biol.* **14,** 1109–1117 (2018).

16. Röttig, M., Rausch, C. & Kohlbacher, O. Combining structure and sequence information allows automated prediction of substrate specificities within enzyme families. *PLoS Comput. Biol.* **6,** e1000636 (2010).

17. Chevrette, M. G., Aicheler, F., Kohlbacher, O., Currie, C. R. & Medema, M. H. SANDPUMA: ensemble predictions of nonribosomal peptide chemistry reveal biosynthetic diversity across Actinobacteria. *Bioinformatics* **33,** 3202–3210 (2017).

18. Visani, G. M., Hughes, M. C. & Hassoun, S. Enzyme promiscuity prediction using hierarchy-informed multi-label classification. *Bioinformatics* **37,** 2017–2024 (2021).

19. Ryu, J. Y., Kim, H. U. & Lee, S. Y. Deep learning enables high-quality and high-throughput prediction of enzyme commission numbers. *PNAS* **116,** 13996–14001 (2019).

20. Li, Y. *et al.* DEEPre: sequence-based enzyme EC number prediction by deep learning. *Bioinformatics* **34,** 760–769 (Oct. 2017).

21. Sanderson, T., Bileschi, M. L., Belanger, D. & Colwell, L. J. ProteInfer: deep networks for protein functional inference. *bioRxiv* (2021).

22. Bileschi, M. L. *et al.* Using deep learning to annotate the protein universe. *Nat. Biotechnol.* (Feb. 2022).

23. Rembeza, E. & Engqvist, M. K. Experimental investigation of enzyme functional annotations reveals extensive annotation error. *bioRxiv* (2020).

24. Öztürk, H., Özgür, A. & Ozkirimli, E. DeepDTA: deep drug–target binding affinity prediction. *Bioinformatics* **34,** i821–i829 (2018).

25. Feng, Q., Dueva, E., Cherkasov, A. & Ester, M. Padme: A deep learning-based framework for drug-target interaction prediction. *arXiv* (2018).

26. Karimi, M., Wu, D., Wang, Z. & Shen, Y. DeepAffinity: interpretable deep learning of compound–protein affinity through unified recurrent and convolutional neural networks. *Bioinformatics* **35,** 3329–3338 (2019).

27. Kroll, A., Engqvist, M. K., Heckmann, D. & Lercher, M. J. Deep learning allows genome-scale prediction of Michaelis constants from structural features. *PLoS Biol.* **19,** e3001402 (2021).

28. Li, F. *et al.* Deep learning based kcat prediction enables improved enzyme constrained model reconstruction. *bioRxiv* (2021).

29. Weininger, D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.* **28,** 31–36 (1988).

30. Rogers, D. & Hahn, M. Extended-connectivity fingerprints. *J. Chem. Inf. Model.* **50,** 742–754 (2010).

31. Zhou, J. *et al.* Graph neural networks: A review of methods and applications. *AI Open* **1,** 57–81 (2020).

32. Yang, K. *et al.* Analyzing learned molecular representations for property prediction. *J. Chem. Inf. Model.* **59,** 3370–3388 (2019).

33. Rives, A. *et al.* Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *PNAS* **118,** 622226 (2021).

34. Alley, E. C., Khimulya, G., Biswas, S., AlQuraishi, M. & Church, G. M. Unified rational protein engineering with sequence-based deep representation learning. *Nat. Methods* **16,** 1315–1322 (2019).

35. Xu, Y. *et al.* Deep dive into machine learning models for protein engineering. *J. Chem. Inf. Model.* **60,** 2773–2790 (2020).

36. Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* (2018).

37. Kearnes, S., McCloskey, K., Berndl, M., Pande, V. & Riley, P. Molecular graph convolutions: moving beyond fingerprints. *J. Comput.-Aided Mol. Des.* **30,** 595–608 (2016).

38. Duvenaud, D. K. *et al. Convolutional networks on graphs for learning molecular fingerprints* in *Advances in Neural Information Processing Systems* (2015), 2224–2232.

39. Zhou, J. *et al.* Graph neural networks: A review of methods and applications. *arXiv,* arXiv:1812.08434 (2018).

40. Dimmer, E. C. *et al.* The UniProt-GO annotation database in 2011. *Nucleic Acids Res.* **40,** D565–D570 (2012).

41. Bekker, J. & Davis, J. Learning from positive and unlabeled data: A survey. *Mach. Learn.* **109,** 719–760 (2020).

42. Tian, W. & Skolnick, J. How well is enzyme function conserved as a function of pairwise sequence identity? *J. Mol. Biol.* **333,** 863–882 (2003).

43. AlQuraishi, M. ProteinNet: a standardized data set for machine learning of protein structure. *BMC bioinformatics* **20,** 1–10 (2019).

44. Hu, W. *et al.* Strategies for pre-training graph neural networks. *arXiv* (2019).

45. Capela, F., Nouchi, V., Van Deursen, R., Tetko, I. V. & Godin, G. Multitask learning on graph neural networks applied to molecular property predictions. *arXiv* (2019).

46. Vaswani, A. *et al. Attention is all you need* in *Advances in neural information processing systems* (2017), 5998–6008.

47. Suzek, B. E. *et al.* UniRef clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics* **31,** 926–932 (2015).

48. Elnaggar, A. *et al.* ProtTrans: Towards Cracking the Language of Lifes Code Through Self-Supervised Deep Learning and High Performance Computing. *IEEE Trans. Pattern Anal. Mach. Intell.* **PP** (July 2021).

49. Chen, T. & Guestrin, C. *Xgboost: A scalable tree boosting system* in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (2016), 785–794.

50. Hsu, C., Nisonoff, H., Fannjiang, C. & Listgarten, J. Learning protein fitness models from evolutionary and assay-labeled data. *Nat. Biotechnol.,* 1–9 (2022).

51. Chicco, D. & Jurman, G. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC genomics* **21,** 1–13 (2020).

52. Chang, A. *et al.* BRENDA, the ELIXIR core data resource in 2021: new developments and updates. *Nucleic Acids Res.* **49,** D498–D508 (Jan. 2021).

53. Wittig, U., Rey, M., Weidemann, A., Kania, R. & Müller, W. SABIO-RK: an updated resource for manually curated biochemical reaction kinetics. *Nucleic Acids Res.* **46,** D656–D660 (2018).

54. Jumper, J. *et al.* Highly accurate protein structure prediction with AlphaFold. *Nature* **596,** 583–589 (2021).

55. Baek, M. *et al.* Accurate prediction of protein structures and interactions using a three-track neural network. *Science* **373,** 871–876 (2021).

56. Van Rossum, G. & Drake, F. L. *Python 3 Reference Manual* (CreateSpace, Scotts Valley, CA, 2009).

57. Paszke, A. *et al.* Pytorch: An imperative style, high-performance deep learning library. *Adv. Neur. In.* **32,** 8026–8037 (2019).

58. The Gene Ontology resource: enriching a GOld mine. *Nucleic Acids Res.* **49,** D325–D334 (2021).

59. Bansal, P. *et al.* Rhea, the reaction knowledgebase in 2022. *Nucleic Acids Res.* (2021).

60. Kanehisa, M. & Goto, S. KEGG: kyoto encyclopedia of genes and genomes. *Nucleic Acids Res.* **28,** 27–30 (2000).

61. Kim, S. *et al.* PubChem in 2021: new data content and improved web interfaces. *Nucleic Acids Res.* **49,** D1388–D1395 (2021).

62. Landrum, G. *et al. RDKit: Open-source cheminformatics* http://www.rdkit.org. 2006.

63. Fu, L., Niu, B., Zhu, Z., Wu, S. & Li, W. CD-HIT: accelerated for clustering the next-generation sequencing data. *Bioinformatics* **28,** 3150–3152 (2012).

64. Caspi, R. *et al.* The MetaCyc database of metabolic pathways and enzymes-a 2019 update. *Nucleic Acids Res.* **48,** D445–D453 (2020).

65. Jolliffe, I. Principal component analysis. *Encyclopedia of Statistics in Behavioral Science* (2005).

66. Bahdanau, D., Cho, K. & Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* (2014).

67. Bergstra, J., Yamins, D. & Cox, D. *Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures* in *International conference on machine learning* (2013), 115–123.

68. Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **12,** 2825–2830 (2011).

69. Dietterich, T. G. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Comput.* **10,** 1895–1923 (1998).

70. Seabold, S. & Perktold, J. *Statsmodels: Econometric and statistical modeling with python* in *Proceedings of the 9th Python in Science Conference* **57** (2010), 61.

71. Virtanen, P. *et al.* SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat. Methods* **17,** 261–272 (2020).

Supplementary Information

Supplementary Tables S1-S5

Supplementary Table 1. Results of hyperparameter optimizations of the gradient boosting models for all four combinations of small molecule representations (ECFPs and GNN generated fingerprints) and enzyme representations (*ESM-1b* and *ESM-1b$_{ts}$* vectors). The hyperparameter optimizations were performed with 5-fold cross-validation on the training set.

	mean ROC-AUC (CV)	**learning rate**	**max. delta step**	**max. depth**	**min. child weight**	**num. of trees**	**alpha coeff.**	**beta coeff.**	**weight**
***ESM-1b* & ECFP**	0.861	0.127	3.08	13	2.69	333	1.43	0.12	0.114
***ESM-1b$_{ts}$* & ECFP**	0.911	0.316	1.77	10	1.38	343	0.53	3.74	0.262
***ESM-1b* & GNN**	0.888	0.081	4.90	11	4.48	347	0.35	0.62	0.127
***ESM-1b$_{ts}$* & GNN**	0.926	0.198	3.82	12	0.96	358	0.37	4.44	0.113

Supplementary Table 2. Results of three different machine learning algorithms on the test set. Hyperparameter optimizations for all models were performed with 5-fold cross-validation on the training set.

	ROC-AUC score	**Accuracy**	**MCC**
Gradient Boosting	0.955	91.5%	0.78
Random Forest	0.945	87.7%	0.67
Logistic Regression	0.621	63.0%	0.14

Supplementary Table 3. Results of a gradient boosting model with *ESM-1b$_{ts}$* vectors and GNN-generated fingerprints (with a pre-trained GNN) compared to a gradient boosting model with *ESM-1b$_{ts}$* vectors and GNN-generated fingerprints (with a not pre-trained GNN). Results are shown for the test set. The hyperparameter optimizations for all models were performed with 5-fold cross-validation on the training set.

	ROC-AUC score	**Accuracy**	**MCC**
GNN-generated fingerprints with pre-trained GNN	0.955	91.5%	0.78
GNN-generated fingerprints with not pre-trained GNN	0.954	90.7%	0.77

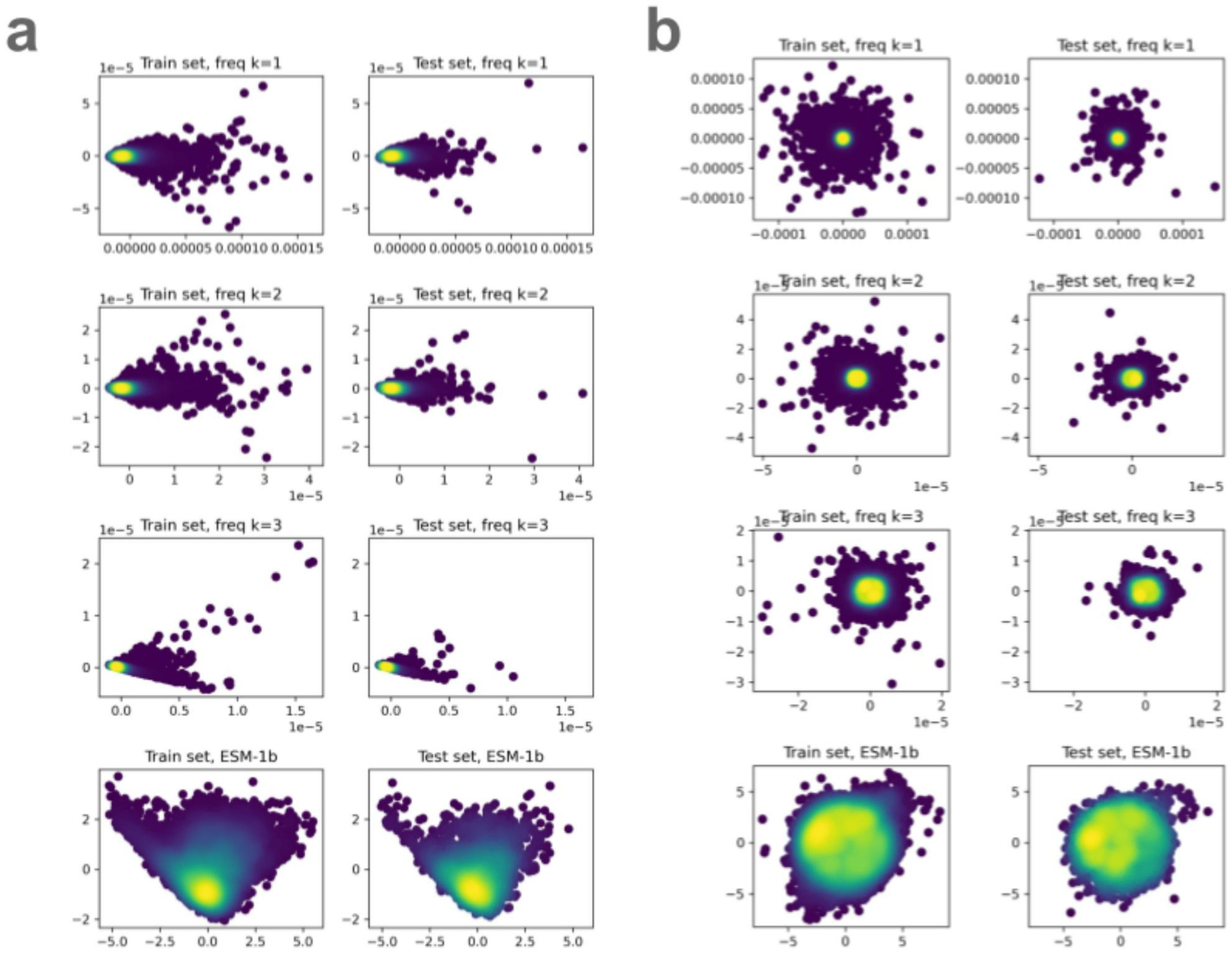
Supplementary Table 4. Results of three gradient boosting models with different enzyme representations. Models were trained with GNN-generated fingerprints as small molecule representations combined with three different enzyme representations: *ESM-1b* vectors, *ESM-1b$_{ts}$* vectors created without an extra token for the whole enzyme, and *ESM-1b$_{ts}$* vectors created with an extra token for the whole enzyme. Results are shown for the test set. The hyperparameter optimizations for all models were performed with 5-fold cross-validation on the training set.

	ROC-AUC score	**Accuracy**	**MCC**
ESM-1b	0.940	88.8%	0.72
ESM-1b$_{ts}$* *(mean representation)	0.956	90.9%	0.77
ESM-1b$_{ts}$* *(enzyme token)	0.956	91.5%	0.78

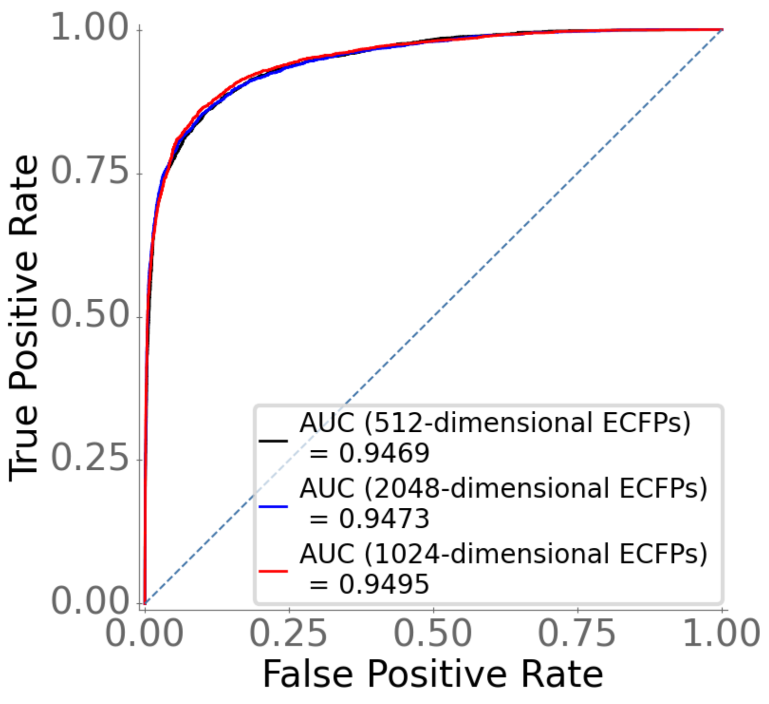
Supplementary Table 5. Results of validating the ESP model on the test sets from Yang et al.[15] and Mou et al.[14] without adding any new training data to our training set.

	ROC-AUC score	**Accuracy**	**MCC**
Yang et al. Avena strigosa	0.59	68%	0.12
Yang et al. Lycium barbarum	0.56	66%	0.01
Mou et al.	0.41	0.5%	0.00

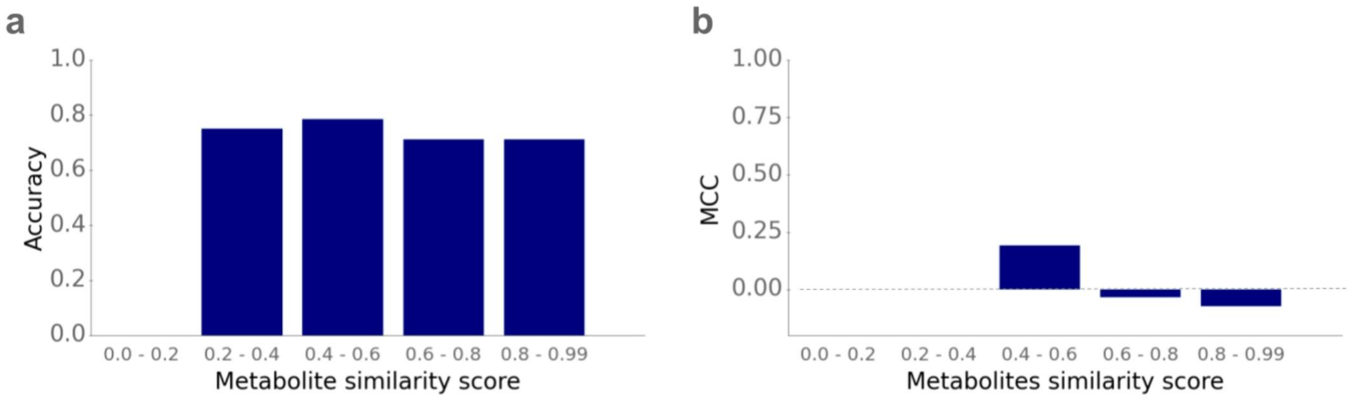
Supplementary Figures S1-S5



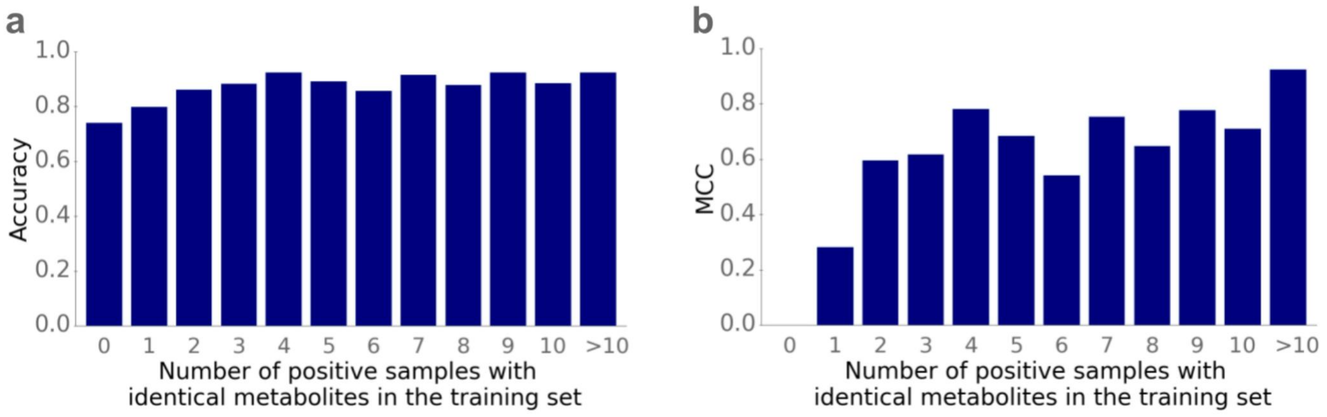
Supplementary Fig. 1. Similar distributions of enzymes in the training and the test sets. We projected numerical representations of the enzyme amino acid sequences to two-dimensional spaces, using two different types of representations: (i) We created vectors with the frequencies of specific k-mers of amino acids within the protein amino sequences (for $k = 1, 2, 3$) for all enzymes in the training and the test set; and (ii) we used the *ESM-1b* vectors. **(a)** Projection onto the first two principal components after Principal Component Analysis (PCA). **(b)** Multidimensional scaling (MDS) onto a two-dimensional space.



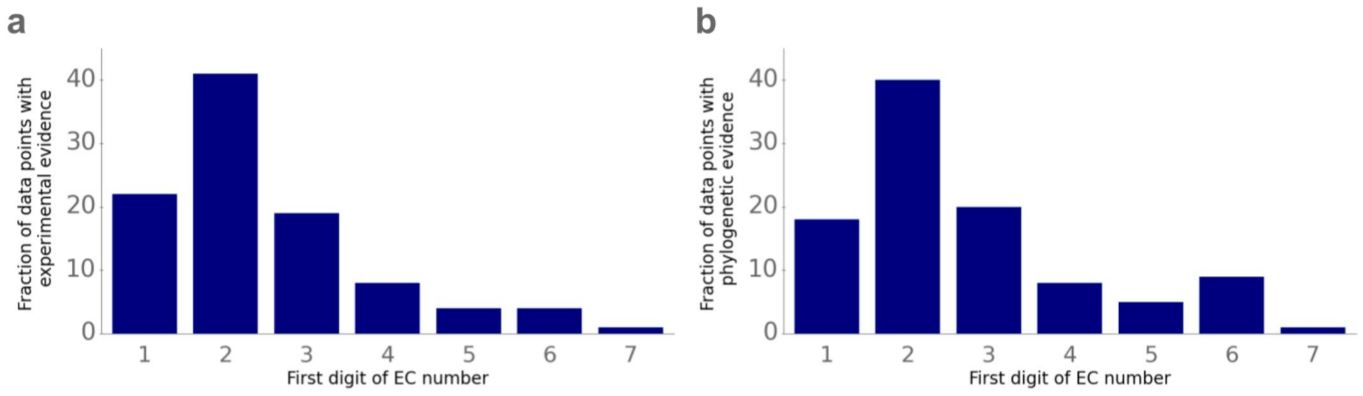
Supplementary Fig. 2. Effect of different dimensions of ECFPs. We calculated extended-connectivity fingerprints (ECFPs) as representations for small molecules using different dimensions, comparing 512-, 1 024-, and 2 048-dimensional ECFPs. The plot shows ROC curves resulting from optimized gradient boosting models. For training these models, ECFPs of different dimensions were combined with *ESM-1b*$_{ts}$ vectors as enzyme representations.



Supplementary Fig. 3. Effect of the metabolite similarity score on model performance. For all small molecules in the test set that do not also occur in the training set, we calculated the maximal pairwise similarity score across all small molecules in the training set. The similarity score is a value between 0 and 1, where a higher value indicates higher similarity between a pair of metabolites. We divided all test data points with small molecules that do not occur in the training set into five subsets dependent on their maximal similarity scores. **(a)** shows the accuracy and **(b)** shows the MCC for each subset.



Supplementary Fig. 4. Effect of the number of identical substrates in the training set on model performance. We grouped small molecules by how often they occur as substrates among all positive data points in the training set. **(a)** shows the accuracy and **(b)** shows the MCC for each group.



Supplementary Fig. 5. Enzymes with experimental evidence and with phylogenetic evidence do not differ strongly in their distribution across top level Enzyme Commission (EC) numbers. (a) Distribution across the first digit of EC numbers for all enzyme-substrate pairs with experimental evidence. **(b)** Distribution across the first digit of EC numbers for all enzyme-substrate pairs with phylogenetically inferred evidence.

4 OUTLOOK

In this thesis, I presented general methods for predicting the substrate scope of enzymes and for predicting enzyme kinetic parameters. The developed models are either the first of their kind or achieve substantially improved performance compared to all previous methods developed for the same tasks. The ESP model for predicting enzyme-substrate pairs achieved an accuracy of over 91% on an independent test set, and the model can help to find unknown primary and secondary functions of enzymes. Moreover, it was previously not possible to parameterize any genome-scale metabolic network model with realistic kinetic parameters for all enzymatic reactions. The prediction models presented in this thesis now allow full kinetic parameterizations of metabolic networks with K_M and k_{cat} values. On average, K_M and k_{cat} can be predicted with an accuracy of up to an order of magnitude. By parameterizing a metabolic network model with predicted k_{cat} values, I have shown that such an accuracy is sufficient to improve previous predictions for proteome allocation predictions. Additional work is required to investigate if full kinetic parameterizations of metabolic network models with predicted parameters K_M and k_{cat} can further help to gain deeper insights into cellular metabolism. Furthermore, knowing K_M and k_{cat} is not only desirable for parameterizing metabolic networks but is also highly relevant when studying single enzymes [77]. Values predicted with the presented models could thus be used to achieve first estimates before executing labor-intensive experiments.

Despite of the promising results, the presented models still have some limitations and there is room for improvement. One reason for this is that the field of applying deep learning to predict properties and numerical representations of proteins and small molecules is rapidly evolving. In the past two years alone, a large number of methods that improved previous state-of-the-art performances have been published in this area, including methods for the prediction of protein 3D structures [64, 65] and approaches to create improved protein and small molecule representations [62, 63, 78–80]. In this chapter, I will discuss some of the limitations of the models presented in this thesis, and I will give an outlook on how these limitations could be overcome. Moreover, I am going to discuss which prediction task could be solved next to further close knowledge gaps in metabolic networks.

4.1 NUMERICAL ENZYME REPRESENTATIONS

Enzyme representations obtained from linear amino acid sequences

To create enzyme representations, we used the state-of-the-art architecture for NLP tasks, a Transformer Network [81], which is typically very large and requires many training data points. The Facebook AI research (FAIR) team trained such a Transformer network, the *ESM-1b* model, with ~27 million amino acid sequences in a self-supervised way to create 1 280-dimensional numerical protein representations, which contain information about the structure and the function of the proteins [60].

For the task of predicting enzyme-substrate pairs, I fine-tuned and re-trained the *ESM-1b* model to create task-specific enzyme-representations, which led to improved model performance. This was possible because many training data points were available for this task. Unfortunately, for the task of predicting k_{cat}, not enough training data was available to perform a successful fine-tuning of the large *ESM-1b* model. Nevertheless, to increase model performance for predicting k_{cat}, we were able to use the fine-tuned representations that were created for predicting enzyme-substrate pairs.

In my first work, the prediction of Michaelis constants K_M, we used UniRep vectors [59] as enzyme representations. These vectors were created with a model using the former state-of-the-art architecture for sequence processing, the LSTM model [82]. Using the newer and improved *ESM-1b* vectors will likely increase model performance. As it was the case for the task of predicting k_{cat}, it is likely that not enough training data is available to perform a successful fine-tuning of the *ESM-1b* model to create task-specific enzyme representations. However, using the fine-tuned *ESM-1b* vectors, which were created for predicting enzyme-substrate pairs, could further improve the K_M prediction model.

Enzyme representations obtained from the protein 3D structure

Instead of using the linear amino acid sequence to create protein representations, alternatively the protein 3D structure can be used [61–63]. The structure of a protein defines its function, and thus, in principle it would be preferable to use such structure-based protein models. Unfortunately, until recently the structure was unknown for the vast majority of proteins. As a result, the amount of data points available to train structure-based models was very sparse. However, the recent development of AlphaFold [64] and RoseTTAFold [65] facilitates to predict the 3D structure of almost any protein using its amino acid sequence. In the near future this development will likely lead to improved deep learning models that make use of protein structures to create numerical protein representations. These vectors could replace or extend information extracted from the linear protein amino acid sequences, and hence, they could help to further improve the predictions for K_M and k_{cat} values and of enzyme-substrate pairs.

4.2 NUMERICAL METABOLITE REPRESENTATIONS

To represent metabolites as numerical vectors, I created task-specific fingerprints with deep learning models. I represented metabolites as graphs and used them as the input for graph neural networks (GNN), which were trained to either predict a substrate-specific K_M value or to predict enzyme-substrate pairs (*Manuscript 1* and *Manuscript 3*). The resulting metabolite representations significantly improved model performance in comparison to pre-defined expert-crafted fingerprints.

Recently, alternative deep learning based approaches for creating task-specific metabolite representations have been published [79, 80]. These models are Transformer Networks that receive string representations of metabolites, the simplified molecular-input line-entry system (SMILES) [83], as their input. To create task-specific metabolite representations, the models can be trained to predict a property of the inserted molecules. It has been shown that the resulting representations lead to improved performances in various downstream tasks when compared to GNN-generated and expert-crafted fingerprints. Hence, it could be promising to investigate if Transformer Networks that are trained to create task-specific molecular fingerprints for the prediction of K_M and of enzyme-substrate pairs improve model performance.

For the task of predicting turnover numbers k_{cat} of enzyme catalyzed reactions, it was required to represent all metabolites of a reaction numerically in one single vector. To achieve this, I used difference reaction fingerprints [67, 71], which do not vary in length even for varying numbers of reactants. These representations are calculated from pre-defined expert-crafted metabolite fingerprints for all substrates and all products. Similarly to the task-specific metabolite fingerprints that were created for the K_M and enzyme-substrate pair prediction, it would be desirable to create task-specific reaction fingerprints for the prediction of k_{cat}. To achieve this, a GNN or a Transformer network could be slightly modified and trained with all substrates and all products of a reaction as its input to predict the reaction's k_{cat} value. After training, task-specific reaction fingerprints could be extracted from the trained model. However, GNNs and Transformer Networks typically require large amounts of training data and it is thus unclear if enough data is available to create such representations. Pre-training the deep learning models in a self-supervised fashion on large reaction datasets or gathering more k_{cat}

training data could help in the process of creating task-specific reaction fingerprints.

4.3 AVAILABILITY AND QUALITY OF TRAINING DATA

The availability of experimentally measured or validated data points is a limiting factor for all of the presented prediction models. For example, I have shown that the biggest limitation of the enzyme-substrate prediction model, ESP, is its low performance for metabolites that have not been part of the training set. High-throughput screening methods for substrate scopes of enzymes are currently developed and will make much more training data available in the future [84–86]. This will likely increase the applicability of the ESP model to a much wider range of different substrates.

Especially the k_{cat} dataset with only $\sim 4\,300$ data points is rather small. When training machine learning models for complex tasks such as predicting enzyme turnover numbers, large amount of training data is required. Despite the limited dataset size, the final model is still able to explain $\sim 40\%$ of the variance of k_{cat} values in an independent test set. However, to be even more accurate and to account for experimental conditions such as pH and temperature, much more training data will need to become available.

Not only the availability but also the quality of training data is a crucial factor for model performance. Due to different experimental conditions and experimental procedures, k_{cat} and K_M values that were measured for the same enzyme, substrate, and reaction can vary widely between different studies; on average they differ by a factor of 2.5 and 2.9, respectively [87]. Moreover, Bar-Even *et al.* [87] found that up to 20% of the values in BRENDA do not correspond to the values in the reference papers. These errors can be caused by copying mistakes and unit mismatches. It is possible that similar errors also occur in the other databases that I used to create training and validation datasets. To correct copying and unit errors, we started to manually review all k_{cat} values that I extracted from BRENDA, and we already reviewed half of those data points (see *Manuscript 2*). However, a further manual review of all used data points from all databases is required to ensure high data quality.

4.4 CHOICE OF MACHINE LEARNING ALGORITHMS

Deep neural networks are the most powerful machine learning algorithms when dealing with homologous input data such as images or sound. However, when the input is heterogeneous and presented in the form of tabular data, algorithms based on gradient boosted decision trees usually outperform deep neural networks [88]. In this thesis, I created numerical enzyme, metabolite, and reaction vectors as the input for machine learning models. This input data is tabular, and hence, one would expect that gradient boosting models outperform deep neural networks. Indeed, for the prediction of Michaelis constants K_M, turnover numbers k_{cat}, and of enzyme-substrate pairs, gradient boosting models achieved better performance than fully-connected neural networks.

Although neural networks perform slightly worse on the presented prediction tasks, an ensemble of different machine learning models, including neural networks, could improve performances [89]. Often, different machine learning models focus on different aspects of the input features to make predictions. As a result, combining the prediction results of such models, e.g., by taking a weighted mean of the predicted values, could make predictions more robust and accurate in the future.

4.5 PREDICTING SUBSTRATES FOR TRANSPORT PROTEINS

In *Manuscript 3*, I presented a model that can help to determine the substrate scope of enzymes with yet not fully characterized function. A related problem is the prediction of substrates for transport proteins. Knowing the function of all transporters of a metabolic network, and hence, knowing which metabolites can be imported into a cell and exported out of a cell is crucial for simulating cellular metabolism. However, the partially hydrophobic surface of transporters and their lack of stability make it challenging to experimentally identify substrates of transporters [90]. Consequently, a similar prediction model as developed for enzymes could immensely help to determine the function of transporters.

Previously developed models for predicting functions of transport proteins are either based on calculating protein similarities to infer function [91, 92], or they do not map specific substrates to transport proteins but instead only connect transporters with large groups of substrates [93, 94]. Using state-of-the-art deep and machine learning models has the potential to increase the performance of these models and to connect transporters with specific substrates. Training such a model would require the creation of a large dataset with experimentally validated transport-substrate pairs. This could be achieved by searching existing protein databases [74, 75]. To numerically encode information about the prediction task, i.e., about the transport protein and the potential substrates, similar methods as used in the prediction models for K_M and k_{cat} values and of enzyme-substrate pairs could be used.

Training a model that classifies molecules as substrates and non-substrates for given transport proteins requires negative training data, i.e., transporter-molecule pairs where the molecule is not a substrate for the transporter. However, public protein databases only list positive instances. The same problem arose when training the enzyme-substrate pair prediction model, where I solved this issue by successfully creating negative data points by sampling from unlabeled data. This process led to good model performance on independent test data. A similar approach could be applied to the transporter-substrate pair prediction task.

4.6 CONCLUSION

The work presented in this thesis is a first step towards closing some of the knowledge gaps in metabolic reaction networks. I have developed prediction tools and methods that have the potential to provide deeper insights into cellular metabolism. The performances of the trained models are promising, but it remains to be seen in future work how valuable these will be in practice. As outlined in this chapter, there are still numerous possibilities and approaches that could be taken to improve the developed methods and to develop additional prediction models that further help to gain a fundamental understanding of metabolic processes in living organisms.

BIBLIOGRAPHY

1. Fang, X., Lloyd, C. J. & Palsson, B. O. Reconstructing organisms in silico: genome-scale models and their emerging applications. *Nat. Rev. Microbiol.* **18,** 731–743 (2020).
2. Orth, J. D., Thiele, I. & Palsson, B. Ø. What is flux balance analysis? *Nat. Biotechnol.* **28,** 245–248 (2010).
3. Kroll, A., Engqvist, M. K. M., Heckmann, D. & Lercher, M. J. Deep learning allows genome-scale prediction of Michaelis constants from structural features. *PLoS Biol.* **19,** 1–21 (Oct. 2021).
4. Davidi, D., Noor, E., Liebermeister, W., Bar-Even, A., Flamholz, A., Tummler, K., Barenholz, U., Goldenfeld, M., Shlomi, T. & Milo, R. Global characterization of in vivo enzyme catalytic rates and their correspondence to in vitro kcat measurements. *PNAS* **113,** 3401–3406 (2016).
5. Heckmann, D., Lloyd, C. J., Mih, N., Ha, Y., Zielinski, D. C., Haiman, Z. B., Desouki, A. A., Lercher, M. J. & Palsson, B. O. Machine learning applied to enzyme turnover numbers reveals protein structural correlates and improves metabolic models. *Nat. Commun.* **9,** 1–10 (2018).
6. Borger, S., Liebermeister, W. & Klipp, E. Prediction of enzyme kinetic parameters based on statistical learning. *Genom. Inform.* **17,** 80–87 (2006).
7. Khodayari, A., Zomorrodi, A. R., Liao, J. C. & Maranas, C. D. A kinetic model of Escherichia coli core metabolism satisfying multiple sets of mutant flux data. *Metab. Eng.* **25,** 50–62 (2014).
8. Khodayari, A. & Maranas, C. D. A genome-scale Escherichia coli kinetic metabolic model k-ecoli457 satisfying flux data for multiple mutant strains. *Nat. Commun.* **7,** 1–12 (2016).
9. Chakrabarti, A., Miskovic, L., Soh, K. C. & Hatzimanikatis, V. Towards kinetic modeling of genome-scale metabolic networks without sacrificing stoichiometric, thermodynamic and physiological constraints. *Biotechnol. J.* **8,** 1043–1057 (2013).
10. Li, F., Yuan, L., Lu, H., Li, G., Chen, Y., Engqvist, M. K., Kerkhoven, E. J. & Nielsen, J. Deep learning-based kcat prediction enables improved enzyme-constrained model reconstruction. *Nat. Catal.,* 1–11 (2022).
11. Ghatak, S., King, Z. A., Sastry, A. & Palsson, B. O. The y-ome defines the 35% of Escherichia coli genes that lack experimental evidence of function. *Nucleic Acids Res.* **47,** 2446–2454 (2019).

12. Milo, R. What is the total number of protein molecules per cell volume? A call to rethink some published values. *BioEssays* **35,** 1050–1055 (2013).

13. Ho, B., Baryshnikova, A. & Brown, G. W. Unification of protein abundance datasets yields a quantitative Saccharomyces cerevisiae proteome. *Cell Syst.* **6,** 192–205 (2018).

14. Eagon, R. G. Pseudomonas natriegens, a marine bacterium with a generation time of less than 10 minutes. *J. Bacteriol.* **83,** 736–737 (1962).

15. Cooper, G. M., Hausman, R. E. & Hausman, R. E. *The cell: a molecular approach* (ASM press Washington, DC, 2007).

16. Copley, S. D. Shining a light on enzyme promiscuity. *Curr. Opin. Struct. Biol.* **47,** 167–175 (2017).

17. Tawfik, O. K. & S, D. Enzyme promiscuity: a mechanistic and evolutionary perspective. *Annu. Rev. Biochem.* **79,** 471–505 (2010).

18. Nobeli, I., Favia, A. D. & Thornton, J. M. Protein promiscuity and its implications for biotechnology. *Nat. Biotechnol.* **27,** 157–167 (2009).

19. Lane, N. *The vital question: energy, evolution, and the origins of complex life* (WW Norton & Company, 2015).

20. Li, Z., Wang, R.-S. & Zhang, X.-S. Drug target identification based on flux balance analysis of metabolic networks. *J. Comput. Syst. Biol.* **13,** 331–338 (2010).

21. Bordbar, A., McCloskey, D., Zielinski, D. C., Sonnenschein, N., Jamshidi, N. & Palsson, B. O. Personalized whole-cell kinetic models of metabolism for discovery in genomics and pharmacodynamics. *Cell Syst.* **1,** 283–292 (2015).

22. Murabito, E., Smallbone, K., Swinton, J., Westerhoff, H. V. & Steuer, R. A probabilistic approach to identify putative drug targets in biochemical networks. *J. R. Soc. Interface* **8,** 880–895 (2011).

23. Haanstra, J. R., Gerding, A., Dolga, A. M., Sorgdrager, F. J., Buist-Homan, M., Du Toit, F., Faber, K. N., Holzhütter, H.-G., Szöör, B., Matthews, K. R., *et al.* Targeting pathogen metabolism without collateral damage to the host. *Sci. Rep.* **7,** 1–15 (2017).

24. Adrio, J. L. & Demain, A. L. Microbial enzymes: tools for biotechnological processes. *Biomolecules* **4,** 117–139 (2014).

25. Wang, S., Fu, C., Liu, K., Cui, J., Hu, H., Wang, W. & Zhang, X. Engineering a Synthetic Pathway for Gentisate in Pseudomonas Chlororaphis P3. *Front. Bioeng. Biotechnol.* **8,** 1588 (2021).

26. Wu, M.-C., Law, B., Wilkinson, B. & Micklefield, J. Bioengineering natural product biosynthetic pathways for therapeutic applications. *Curr. Opin. Biotechnol.* **23,** 931–940 (2012).

27. Feist, A. M. & Palsson, B. O. The biomass objective function. *Curr. Opin. Microbiol.* **13,** 344–349 (2010).

28. Schuster, S., Pfeiffer, T. & Fell, D. A. Is maximization of molar yield in metabolic networks favoured by evolution? *J. Theor. Biol.* **252.** In Memory of Reinhart Heinrich, 497–504. ISSN: 0022-5193 (2008).

29. Montagud, A., Navarro, E., Fernandez de Cordoba, P., Urchueguía, J. F. & Patil, K. R. Reconstruction and analysis of genome-scale metabolic model of a photosynthetic bacterium. *BMC Syst. Biol.* **4,** 1–16 (2010).

30. Motter, A. E., Gulbahce, N., Almaas, E. & Barabási, A.-L. Predicting synthetic rescues in metabolic networks. *Mol. Syst. Biol.* **4,** 168 (2008).

31. Hastings, J., Mains, A., Virk, B., Rodriguez, N., Murdoch, S., Pearce, J., Bergmann, S., Le Novère, N. & Casanueva, O. Multi-omics and genome-scale modeling reveal a metabolic shift during C. elegans aging. *Front. Mol. Biosci.* **6,** 2 (2019).

32. Folger, O., Jerby, L., Frezza, C., Gottlieb, E., Ruppin, E. & Shlomi, T. Predicting selective drug targets in cancer through metabolic networks. *Mol. Syst. Biol.* **7,** 501 (2011).

33. Burgard, A. P., Pharkya, P. & Maranas, C. D. Optknock: a bilevel programming framework for identifying gene knockout strategies for microbial strain optimization. *Biotechnol. Bioeng.* **84,** 647–657 (2003).

34. Guan, N., Du, B., Li, J., Shin, H.-d., Chen, R. R., Du, G., Chen, J. & Liu, L. Comparative genomics and transcriptomics analysis-guided metabolic engineering of Propionibacterium acidipropionici for improved propionic acid production. *Biotechnol. Bioeng.* **115,** 483–494 (2018).

35. Cautha, S. C., Gowen, C. M., Lussier, F.-X., Gold, N. D., Martin, V. J. & Mahadevan, R. Model-driven design of a Saccharomyces cerevisiae platform strain with improved tyrosine production capabilities. *IFAC Proc. Vol.* **46,** 221–226 (2013).

36. McAnulty, M. J., Yen, J. Y., Freedman, B. G. & Senger, R. S. Genome-scale modeling using flux ratio constraints to enable metabolic engineering of clostridial metabolism in silico. *BMC Syst. Biol.* **6,** 1–15 (2012).

37. Michaelis, L., Menten, M. L., *et al.* Die Kinetik der Invertinwirkung. *Biochem. z* **49,** 352 (1913).

38. Smallbone, K., Simeonidis, E., Swainston, N. & Mendes, P. Towards a genome-scale kinetic model of cellular metabolism. *BMC Syst. Biol.* **4,** 1–9 (2010).

39. Smallbone, K., Simeonidis, E., Broomhead, D. S. & Kell, D. B. Something from nothing- bridging the gap between constraint-based and kinetic modelling. *FEBS J.* **274,** 5576–5585 (2007).

40. Dourado, H. & Lercher, M. J. An analytical theory of balanced cellular growth. *Nat. Commun.* **11,** 1–14 (2020).

41. Lerman, J. A., Hyduke, D. R., Latif, H., Portnoy, V. A., Lewis, N. E., Orth, J. D., Schrimpe-Rutledge, A. C., Smith, R. D., Adkins, J. N., Zengler, K., *et al.* In silico method for modelling metabolism and gene product expression at genome scale. *Nat. Commun.* **3,** 1–10 (2012).

42. O'brien, E. J., Lerman, J. A., Chang, R. L., Hyduke, D. R. & Palsson, B. Ø. Genome-scale models of metabolism and gene expression extend and refine growth phenotype prediction. *Mol. Syst. Biol.* **9,** 693 (2013).

43. Saa, P. A. & Nielsen, L. K. Formulation, construction and analysis of kinetic models of metabolism: A review of modelling frameworks. *Biotechnol. Adv.* **35,** 981–1003 (2017).

44. Strutz, J., Martin, J., Greene, J., Broadbelt, L. & Tyo, K. Metabolic kinetic modeling provides insight into complex biological questions, but hurdles remain. *Curr. Opin. Biotechnol* **59,** 24–30 (2019).

45. Kroll, A., Hu, X.-P., Liebrand, N. A. & Lercher, M. J. Turnover number predictions for kinetically uncharacterized enzymes using machine and deep learning. *bioRxiv* (2022).

46. Yan, S.-M., Shi, D.-Q., Nong, H. & Wu, G. Predicting Km values of beta-glucosidases using cellobiose as substrate. *Interdiscip. sci. comput. life sci.* **4,** 46–53 (2012).

47. Ryu, J. Y., Kim, H. U. & Lee, S. Y. Deep learning enables high-quality and high-throughput prediction of enzyme commission numbers. *PNAS* **116,** 13996–14001 (2019).

48. Li, Y., Wang, S., Umarov, R., Xie, B., Fan, M., Li, L. & Gao, X. DEEPre: sequence-based enzyme EC number prediction by deep learning. *Bioinformatics* **34,** 760–769 (2018).

49. Sanderson, T., Bileschi, M. L., Belanger, D. & Colwell, L. ProteInfer: deep networks for protein functional inference. *bioRxiv* (2021).

50. Bileschi, M. L., Belanger, D., Bryant, D. H., Sanderson, T., Carter, B., Sculley, D, Bateman, A., DePristo, M. A. & Colwell, L. J. Using deep learning to annotate the protein universe. *Nat. Biotechnol.,* 1–6 (2022).

51. Visani, G. M., Hughes, M. C. & Hassoun, S. Enzyme promiscuity prediction using hierarchy-informed multi-label classification. *Bioinformatics* **37,** 2017–2024 (2021).

52. Rembeza, E. & Engqvist, M. K. Experimental investigation of enzyme functional annotations reveals extensive annotation error. *bioRxiv* (2020).

53. Mou, Z., Eakes, J., Cooper, C. J., Foster, C. M., Standaert, R. F., Podar, M., Doktycz, M. J. & Parks, J. M. Machine learning-based prediction of enzyme substrate scope: application to bacterial nitrilases. *Proteins Struct. Funct. Bioinf.* **89,** 336–347 (2021).

54. Yang, M., Fehl, C., Lees, K. V., Lim, E.-K., Offen, W. A., Davies, G. J., Bowles, D. J., Davidson, M. G., Roberts, S. J. & Davis, B. G. Functional and informatics analysis enables glycosyltransferase activity prediction. *Nat. Chem. Biol.* **14,** 1109–1117 (2018).

55. Pertusi, D. A., Moura, M. E., Jeffryes, J. G., Prabhu, S., Biggs, B. W. & Tyo, K. E. Predicting novel substrates for enzymes with minimal experimental effort with active learning. *Metab. Eng.* **44,** 171–181 (2017).

56. Röttig, M., Rausch, C. & Kohlbacher, O. Combining Structure and Sequence Information Allows Automated Prediction of Substrate Specificities within Enzyme Families. *PLoS Comput. Biol.* **6,** 1–8 (Jan. 2010).

57. Chevrette, M. G., Aicheler, F., Kohlbacher, O., Currie, C. R. & Medema, M. H. SANDPUMA: ensemble predictions of nonribosomal peptide chemistry reveal biosynthetic diversity across Actinobacteria. *Bioinformatics* **33,** 3202–3210 (2017).

58. Kulmanov, M., Khan, M. A. & Hoehndorf, R. DeepGO: predicting protein functions from sequence and interactions using a deep ontology-aware classifier. *Bioinformatics* **34,** 660–668 (2018).

59. Alley, E. C., Khimulya, G., Biswas, S., AlQuraishi, M. & Church, G. M. Unified rational protein engineering with sequence-based deep representation learning. *Nat. Methods* **16,** 1315–1322 (2019).

60. Rives, A. *et al.* Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *PNAS* **118,** e2016239118 (2021).

61. Derevyanko, G., Grudinin, S., Bengio, Y. & Lamoureux, G. Deep convolutional networks for quality assessment of protein folds. *Bioinformatics* **34,** 4046–4053 (2018).

62. Zhang, Z., Xu, M., Jamasb, A., Chenthamarakshan, V., Lozano, A., Das, P. & Tang, J. Protein Representation Learning by Geometric Structure Pretraining. *arXiv* (2022).

63. Gligorijević, V., Renfrew, P. D., Kosciolek, T., Leman, J. K., Berenberg, D., Vatanen, T., Chandler, C., Taylor, B. C., Fisk, I. M., Vlamakis, H., *et al.* Structure-based protein function prediction using graph convolutional networks. *Nat. Commun.* **12,** 1–14 (2021).

64. Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., *et al.* Highly accurate protein structure prediction with AlphaFold. *Nature* **596,** 583–589 (2021).

65. Baek, M., DiMaio, F., Anishchenko, I., Dauparas, J., Ovchinnikov, S., Lee, G. R., Wang, J., Cong, Q., Kinch, L. N., Schaeffer, R. D., *et al.* Accurate prediction of protein structures and interactions using a three-track neural network. *Science* **373,** 871–876 (2021).

66. Rogers, D. & Hahn, M. Extended-connectivity fingerprints. *J. Chem. Inf. Model.* **50,** 742–754 (2010).

67. Landrum, G. *et al. RDKit: Open-source cheminformatics* http://-www.rdkit.org. 2006.

68. Durant, J. L., Leland, B. A., Henry, D. R. & Nourse, J. G. Reoptimization of MDL keys for use in drug discovery. *J. Chem. Inf. Comput. Sci.* **42,** 1273–1280 (2002).

69. Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C. & Sun, M. Graph neural networks: A review of methods and applications. *AI Open* **1,** 57–81 (2020).

70. Yang, K., Swanson, K., Jin, W., Coley, C., Eiden, P., Gao, H., Guzman-Perez, A., Hopper, T., Kelley, B., Mathea, M., *et al.* Analyzing learned molecular representations for property prediction. *J. Chem. Inf. Model.* **59,** 3370–3388 (2019).

71. Hu, Q.-N., Zhu, H., Li, X., Zhang, M., Deng, Z., Yang, X. & Deng, Z. Assignment of EC numbers to enzymatic reactions with reaction difference fingerprints. *PLoS One* **7,** 1–6 (2012).

72. Chang, A., Jeske, L., Ulbrich, S., Hofmann, J., Koblitz, J., Schomburg, I., Neumann-Schaal, M., Jahn, D. & Schomburg, D. BRENDA, the ELIXIR core data resource in 2021: new developments and updates. *Nucleic Acids Res.* **49,** D498–D508 (2021).

73. Wittig, U., Rey, M., Weidemann, A., Kania, R. & Müller, W. SABIO-RK: an updated resource for manually curated biochemical reaction kinetics. *Nucleic Acids Res.* **46,** D656–D660 (2018).

74. Consortium, U. UniProt: a worldwide hub of protein knowledge. *Nucleic Acids Res.* **47,** D506–D515 (2019).

75. Dimmer, E. C., Huntley, R. P., Alam-Faruque, Y., Sawford, T., O'Donovan, C., Martin, M. J., Bely, B., Browne, P., Mun Chan, W., Eberhardt, R., *et al.* The UniProt-GO annotation database in 2011. *Nucleic Acids Res.* **40,** D565–D570 (2012).

76. Kroll, A., Ranjan, S., Engqvist, M. K. & Lercher, M. J. The substrate scopes of enzymes: a general prediction model based on machine and deep learning. *bioRxiv* (2022).

77. McDonald, A. G. & Tipton, K. F. Parameter Reliability and Understanding Enzyme Function. *Molecules* **27,** 263 (2022).

78. Meier, J., Rao, R., Verkuil, R., Liu, J., Sercu, T. & Rives, A. Language models enable zero-shot prediction of the effects of mutations on protein function. *Adv Neural Inf Process Syst.* **34,** 29287–29303 (2021).

79. Irwin, R., Dimitriadis, S., He, J. & Bjerrum, E. J. Chemformer: a pre-trained transformer for computational chemistry. *Mach. Learn.: Sci. Technol.* **3,** 015022 (2022).

80. Fabian, B., Edlich, T., Gaspar, H., Segler, M., Meyers, J., Fiscato, M. & Ahmed, M. Molecular representation learning with language models and domain-relevant auxiliary tasks. *arXiv* (2020).

81. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. & Polosukhin, I. Attention is all you need. *Adv Neural Inf Process Syst.* **30** (2017).

82. Hochreiter, S. & Schmidhuber, J. Long short-term memory. *Neural Comput.* **9,** 1735–1780 (1997).

83. Weininger, D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.* **28,** 31–36 (1988).

84. Rembeza, E., Boverio, A., Fraaije, M. W. & Engqvist, M. K. Discovery of Two Novel Oxidases Using a High-Throughput Activity Screen. *ChemBioChem* **23,** e202100510 (2022).

85. Longwell, C. K., Labanieh, L. & Cochran, J. R. High-throughput screening technologies for enzyme engineering. *Curr. Opin. Biotechnol.* **48,** 196–202 (2017).

86. Black, G. W., Brown, N. L., Perry, J. J., Randall, P. D., Turnbull, G. & Zhang, M. A high-throughput screening method for determining the substrate scope of nitrilases. *Chem. Commun.* **51,** 2660–2662 (2015).

87. Bar-Even, A., Noor, E., Savir, Y., Liebermeister, W., Davidi, D., Tawfik, D. S. & Milo, R. The moderately efficient enzyme: evolutionary and physicochemical trends shaping enzyme parameters. *Biochemistry* **50,** 4402–4410 (2011).

88. Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M. & Kasneci, G. Deep neural networks and tabular data: A survey. *arXiv* (2021).

89. Pintelas, P. & Livieris, I. E. Special Issue on Ensemble Learning and Applications. *Algorithms* **13.** ISSN: 1999-4893 (2020).

90. Larsen, B., Xu, D., Halkier, B. A. & Nour-Eldin, H. H. Advances in methods for identification and characterization of plant transporter function. *J. Exp. Bot.* **68,** 4045–4056 (2017).

91. Elbourne, L. D., Tetu, S. G., Hassan, K. A. & Paulsen, I. T. TransportDB 2.0: a database for exploring membrane transporters in sequenced genomes from all domains of life. *Nucleic Acids Res.* **45,** D320–D324 (2017).

92. Saier Jr, M. H., Reddy, V. S., Moreno-Hagelsieb, G., Hendargo, K. J., Zhang, Y., Iddamsetty, V., Lam, K. J. K., Tian, N., Russum, S., Wang, J., *et al.* The transporter classification database (TCDB): 2021 update. *Nucleic Acids Res.* **49,** D461–D467 (2021).

93. Alballa, M., Aplop, F. & Butler, G. TranCEP: Predicting the substrate class of transmembrane transport proteins using compositional, evolutionary, and positional information. *PLoS One* **15,** 1–23 (2020).

94. Mishra, N. K., Chang, J. & Zhao, P. X. Prediction of membrane transport proteins and their substrate specificities using primary sequence information. *PLoS One* **9,** 1–14 (2014).