CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Prediction of Drug Metabolites Using a Deep Learning Language Model

Master's thesis in Computer science and engineering

AMANDA DEHLÉN
PÄR ARONSSON

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Master's thesis 2024

Prediction of Drug Metabolites Using a Deep Learning Language Model

AMANDA DEHLÉN
PÄR ARONSSON

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Data Science and AI Division
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2024

Prediction of Drug Metabolites Using a Deep Learning Language Model
AMANDA DEHLÉN
PÄR ARONSSON

Supervisor: Dr. Rocío Mercado Oropeza, Department of Computer Science and Engineering, Chalmers
Advisor: Dr. Filip Miljković, AstraZeneca
Examiner: Ola Engkvist, Department of Computer Science and Engineering, Chalmers

Master's Thesis 2024
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2024

Prediction of Drug Metabolites Using a Deep Learning Language Model
AMANDA DEHLÉN
PÄR ARONSSON
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

The understanding of metabolism is essential in drug development, but conducting drug metabolism experiments is resource-intensive. To support this, *in silico* experiments using machine learning have been explored, with several tools available, but these rely on rule-based assessments and are restricted in their scalability. To build a better model for metabolite prediction in drug discovery, a deep neural network model called the *Focused Transformer* has been explored.

For the model, metabolite data was gathered and curated. Several strategies were explored to improve the model's performance, including a novel pretraining strategy involving pairs of structurally analogous molecules termed matched molecular pairs. The best derived model managed to find one true metabolite and had a validity of 4.5% when evaluated on an internal test set. While the model shows reasonable prediction for metabolite prediction, there is potential to achieve higher performance in future work and we conclude by suggesting several potential strategies that can be explored further, such as handling of data during training.

Keywords: drug development, deep learning, drug metabolites, Focused transformer, language model, metabolism, neural network.

Acknowledgements

We would like to thank our academic supervisor Rocío Mercado Oropeza and company advisor Filip Miljović for their invaluable feedback, insightful discussions, and for keeping us on the right track. We also want to acknowledge the feedback and support given by the two teams at Chalmers: AI Laboratory for Biomolecular Engineering (AIBE) team and Artificial Intelligence in the Natural Sciences (AIMLeNS) group. A thanks to AstraZeneca is also given, for the opportunity to perform this master thesis. Ola Engkvist also deserves our thanks for his dedicated service as our examiner.

We want to also acknowledge the work of the creators of the model Focused Transformer, which have made this project possible.

Finally, we want to say thank you to our friends and family for their support during our time at the university and this thesis.

Amanda Dehlén, Pär Aronsson, Gothenburg, 2024-07-18

Contents

List of Figures

List of Tables

List of Terms

endogenous substance that is native to a living organism. 8, 17, 43

excretion elimination of metabolic waste via, e.g., urine. 8

exogenous substance foreign to living organisms. 43

hydrolosis a chemical reaction with water as one of the reactants. 8

hydrophilic soluble in water. 8

lipophilic soluble in lipids, or oil. 8

metabolite a product of metabolism [1]. ix, 7, 17, 18

oxidation a chemical reaction in which the number of electrons associated with an atom or a molecule is decreased [2]. 8

polar a molecule with the sides positively and negatively charged[3]. 8

reduction a chemical reaction in which the number of electrons associated with an atom or a molecule is increased [2]. 8

xenobiotics chemical substances not naturally produced by the body[4]. 8

1

Introduction

Metabolism is a set of biochemical reactions that, mediated by enzymes, sustain life, either by providing energy and building blocks for cells or by eliminating potentially harmful compounds [5]. Although metabolic processes can act as a protection mechanism against foreign subjects, they can also alter the structure of a drug, affecting its efficacy and safety. Therefore, a comprehensive understanding of the metabolism of a potential drug is essential during drug development [6]. The study of drug metabolism helps determine the appropriate dosage, the location for effect, and how the drug interacts with distinct environments. The study of metabolism is crucial to the management of drug absorption, distribution, metabolism, excretion, and toxicity (ADMET) within the body, so that drug development efforts are focused on promising therapeutic molecules only [6].

Currently, conducting drug metabolism experiments is resource-intensive in regards to the preparation, execution, and analysis that goes into it. This is further compounded by the number of skilled personnel and technical equipment necessary to conduct the experiments, which further adds to the costs [7]. Therefore, computational experimentation, also called *in silico* techniques, is frequently explored as an alternative [8]. Determining the chemical structure of metabolites, the by-products of metabolic reactions, via *in silico* techniques has the potential to decrease the number of traditional experiments and speed up the design-make-test-analyze (DMTA) cycle. However, many of such methods are enzyme-focused and dependent on strictly derived transformation rules in the post-processing step, which limits their scope and scalability in terms of the metabolites that can be predicted and identified [8].

There is a potential to leverage deep neural networks (DNN) for metabolite prediction, which are not constrained by specific transformation rules. Instead, they can identify unseen connections through complex networks with many trainable parameters [9]. The use of DNNs could reveal new connections and complex patterns in the drug metabolism domain, providing valuable insights for researchers to make future predictions and more informed decisions during drug development. In addition, DNNs can use the acquired knowledge to predict metabolites of novel drug candidates that were not part of the training set [8]. This sets them apart from traditional techniques that rely on rule-based assessments and are restricted in their scalability.

This project will build on a previous literature report by Tworkowski, Staniszewski, Pacek, *et al.* [10] with the implementation of a DNN model capable of predicting

potential metabolites. The Focused Transformer [10] (FoT) is a model that has been recently proposed for extending the context length of the model via a training procedure inspired by contrastive learning. The hypothesis is that this could make the FoT better at learning from small data sets, such as metabolite data, thus making it ideal for training a model for metabolite prediction. The proposed model will be developed and evaluated using publicly available metabolite transformation data of biologically active molecules.

The code for this project can be found here:
`https://github.com/giffel99/metabolism_prediction`

1.1 Aim

This project aims to further explore the potential of deep learning models for predicting metabolites of a given drug molecule. A deep learning model called the Focused Transformer will be used. With the use of data consisting of metabolic reactions, the model will be trained to predict potential metabolites for drug molecules. The exploration will be performed with the aid of existing research and relevant strategies, such as pretraining, ensemble modeling, and augmentation of data, thereby enhancing the model's performance further. In the final steps of the evaluation, the model will be compared with other established metabolic prediction models to measure its predictive power.

1.2 Objective

In summary, this thesis aims to realize the following goals:

- to curate publicly available metabolite reaction data to be adapted for deep learning approaches,
- to explore a deep learning approach, the Focused Transformer (FoT)[10], for predicting drug metabolites,
- and to evaluate the model against state-of-the-art metabolite prediction tools.

1.3 Thesis Outline

Chapter 2 will present relevant past research that aided the project giving insights into what has been done within this field of study.

Chapter 3 will present the theoretical background for understanding the thesis. It will discuss chemistry applicable to drug metabolism and machine learning concepts.

Chapter 4 will present the methods used to conduct the project. It will present the background of the data sources, how the data was curated and preprocessed, as well as how the model framework and its individual components were designed. Lastly, it will describe how the evaluation was conducted.

Chapter 5 will present the results of the project. It will display the performance of different strategies applied to the model, including the comparison with the state-of-the-art models in the public domain.

Chapter 6 will discuss the choices taken in the project in regards to data curation, machine learning concepts, and strategies. It will also review the outcome of the model in regard to the results from Chapter 5.

Chapter 7 will present insights gained from the results and a conclusion of the project.

2

Related Works

In early 2019, Djoumbou-Feunang, Yannick, Fiamoncini, *et al.*[8] presented Biotransformer, a metabolite prediction and identification tool that combines machine learning and knowledge-based approaches to predict metabolites of small molecules in different metabolic enzyme environments (e.g. CYP450, human gut microbial, etc.). The model was comprised of a reasoning engine along with five modules, each assigned one environment. The reasoning engine's task was to send an input molecule to the most suitable module. Each module, consisting of a trained model for a single environment, would then predict the metabolites. During its final evaluation, the model was tested to predict human single-step metabolism for 40 pharmaceuticals and pesticides, scoring a recall of 0.88 and a precision of 0.49. The results were described as a success as it held up to other similar tools. Among the potential future improvements, the authors discussed the inclusion of a more diverse metabolic reaction set which would enable greater prediction coverage.

In 2020, Bruyn Kops, Sícho, Mazzolari, *et al.* [11] presented the prediction tool GLORYx, which combined machine learning with a rule-based approach (i.e., reaction rules) for metabolite prediction. The predicted metabolites were ranked using the machine learning-based site of metabolism (SoM) prediction tool FAME 3 [12]. The goal was to predict the structures of the metabolites that could potentially be formed by metabolism in either phase 1 and/or phase 2 [13]. GLORYx was evaluated on a manually curated data set consisting of the 100 best-selling drugs of 2018. Similar models have later used this test set to evaluate their performance in comparison to what GLORYx achieved, where the reported area under the curve (AUC) score was 0.79%

In 2020, Litsa, Das, and Kavraki[9] presented a neural machine translation model called MetaTrans that enabled metabolite prediction in a rule-free fashion [9]. It demonstrated the potential of computational approaches not relying on strictly defined metabolic transformation rules to predict possible drug metabolites. The deep learning transformer model showed equivalent performance to existing approaches for identifying metabolites by the major enzyme families, while also finding metabolites by more uncommon enzymes. They found that the approach had the potential to provide a more extensive study of drug metabolism than rule-based approaches, addressing the problems with limited scalability and lack of generalization.

To the best of our knowledge, MetaTrans is the only model that operates solely on a deep machine learning (DML) approach, not relying on any rule-based sys-

tem. When compared to the other models (GLORYx and Biotransformer), MetaTrans performs equivalently and better in certain metrics, e.g., finding less common metabolites [9]. The only evaluation performed by all three models was predicting the top-13 ranked metabolites. Some takeaways were that MetaTrans achieved the highest on the metrics "at least one metabolite found" and "all metabolites" for a drug molecule, but Biotransformer achieved the highest precision (13.5%) and recall (64.2%) compared to MetaTrans (precision: 12.0%; recall: 60.9%).

Evidently, the machine learning approaches for metabolite prediction have only partly been explored, in particular the deep learning rule-free method, thus creating the foundation and motivation for this thesis project.

3

Background

This section details the background and information needed to understand the rest of the project. Section 3.1 discusses the cheminformatics concepts and Section 3.2 presents machine learning concepts relevant to the project.

3.1 Cheminformatics

This section describes the key aspects of biochemistry and cheminformatics associated with the report.

3.1.1 Metabolism

Metabolism, which can also be referred to as biotransformations, is a set of biochemical reactions that occur within each organism [14]. The majority takes place within the liver [15]. However, it can also occur in other tissues, such as the intestine, kidney, lung, and skin.

Metabolic transformations generally occur through a series of enzyme-catalyzed reactions in a defined sequence, known as metabolic pathways [16]. Enzymes are mostly, but not only, proteins that regulate the rate at which the reactions occur without itself being altered in the process.

The reactions are divided into two subcategories; catabolism and anabolism [14]. Catabolism is comprised of metabolic pathways that degrade or break down compounds, typically larger molecules, into simpler products that can be used as building blocks. They release chemical energy in the process.

Anabolism are the sequences of enzyme-catalyzed reactions where compounds are formed in living cells from relatively simple structures [17]. For this, energy produced by catabolism is required. The anabolic processes dominate over catabolic ones in growing cells, whereas a balance between the two processes exists in non-growing cells.

The byproducts of metabolism, i.e. metabolites, can be characterized by active, inactive, or toxic [4]. Active metabolites and toxic metabolites are both biochemically active compounds, with the former exhibiting therapeutic effects or fulfilling a particular physiological function and the latter displaying various harmful effects.

Inactive metabolites are biochemically inactive compounds with neither therapeutic nor toxic effects.

3.1.1.1 Stages of metabolism

The pathways of metabolism are divided into phase I, phase II and phase III. These reactions may take place as single reactions, sequentially, simultaneously, or in reverse [15].

Phase I metabolism consists of reactions of oxidation, reduction, and hydrolosis, the former two meaning the loss or addition of one or more electrons, and the later a reaction with water [15], [2], [18]. These enzyme-catalyzed reactions convert lipophilic compounds (soluble in lipids, or oil) by adding or exposing a polar functional group such as -NH2 or -OH. This produces a polar, water-soluble metabolite that is often still active. Many of the products of this phase can become substrates for phase II [15].

Phase II metabolism consists of reactions that add endogenous hydrophilic (water-soluble) groups to the compound [15]. Endogenous molecules are chemical substances naturally existing in the body. Phase II produces a large polar, water-soluble, inactive metabolite that can be excreted from the body.

Phase III occurs post-phase II, where the compound goes through further metabolism and excretion [15].

3.1.1.2 Drug metabolism

Most drugs are xenobiotics, i.e., chemical substances not naturally produced by the body [4]. Xenobiotics go through metabolic transformations, thus reducing their bioactivity and allowing them to be available for excretion. The metabolism of drugs can occur in various reactions, as previously mentioned, categorized as phase I, phase II, and in some instances, phase III (additional modification and excretion).

Quantitatively, the most important source of drug-metabolizing enzymes is the liver, although enzymes can be found throughout the body [19]. Cytochrome P450 (CYP450) is a group of enzyme families that catalyze the oxidation and metabolism of a large number of xenobiotics and endogenous compounds [20]. It can be seen as the primary defense against xenobiotics. CYP450 enzymes primarily exist in the liver but also in other cells throughout the body. CYP450 enzymes are divided, based on sequence similarity, into 18 families. In humans, almost 80% of oxidative metabolism and approximately 50% of the overall elimination of common clinical drugs can be attributed to one or more of the various CYPs, from the CYP families 1 - 3 [21].

3.1.2 Stereochemistry and chirality

Stereochemistry is the study of relative arrangement of atoms in molecules, and part of that is stereoisomers [22]. Stereoisomers are molecules with identical atoms or

groups of atoms that have the same sequence of bonds but are oriented differently in space.

Chiral molecules are molecules that cannot be superimposed (become identical with) on its mirror image by any translation, rotational, or conformational changes. The geometric property is called chirality [22].

3.1.3 Aromaticity

Aromatic compounds are a large class of chemical compounds that have particular electronic, structural, or chemical properties. They are characterized by one or more planar rings of atoms, i.e. rings of atoms in the same plane, joined by two different kinds of bonds. These particular bonding arrangements cause a certain number of electrons within a molecule to be held more strongly. The unique stability of these compounds is referred to as aromaticity and is associated with low reactivity [23]. An example can be seen in Figure 3.1.

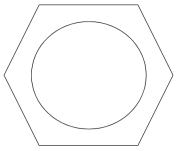


Figure 3.1: Aromatic molecule

3.1.4 Molecular representations in software: SMILES

SMILES (Simplified Molecular Input Line Entry System) is a chemical notation language designed for chemical information processing [24]. It denotes a molecular structure as a two-dimensional graph using ASCII (American Standard Code for Information Interchange) characters representing atoms and bonds. Two examples can be found in Figure 3.2.

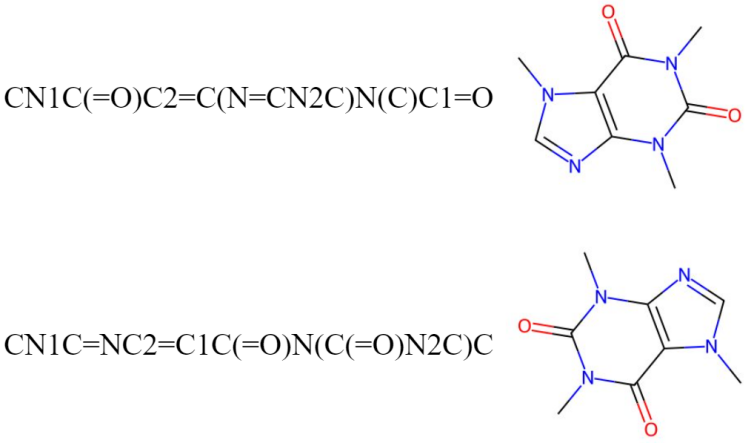
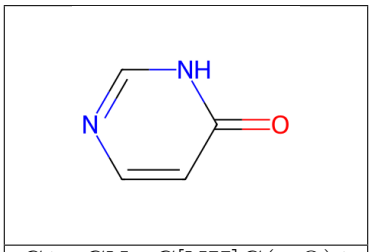


Figure 3.2: Different SMILES representations of a caffeine molecule.

In simple terms, atoms are represented using atomic symbols and single, double, triple, and aromatic bonds are represented by the symbols -, =, #, and :, respectively. Branches are specified by enclosures in parenthesis and cyclic structures are represented by breaking one bond in the ring and indicate the ring closures by matching digits appended to symbols [24].

Atoms in aromatic rings are specified by lower case letters, e.g. aromatic carbon is represented by lower case **c** [25]. Explicit aromaticity depiction can be avoided by entering the Kekulé form instead (see Figure 3.3) [25].



C1=CN=C[NH]C(=O)1
c1cnc[nH]c(=O)1

Figure 3.3: Representing the molecule in SMILES with (top) or without explicit aromaticity (bottom).

SMILES can optionally denote chirality [25] (see first example in Figure 3.4). SMILES that describe the molecular graph with atoms and bonds, but no chiral or isotopic information, are known as "generic SMILES", in contrast with "isomeric SMILES" that include isotopic information[25]. See examples in Figure 3.4.

The SMILES representation for a single molecule can be written in different ways, depending on the starting node for traversing the molecular graph[25]. This is exemplified in Figure 3.2, where both SMILES correspond to caffeine. There exists a canonicalization algorithm to generate a single generic SMILES among all valid possibilities, known as unique SMILES[25].

Input SMILES	**Unique SMILES**
[CH3][CH2][OH]	COO
C-C-O	COO
OC(=O)C(Br)(Cl)N	NC(Cl)(Br)C(=O)O
ClC(Br)(N)C(=O)O	NC(Cl)(Br)C(=O)O

Figure 3.4: Molecules represented in SMILES before and after undergoing canonicalizing algorithm.

3.1.5 Molecular fingerprint similarity

Molecular similarity is a concept to identify compounds with similar properties or structures. Morgan Fingerprint allows computing a molecular structure and generating a descriptive representation including atomic connectivity, atomic charge, and

chemical bond type [26]. Tanimoto similarity is a measurement used in molecular fingerprint comparison. It measures the overlap of structural features in chemical compounds and gives a value between 0 - 1, with closer to 1 indicating higher structural similarity [27].

3.2 Machine learning

This section explains the methods and resources used for the machine-learning aspects of the thesis.

3.2.1 Transformer

The deep learning model named Transformer was proposed in 2019 by Vaswani, Shazeer, Parmar, *et al.* [28]. Since its release, it has been applied within the field of natural language processing for tasks such as text translation, summarization, and generation [29]. A key feature of the model is its ability to capture relations between a sequence of words, also called tokens. It uses an attention-based mechanism to find features and relations. This feature is similar to the recurrent neural network (RNN) structure, although the Transformer allows parallel processing of the input sequences whereas the RNN does not. This, in turn, speeds up the training process while still capturing key relations between sequences [30].

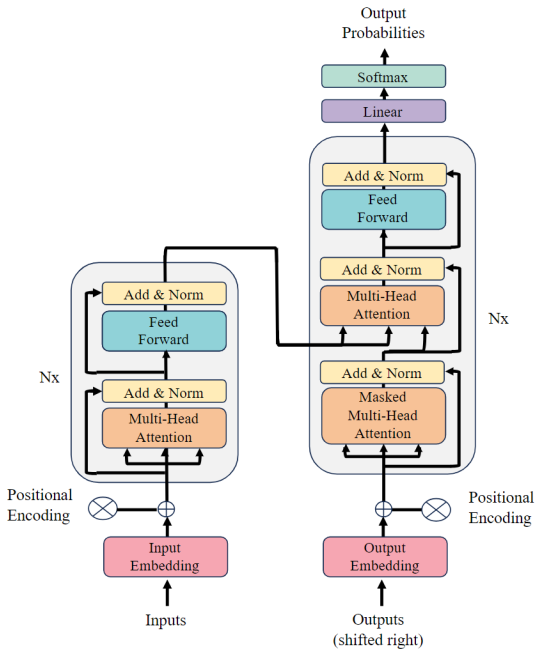


Figure 3.5: The Transformer - model architecture.

The Transformer architecture consists of two modules: an encoder, and a decoder (see Figure 3.5). Inside both modules exist N_x stacks of hidden layers containing the model's learnable parameters. The two modules focus on different aspects where the encoder aims to find relations present in the input and output sequence. The

decoder, however, focuses on a generational aspect, where it learns relations for token arrangement in the output sequence. Together, the encoder guides the decoder by sharing the input context such that the decoder output is related to the input.

The input of the Transformer is a text sequence that becomes tokenized into numeric values which the model can interpret. More about the tokenizer can be read in Section 4.3.1. The tokenized sequence will pass through each layer in a module and at some point the attention layer. Depending on the number of heads in the attention layer, the input is split and distributed equally among the heads.

This attention layer consists of a series of computations involving three components: the key (K), query (Q), and value (V) vectors (see Figure 3.6). The key vector represents information on the stored tokens. The query vector represents what tokens should be attended to, and together with the key vector, they compute the attention weights. These weights determine the distribution of tokens to be used in the final computation alongside the value vector, which holds the actual data values assigned to the tokens.

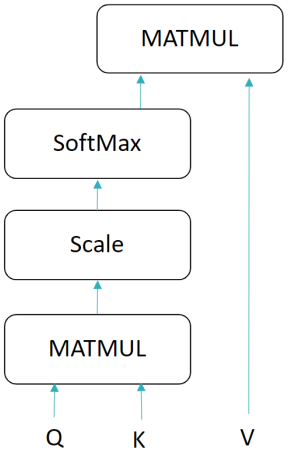


Figure 3.6: Attention layer computing Scaled Dot-Product Attention.

A critical part of the computation is a scaling operation, not shown in Figure 3.6 but present in Equation 3.1. The scaling factor is $\sqrt{d_k}$, where d_k represents the dimensionality of the key (and query) vectors. Dividing by $\sqrt{d_k}$ normalizes the values before applying the Softmax function, thereby preventing unstable gradients.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \tag{3.1}$$

The attention layer allows parallelization by dividing the matrices into several slices, performing the same calculations on each slice, and finally concatenating the results. This parallelization is referred to as *multi-head attention* and otherwise, *single-head attention*. Both the encoder and decoder make use of attention mechanisms. However, the decoder has an additional modified attention layer, called the *masked attention layer*, which can hide parts of a sequence [28]. When the decoder learns

the features of a new token, the already generated tokens are shown to the model, and all succeeding tokens remain hidden. This forces the decoder to only take prior generated tokens into consideration. The output of the decoder is a matrix consisting of probabilities for each possible token in all positions in a sequence. This information can later be used to generate target sequences based on an input by using generation strategies, which typically is an algorithm that uses the outputted matrix to build a probable target sequence.

3.2.2 Focused Transformer

A variation of the transformer model is the Focused Transformer (FoT) created by Tworkowski, Staniszewski, Pacek, *et al.*[10]. The model was created as a plug-and-play extension of the large language model named *Llama* [31], created by Meta®, which uses the Transformer architecture. A key feature of the FoT is that it aims to improve the context length of the Llama model when dealing with multiple documents, meaning it aims to handle larger context sizes and still keep high accuracy in token retrieval.

In regards to multi-document context, the FoT achieves a higher accuracy for token (key, value) retrieval compared to the *Llama* model, and the authors noted that a reason for this was due to the distraction problem [10]. The Transformer architecture is not incentivized to distinguish between keys with closely related meaning. This problem scales with the number of documents as they could include more tokens with similar or the same token structure. Comparing the FoT to Llama, Llama is limited to $2k$ tokens whereas the FoT could work with $100k$ tokens and retrain a correct token retrieval accuracy of 94.5% [10].

The FoT has two main techniques for improving its performance. During training, it allows the memory attention layers $l \in \mathcal{L}$ to access additional context, retrieving information about the preceding (keys, values) from its local context window where it attends to tokens. These (key, value) pairs are stored in an external memory, which can be fetched when needed, extending the context lengths. Figure 3.7 showcases this concept. To fetch the data from its external memory, the attention layer uses a top-k algorithm to find the k most probable tokens [32].

The second technique is crossbatch training, which is inspired by *contrastive learning* where the model is exposed to both positive and negative samples [33]. The idea of crossbatch training is that the attention layers learn from a mix of relevant and irrelevant information. Attention layers are exposed to the previous local context of a document, described as *positives* as they contain relevant information, and $d - 1$ context from unrelated documents, termed *negatives* [10]. It aims to improve the quality of representation for tokens.

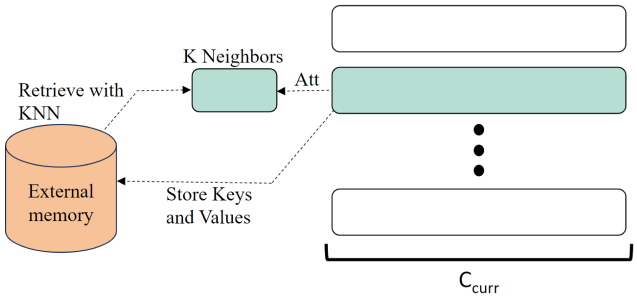


Figure 3.7: Showcasing how the Focused transformer retrieves key, values from its external memory. C_{curr} represents the tokens in the current context and *Att* the attended document.

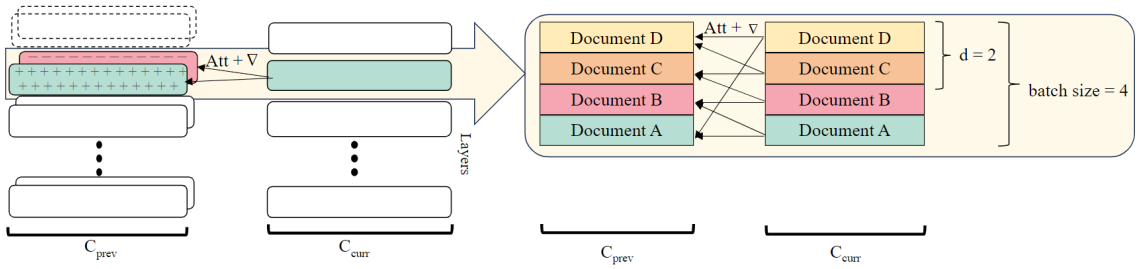


Figure 3.8: An illustration of how the Focused Transformer learns using positive and negative documents during its inference.

The FoT framework includes two variants that act as wrappers to the FoT, targeting different tasks. One specializes in sequence classification and the other in sequence generation. The sequence generation model is designed to generate text by predicting one token at a time, taking into account the context from previously generated tokens [34], [35]. This approach is applicable for text generation, where the output is not predetermined. Instead, the model learns to generate the most probable sequence based on a given input and the previously generated context. This allows for dynamic text generation. Additionally, the FoT includes a configuration class for customizing the model structure [10]. Parameters in the configuration class related to its structure are:

- Hidden layer size which specifies the dimensions of the hidden representations found in the input and output of the Transformer.
- Number of hidden layers that sets the number of Transformer blocks. A Transformer block consists of all the internal layers that process the input sequence and can be stacked on top of each other.
- Intermediate layer size which specifies the dimension of the hidden layers within the feed-forward network of each Transformer block.
- The number of attention heads present in the attention layer.

3.2.3 Tokenizer

A Transformer expects inputs to be in numeric format since the architecture relies on numerical values for its internal computations [34]. Therefore, when training a model on text or other non-numerical data, it is necessary to convert these inputs into numerical representations. This conversion process involves mapping the desired token characters to unique numerical values. Typically, this mapping is achieved using a tokenizer, which is a mechanism that maps a vocabulary of tokens to numerical indices. Once the Transformer completes its internal computations, the resulting numerical outputs can be easily converted back into tokens for interpretation [34].

3.2.4 Ensemble model

An ensemble model is a collection of multiple models, called base models, that predict a similar outcome. An ensemble model can be composed of two to several base models, often referred to as ensemble members. There are two components involved: A) the selection of ensemble members and B) the combination of the ensemble member predictions into an ensemble model prediction [36].

3.2.5 Beam search

Beam search is a heuristic search algorithm that explores a graph through a limited set of the most probable nodes, an optimization of the best-first search. Best-first search is a graph search that orders the partial solutions according to some heuristic. In beam search, only a predetermined number of best partial solutions are kept as candidates [37].

Beam search is used as a sequence decoding strategy, and can be seen as balancing the efficiency of greedy search and the optimality of exhaustive search. It is characterized by a hyperparameter, the beam size k. At the first step, the k tokens with the highest predicted probabilities will be chosen. At each subsequent time step, based on the previous choices, the k candidate output sequences with the highest predicted probabilities are chosen [38].

The process of the beam search algorithm is illustrated in Figure 3.9. It is assumed that the output vocabulary contains the elements $y = A, B, C, D$, the beam size is two, and the maximum length of the sequence output is three. The tokens at the first step (y_1) with the highest conditional probabilities $P(y_1|c)$ with the particular circumstance c are B and D, which the beam search chooses. In the next step, $P(B, C|c)$ and $P(D, A|c)$ have the highest conditional probabilities and are chosen. These steps continue until the maximum length ($L = 3$) is reached, or until a stopping criteria is achieved. The chosen output sequence should maximize the Equation 3.2, where L is the length of the final candidate sequence [38].

$$\frac{1}{L^\alpha} log\mathcal{P}(y_1, ..., y_L|c) = \frac{1}{L^\alpha} \sum_{t'=1}^{L} log\mathcal{P}(y'_t|y_1, ..., y_{t'-1}, c) \tag{3.2}$$

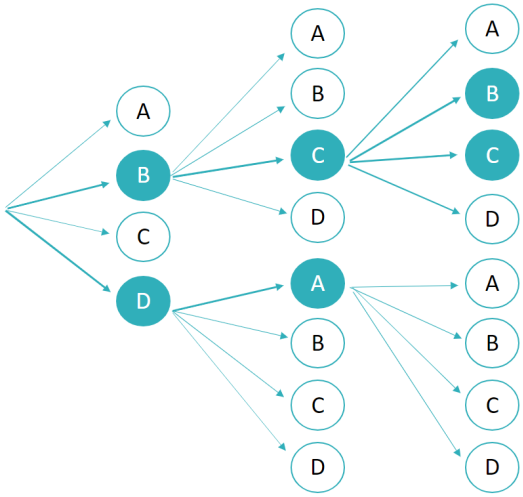


Figure 3.9: The choices made by beam search with beam size $k = 2$ and maximum length of the sequence output $L = 3$. The result found is $B-C-B$ and $B-C-C$.

3.2.6 Optuna: a hyperparameter optimization framework

Optuna[39] is an automatic hyperparameter optimization software framework. It is specifically designed for machine learning and has a define-by-run style user interface. It creates an optimization based on an objective function, where the goal is to minimize the function's return value. This is done through trials, where the suggested methods of the trial generate hyperparameters within the given bounds (see Listing 3.1).

```
def objective(trial):
    batch_size = trial.suggest_int('batch_size', 4, 64)
    loss = train(batch_size)
    return loss

study = optuna.create_study()
study.optimize(objective, n_trials = 100)
```

Listing 3.1: An example of a simple Optuna setup with batch size as a variable and the loss as the objective criterion.

4

Methods

This chapter covers the two main parts of the project. Section 4.1 details the data sources used, with Section 4.2 describing the data processing, and Section 4.3 explaining the applied machine learning approaches. In addition, we detail the evaluation process in Section 4.4. The details regarding the hardware used are described in Section 4.5

4.1 Data sources

This thesis primarily relies on the data sources MetXBioDB and DrugBank, as described in Section 4.1.1 and 4.1.2, as well as the Matched molecular pairs (see Section 4.1.5). The evaluation of the model are using data from the source GLORYx, described in Section 4.1.6.

4.1.1 MetXBioDB: metabolite reaction database

MetXBioDB [40] is a publicly available collection of over 2000 experimentally confirmed biotransformations derived from literature. In addition, data points were gathered from publicly available databases like DrugBank [41], PharmGKB [42], and SuperCyp [43]. It was created for use in the prediction tool Biotransformer [8]. Each biotransformation includes a starting reactant, a reaction product, the enzyme catalyzing the biotransformation, and the type of reaction. The starting reactants in the database consist primarily of xenobiotics such as drugs, pesticides, and toxins. It also includes a small set of endogenous molecules. Overall, out of the over 2000 biotransformations, around 1500 are cytochrome P450-catalyzed phase I reactions (with around 800 unique starting reactants) and around 600 phase II reactions (with around 500 unique starting reactants). Around 50 metabolic conversions originate from the human gut microbial metabolism [8].

4.1.2 DrugBank

DrugBank [41] is an online public database part of the Metabolomics Innovation Centre [44], a non-profit metabolomic core facility located at the University of Alberta. Its database can be accessed online for academic purposes. DrugBank contains information about drugs and drug targets. This includes information such as sequences, structures, and discovered pathways. The data set includes information on

drug reactions and drug metabolites. In total, there are approximately 4,000 drug-metabolite pairs. DrugBank provides additional information about its metabolite entries, such as the generation in which metabolites are formed, and the catalyzing enzyme, if applicable (e.g., Cytochrome P450)[41].

4.1.3 HMDB: human metabolome database

HMDB [45] is a database containing detailed information about small molecule metabolites found in the human body. It contains around 220,000 metabolite entries, both water-soluble and lipid-soluble ones. The project is supported by the Canadian Institutes of Health Research, Canada Foundation for Innovation, and the Metabolomics Innovation Centre (TMIC).

4.1.4 ChEMBL

ChEMBL [46] is a database of bioactive molecules with drug-like properties that have been manually curated. The content is from published bioactivity data, from seven Medicinal Chemistry journals: e.g. Journal of Medicinal Chemistry and Bioorganic & Medicinal Chemistry Letters [46].

4.1.5 Matched molecular pairs: virtual analogs

A database of virtual analogs [47] was created in an effort to explore and understand the bioactive chemical space [48]. The creators computationally explored the bioactive or drug-like space by a systemic design of analogs. Compounds of more than 1000 protein targets were gathered from CHEMBL [46] and divided into series of structurally related compounds, with core structures and substituents isolated. Combinations of the core-substituents were created, resulting in virtual analogs. The active compounds, sharing the same core structure as the virtual analog, created a data point [48]. The result was around 1 300 000 virtual analogs [47].

4.1.6 Test data set from GLORYx

Bruyn Kops, Sícho, Mazzolari, *et al.* created, for the prediction tool GLORYx [11], a manually assembled test set from scientific literature. Molecules undergoing metabolism were selected from a pool of top 100 best-selling drugs from 2018. From these, small-molecule drugs (with low molecular weight) made up of only organic atoms (H, C, N, S, O, F, Cl, Br, I, and P) were chosen and their human metabolites were extracted from scientific literature. The data set consists of 37 molecules with 136 first-generation metabolites.

4.2 Data processing

The machine learning models used metabolic data from the primary sources MetXBioDB [40] and DrugBank [41], as well as the matched molecular pair (MMP) data set [47]. The process of going from raw data to input suitable for the machine learning models is termed data processing. This section provides an overview of each component and the subsequent sections discuss these in greater detail.

Data extraction The collection of the metabolic data, MMP data, and the GLORYx test set (Section 4.2.1).

Data curation The curation of the chosen data, where data that would be invalid or inaccurate was excluded (Section 4.2.2).

Data split The split of metabolic data into a training, validation, and test data set (Section 4.2.3).

Data augmentation The data augmentation that was performed on the metabolic data set (Section 4.2.4).

Data for ensemble model The split of data for the strategy ensemble model (Section 4.2.5).

Data analysis An analysis of the metabolic data, including molecular weight distributions, enzymes catalyzing the reactions, and fingerprint similarity (Section 4.2.6).

4.2.1 Data extraction

The metabolic data were extracted from MetXBioDB [40] and DrugBank [41]. We performed the extraction of MetXBioDB data by downloading the latest updated version (Accessed Feb 2024)[40]. In this data set, molecules are represented using InChI (International Chemical Identifier) keys, which are a string-based identifier for chemical substances. We extracted the InChI of the parent and metabolite molecules and converted them to SMILES using RDKit [49]. The resulting data were also populated with drug names, enzymes catalyzing the reactions, and reaction types.

We extracted the data from DrugBank [41] by downloading the latest DrugBank data set available (version 5.1). The complete database file from DrugBank included the reaction pairs (structured as parent/child pairs) and the enzyme(s) for the reaction. The pairs were encoded as internal DrugBank identifiers (IDs) and were then used as keys to find the remaining information needed. This information includes SMILES, InChI, drug names, and catalyzing enzymes, which the other files had information about. If the internal DrugBank IDs did not correspond to accessible data, we used the names of the molecules instead. We crosschecked the drug, metabolite, external data sets, and external database HMDB (version 5.0) [50] to fill in incomplete data entries to create a final data set. The entire extraction is visualized in Figure 4.1.

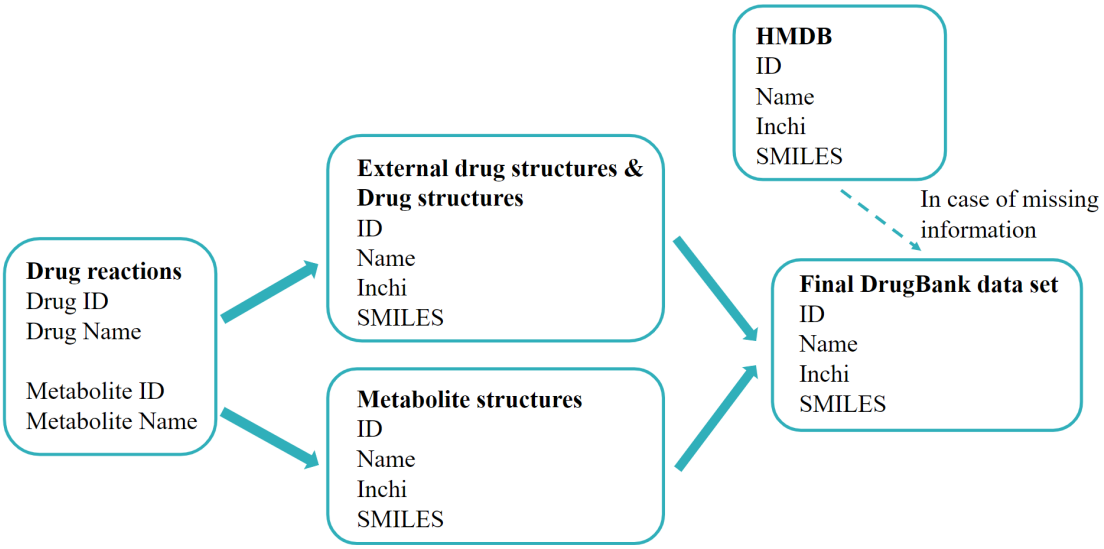


Figure 4.1: Representation of how the DrugBank data was combined together. The data from HMDB supplemented missing data where required.

We combined the data extracted from MetXBioDB[40] and DrugBank[41], excluding duplicate parent/child pairs. The result was a data set of 3116 parent and child pairs, referred to as the *metabolic data set*. The source of the metabolic data set can be seen in Figure 4.2, where the data points derived from both sources are indicated.

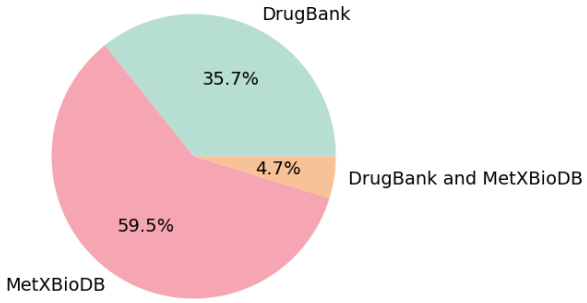


Figure 4.2: Data sources of the metabolic data set. The entire data set comprises around 3116 entries.

For the pretraining of the model, we downloaded a previously curated database of MMPs [47]. The virtual analogs were extracted in combination with their ChEMBL compounds. For the purpose of pretraining, the virtual analog were labeled as the "parent" molecules, and the ChEMBL compounds were labeled as the "metabolites." The SMILES for the ChEMBL compounds were gathered from ChEMBL v33 [46]. By doing so, we were left with a data set of close to 1.1 million parent/child pairs

(see Figure 4.3). This data set was used to pretrain our model, and is referred to as the *MMP data set*.

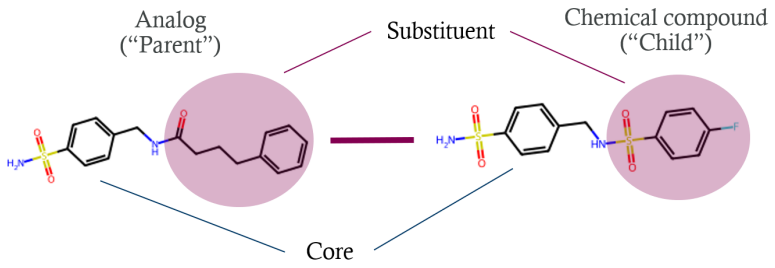


Figure 4.3: Example of a data point from the MMP set

An out-of-distribution (OOD) test set was used for evaluating all models. This was extracted from the previously created GLORYx [11] test set. The original drugs, along with their first generation metabolites, were extracted and used to create separate entries in the data set. Each data entry includes the SMILES of the parent/child molecules, the names of parent/child molecules, and the metabolite generation. The resulting data set contained 37 unique drugs with 136 first-generation metabolites, represented as 136 parent/child pairs, referred to as the *OOD test set*.

4.2.2 Data curation

In order to use the most appropriate data for training, the aforementioned data sets were curated and filtered based on certain criteria. The first criterion was that the SMILES of both the parent and child molecules were valid SMILES, to make sure that proper chemical structures were computationally handled.

The second criteria was that all molecules in the data set must be organic molecules. This means that only molecules with atoms commonly found in organic molecules [12] were included. The element types that were allowed were C, N, S, O, H, F, I, P, B, Cl, Br, and Si.

The third criterion was that all molecules must have a molecular weight between 100 - 1000 Da, similar to the curation performed bySícho, Stork, Mazzolari, *et al.* [12]. The molecules which were outside of this range were removed from the data set.

The fourth criterion was analyzing the fingerprint similarity between the parent and its metabolites using the Tanimoto similarity score. Relations with a score of 1 were excluded for the whole data set, being both MetXBioDB and DrugBank. Additionally, for the DrugBank data set, relations with a similarity score lower than 0.15 were excluded.

To make sure that the DrugBank data set included only xenobiotics, we performed another step of curation. The criteria for the reaction pairs was that the parent needed to have a *drug ID* (e.g., DBXXXXX) in contrast to a *drug metabolite ID*

(e.g., DBMETXXXXX). If the parent had a drug metabolite ID, it was only included if it existed as a metabolite to a drug, or if it could be traced back to a drug, that had a product/medicine assigned to it.

We cleaned the molecules by removing possible mixtures or salts [8]. The molecules were standardized by removing explicit stereochemistry and aromaticity representation and canonicalizing the SMILES , using RDKit [49]. For example, the change in the SMILES for the molecule Caffeine can be seen in Figure 4.4.

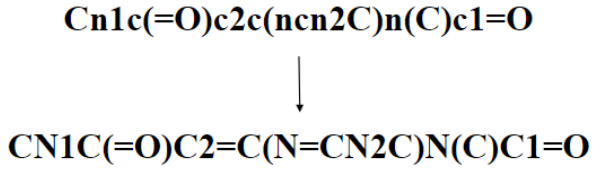


Figure 4.4: Example of SMILES standardization, shown here for Caffeine.

Finally, we excluded four reactions pairs from the DrugBank data that were invalid. These data points had *glucuronic acid* as their metabolite, which does not fill our criteria. The reason being that the acid acts as a conjugation handle (what attaches to the metabolite) rather than the metabolite itself. Thus, those four rows were removed.

4.2.3 Data splitting

We used a data split that simulates a true "out of distribution" (OOD) set. This means that the set can have data that deviates from the rest of the data. This was done by having no overlapping parent molecules between the different splits. For this, we used the GroupShuffleSplit function from sklearn [51], which gives each data point a class label and makes sure classes are in the same split. The class labels in this case are the SMILES of the parent molecules.

The split ratio was 80:10:10 for training, testing, and validation set. These data sets are referred to as the *metabolic train set*, the *metabolic validation set*, and *metabolic test set*. The metabolic test set was removed first, while the validation and the training set were split during each run.

4.2.4 Data augmentation

We used data augmentation to artificially increase the data by creating modified copies using our limited set of training data. In the metabolic train and validation data set, there are data points of a drug with its metabolite and that metabolite having its own metabolite and so on, giving several generations of metabolites that create a chain (see example in Figure 4.5). With this, a new data point was created with the drug molecule and the second generation metabolite. Furthermore, a

mapping was created with the first and the third generation metabolites, and continuously downwards. Augmenting the data in the metabolic data set in this way, with the metabolic test set left out, resulted in 1162 new parent/child pairs. This data set is referred to as the *augmented data set*. The similarity criterion described above was also applied during the selection of the augmented data set.

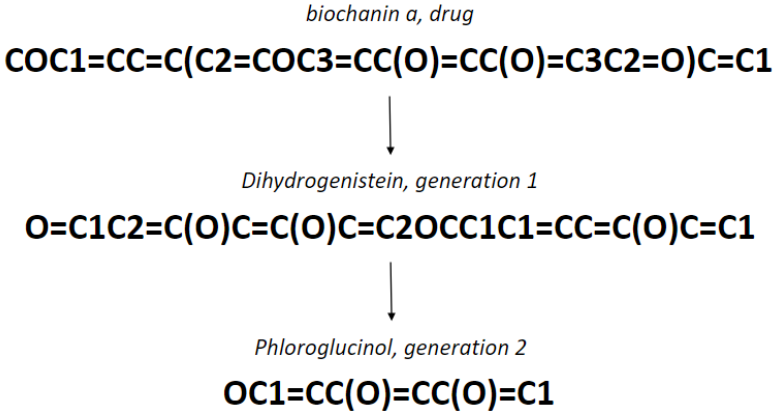


Figure 4.5: Example chain of metabolism for Biochanin A, spanning two generations.

4.2.5 Ensemble model data

We created the data sets for the ensemble model from the metabolic data set. The data set was split into two, based on whether the source is DrugBank or MetXBioDB. The data points that originate from both were split between the two data sets evenly, with identical parents ending up in the same splits. The third data set used was the augmented data set. These three data set were the ones created for the fine tuning of the models in the ensemble model.

4.2.6 Data analysis

The metabolic data set contains reactions that are mediated by different enzymes. The distribution of enzymes in our metabolic data can be seen in Figure 4.6, where almost 70 % of the reactions are mediated by enzymes included in the CYP450 families. Note that this analysis was only performed for the subset of reactions (82 %) which included enzyme information.

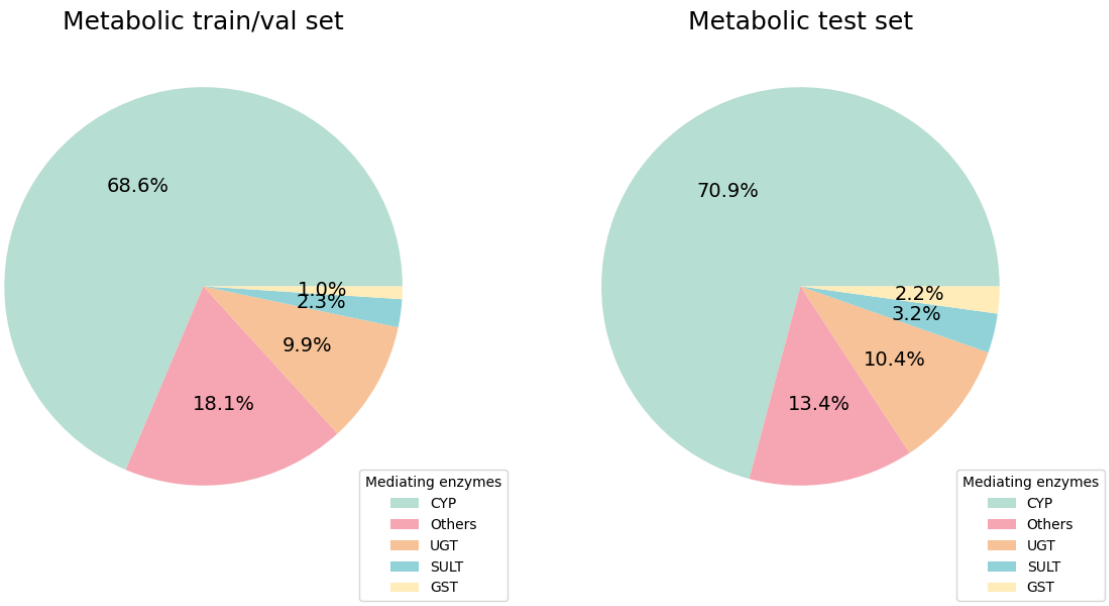


Figure 4.6: Enzyme distribution for the metabolic train/val set and the metabolic test set. It does not include the OOD test set, the augmented data set, or the MMP data set.

The molecular weight (MW) distribution of the parent and child molecules can be seen in Figure 4.7. Note that the parents' MW distribution is based on unique parents only. The distribution of MW differences between each child and its parent is illustrated in Figure 4.8.

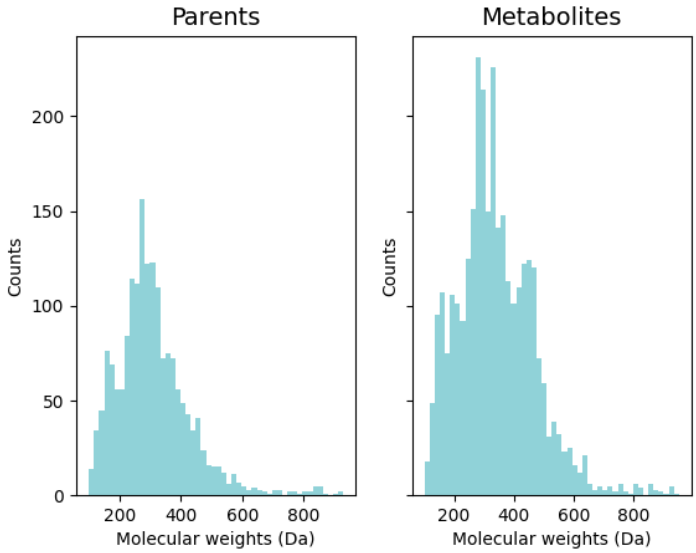


Figure 4.7: Molecular weight distribution in Daltons for all metabolites and parents in the metabolic data set.

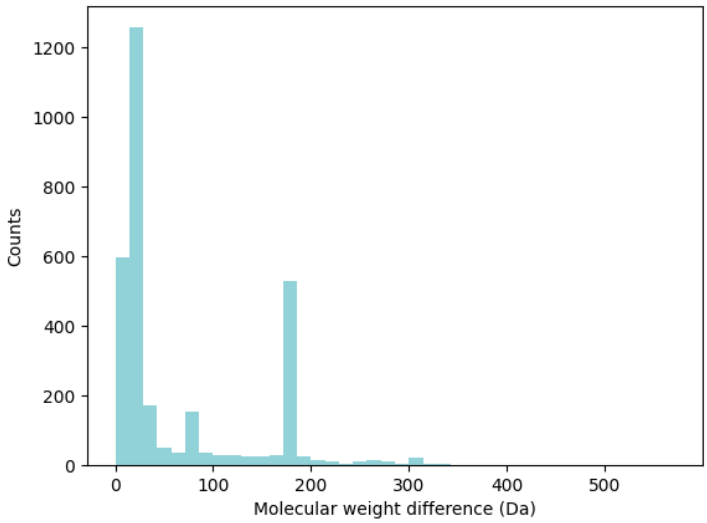


Figure 4.8: Molecular weight difference in Daltons between metabolites and their parents in the metabolic data set.

The Tanimoto fingerprint similarities between metabolites and their respective parents are binned and visualized in Figure 4.9. The average fingerprint similarity between metabolites and their parents is 0.72.

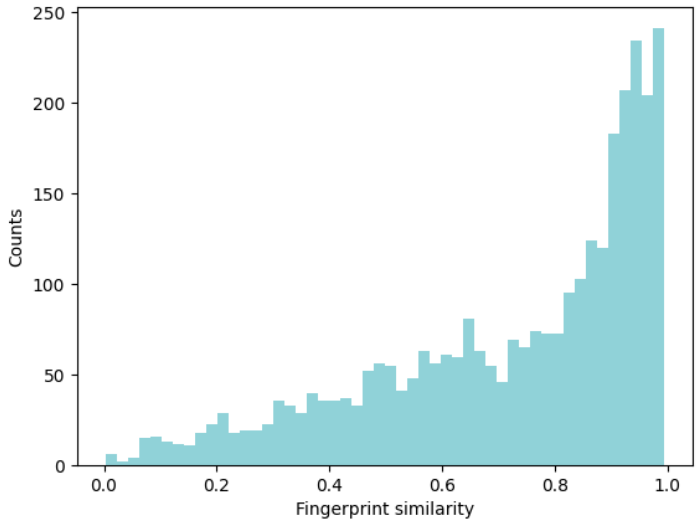


Figure 4.9: Fingerprint Tanimoto similarity between metabolites and their parents for the metabolic data set.

4.3 Machine learning framework

The following section details the structure of the deep learning framework used for the project, along with relevant modules connected to it. The framework can be described as a pipeline where data and information get sent through different modules, ending with the predicted metabolites in the SMILES format. A graphical representation of this pipeline can be found in Figure 4.10. This section will conclude with the hyperparameter configuration chosen.

Figure 4.10: A graphical representation of the model pipeline.

There are four possible models that can be explored:

- The first version is a model that is trained on the metabolic train set. This version is referred to as the *baseline model*.
- The next version is a model that is trained on the combined metabolic and augmented data set. This version is referred to as the *augmented model*
- The third version is a model that is pretrained on the MMP data set, and then finetuned on the metabolic train set. A graphical representation of the entire process can seen in Figure 4.11. The model after pretraining is referred to as the *pretrained model*, and the final finetuned is referred to as the *finetuned model*. See more detail in Section 4.3.3.
- The fourth version is a model that is pretrained on the MMP data set and finetuned into three different models. Each finetuned model is given different parts of the combined metabolic and augmented data set. The predictions of all three models are then combined into what we refer to as *ensemble model*. More details can be seen in Section 4.3.4.

The preprocessed SMILES string for the parent molecule is given to the tokenizer, a module that encodes the sequence of SMILES into integers that the model can interpret (see Section 4.3.1). Given a tokenized sequence, the model will return a matrix of probabilities for the possible metabolite tokens. In other words, the model will predict the probability for each token in each position in the sequence. Using the returned matrix of probabilities, the SMILES sequence will be computed using a specific sequence generation strategy, described in detail below in Section 4.3.5. Once a sequence of tokens is sampled for each predicted metabolite, the generated 'metabolites' are validated in the postprocessing step. The resulting SMILES is the final output of the model.

4.3.1 Tokenizer

The tokenizer used is the *fastLlamaTokenizer* taken from the HuggingFace library [34]. It inputs a sequence of SMILES and encodes them into integers the model

can interpret. These SMILES tokens are listed in a vocabulary and consist of the possible SMILES tokens such as atoms (e.g., 'C', 'Si'), bonds, and other features (e.g., '(', '['). Special tokens are used for language modeling to mark the start and end of the sentence token ('<s>' and '</s>') and padding tokens that are ignored during training ('<pad>').

4.3.2 Application of Focused Transformer

The FoT version we chose was *LongLlamaForCausalLM* by Tworkowski, Staniszewski, Pacek, *et al.* [10], adapted for sequence generation. The model allows modification of its hyperparameters and layers (see Section 3.2.2). The hyperparameters used that were connected to this model are its hidden size, intermediate size, number of hidden layers, and number of attention heads [10].

The input for the model is a batch of tokenized parents with the input target being a tokenized child of the parent. The output from the model is a matrix with the shape (batch size, vocabulary size, sequence length) containing probabilities for every token in every position in the sequence length. It is possible to input several pairs at the same time, decreasing computation time and getting a more generalized output [34].

- **batch size** the number of input pairs.
- **vocabulary size** the amount of possible tokens.
- **sequence length** the amount of tokens in a sequence.

In the training phase, these outputs were used to calculate the loss and modify the model parameters. That meant additional components/parameters needed to be defined: a start learning rate, top learning rate, optimizer, scheduler, and a warmup period. The optimizer used for all training was *Adam* [52]. The learning rate, top learning rate, warmup period, and scheduler were used to change the learning rate and were optimized using Optuna [39]. The schedulers used can be found in the Pytorch library [52] and were: *exponential* and *linear*. Each has its own set of hyperparameters.

4.3.3 Pretraining

The model is pretrained using MMP data set [47]. The pretraining data set is used to train the model in correctly understanding SMILES annotations and chemical transformations syntax, before the finetuning on metabolic data. The pretraining and finetuning are done in almost an identical way, with the exception that the pretraining has a bigger batch size.

The resulting model from the pretraining was preloaded and finetuned using metabolic data, resulting in a finetuned model. A graphical explanation of the process can be found in Figure 4.11.

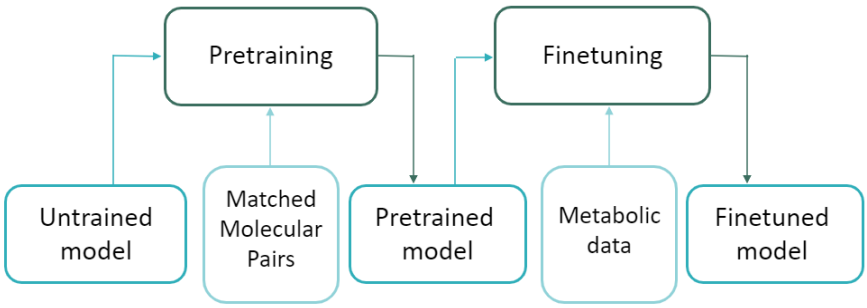


Figure 4.11: Pretraining process

4.3.4 Ensemble model

An ensemble model consists of several models whose results were combined together. The pretrained model, as seen in Section 4.3.3, is taken and finetuned on three different data sets. The three data sets are the DrugBank data set and the MetXBioDB data set with the duplicates split between them evenly, as well as the augmented data set. Each model is separate from each other and their training does not affect any of the other models. The result is an ensemble model consisting of three models that are finetuned on different data sets. A graphical explanation of the process can be found in Figure 4.12.

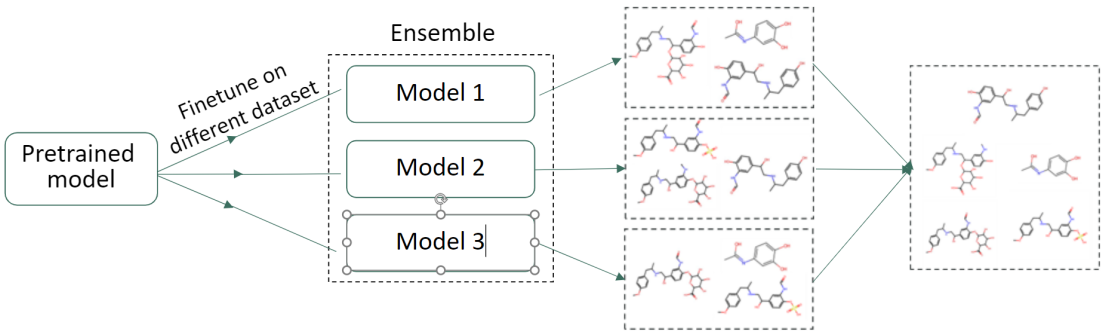


Figure 4.12: Process of creating ensemble model

The ensemble model generates the predicted metabolites by each ensemble member separately, together with the probabilities. These metabolites are ranked by interlacing the metabolites in order based on the probabilities. If two or more identical metabolites are given, the one with the highest probability is kept. This result is returned as the prediction of the ensemble model.

4.3.5 Sequence generation strategy

The sequence generation strategy used was beam search, which uses the output probabilities of the model to generate sequences of metabolites in the SMILES format.

Beam search enables the generation of several different metabolites for a given parent molecule. The beam search can penalize longer sequences, but this option was not used.

A stopping criterion was implemented which signals that the beam is done and no more tokens need to be added. The stopping criterion is either the end of the sentence-token (</s>) or the padding-token (<pad>). If the end of the sentence-token is given, the beam is finished and only padding is added from then on to keep the sequence length between the beams even. The stopping criterion is also used to make it possible to stop the generation earlier, if all beams are done.

The beam sizes are adjustable and can generate different results. Another parameter is n-best, which chooses how many of the beams are returned. For example, beam size = 5 and n-best = 3 means that 5 beams are explored but only the best 3 are returned in the end.

4.3.6 Postprocessing

The output sequence of SMILES tokens that are created through predictions are postprocessed, thus ensuring that the valid results are returned. The first check is that the prediction is a valid SMILES sequence. The molecular weight of the molecule is also required to be between 100 to 1000 Da and only organic molecules are allowed, reflecting what is done in data curation (see Section 4.2.2).

4.3.7 Optimization of hyperparameters

The hyperparameters detailed in Section 4.3.2 were optimized using Optuna [39]. The optimization process was performed with two runs with different types of schedulers: linear and exponential. The hyperparameters concerning linear and exponential runs, together with settings, are detailed in Table 4.1.

Different target areas were optimized, these being the training loop and the model itself. In the training loop, the hyperparameters can be altered for the same model to better suit pretraining or finetuning. However, the hyperparameters inside the FoT must remain in the same configuration for pretraining and finetuning, since it otherwise would corrupt the internal weights of the model.

Optuna used these hyperparameters and tried to optimize an objective function set to minimize the validation loss. The objective consisted of setting up a new training session with the same condition except for the set of hyperparameters which Optuna will modify. The value the objective function returned was:

$$min(\text{Average validation loss for each epoch})$$

Based on the information given by Optuna, the top models were selected for full training on the metabolic train/val set. Each of these models was then evaluated for different epochs, to mitigate any overtraining occurring. Finally, the best performing model, at its optimal number of epochs, was selected as the baseline model. The same hyperparameters were used for the other three models; augmented model, finetuned model, and ensemble model.

Hyperparameter	Method & Selection range
Batch size	Discrete range, $[8, 32]$
Start learning rate	Continuous range, $[10^{-5}, 10^{-1}], log = True$
Top learning rate	Continuous range, $[Learning\_rate, 10^{-1}], log = True$
No. warmup steps	Discrete range, $[1, 20]$
Hidden size	Categorically, {64, 128, 256, 512, 1024}
Intermediate layer size	Categorically, {64, 128, 256, 512, 1024}
No. attention heads	Categorically, {2, 4, 8, 16}
No. hidden layers	Categorically, {2, 3 ,4}
Gamma (exponential)	Categorically, {0.7, 0.8, 0.9, 0.95, 0.98, 0.99}
End factor (linear)	Categorically, {1e-5, 1e-4, 1e-3, 1e-2}
Total iterations (linear)	Discrete range, $[1, (n\_epochs - n\_warmup\_epochs)]$

Table 4.1: The setup for optimization of the hyperparameters for Optuna, for linear and exponential scheduler. It shows the hyperparameters Optuna can modify (first column), and the method and value range that Optuna was allowed to modify (second column).

4.3.8 Chosen model architecture and hyperparameters

The final model architecture and hyperparameters can be found in Table 4.2. The baseline model performed best at epoch 45, the augmented model at epoch 35, the pretrained model at epoch 195, and the finetuned model at epoch 100. The same pretrained model was used for the ensemble models, and the ensemble models each performed best at epoch 60.

Hyperparameter	Value
No. attention heads	16
No. hidden layers	4
Intermediate layer size	512
Hidden layer size	256
Batch size	16
Pretraining batch size	128
Scheduler	Exponential
Start learning rate	$4.8 * 10^{-5}$
Top learning rate	$1.32 * 10^{-4}$
No. warmup steps	9
(Exponential scheduler) Gamma	0.95

Table 4.2: The chosen hyperparameters.

The resulting learning rates during the training are visualized in Figure 4.13.

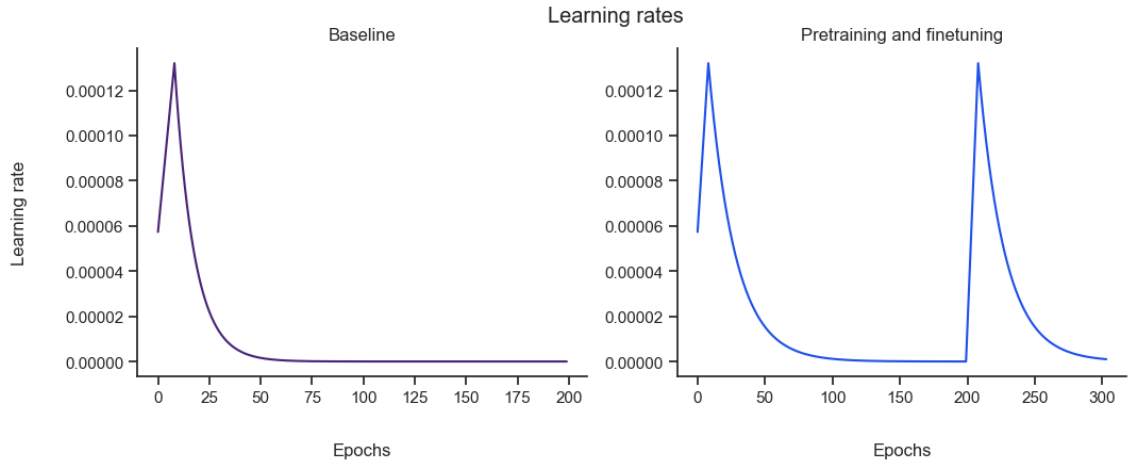


Figure 4.13: Learning rate for the baseline model, and the finetuned model.

4.4 Evaluation

The evaluation of the models was performed periodically. The models were evaluated continuously during the training, on a sample size of 100 data points for both the training set and the validation set. The final models were evaluated using the metabolic test set for comparison between different models. The evaluations were assessed using the top-n predictions. The models were additionally evaluated using the OOD test set, for comparison with other methods (see Section 4.4.1).

To evaluate the model, we let the model predict the metabolites of the unique parent molecules in the chosen data set. The predicted metabolites are then compared to the available metabolites of each parent. We controlled if the predictions were correct by canonicalizing the SMILES of both the predicted and the true metabolites and then comparing them. If the SMILES of one of the predicted metabolites match with any of the true metabolites of the molecule, then it is counted as a correct prediction. These correct predictions are referred to as *true positives*, while the incorrect ones are referred to as *false positives*. *False negatives* in this case are metabolites that are not found by the predictions.

Two of the evaluation methods used are recall and precision (see Equation 4.2, and 4.1). Additionally, the top-n accuracy was calculated for the predictions, which represents the percentage of metabolites that are correctly predicted given the n predictions for each molecule.

$$\text{precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}} \tag{4.1}$$

$$\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} \tag{4.2}$$

The validity of the processed metabolites were evaluated by examining the percentage of valid smiles over the predictions as detailed in Equation 4.3.

$$\text{validity} = mean(\frac{\text{valid smiles per molecule}}{\text{total predictions per molecule}}) \tag{4.3}$$

The mean string similarity of a metabolite and its best prediction was evaluated. This was done by extracting the predictions for a parent and performing string comparison for each metabolite of the parent, using Python's sequence matcher [53]. The most similar predictions per metabolite is found, and the mean string similarity are extracted.

Other evaluation methods were the percentage of molecules where either (a) at least one or (b) all of the metabolites were accurately predicted. The fingerprint similarity average between the correct and the predicted metabolites was also calculated.

4.4.1 Other methods used for benchmarking

In the final stages of the project, we benchmarked the best derived model in comparison with previously published metabolite prediction models. The models used in this benchmark were:

- Metatrans (Extracted 2024-04-22) [9],
- Biotransformer (Version 3.0, extracted 2024-04-22) [8],
- GLORYx (Results extracted from Bruyn Kops, Sícho, Mazzolari, *et al.*) [11],
- and SyGMa (Results extracted from Bruyn Kops, Sícho, Mazzolari, *et al.*) [11].

These models were all tested on the OOD test set, consisting of 37 drugs with a sum of 136 metabolites. The exception was GLORYx and SyGMa, where the results were extracted from the paper presenting GLORYx by Bruyn Kops, Sícho, Mazzolari, *et al.* [11]. MetaTrans required some processing on the evaluation data to fit the model input. MetaTrans offers tools to convert the data into its desired format. The Biotransformer offers different types of coverage for different enzyme families. For this benchmark of predicting human drug metabolites, it was set to cover human and human gut microbial transformations, labeled as *allHuman* in its settings.

4.5 Hardware details

Table 4.3 shows the graphics processing units (GPUs) the training sessions used for each model. The ensemble models utilized the pretraining for the pretrained model, before finetuning the model for the ensemble model. The CUDA driver version used for all runs was: nvidia-cudnn-cu11, version 8.7.0.84.

Model name	GPU
Baseline model	Tesla V100-PCIE-32GB
Augmented model	Tesla V100-PCIE-16GB
Pretrained model	Tesla V100-PCIE-32GB
Finetuned model	Tesla V100-PCIE-32GB
Ensemble model	Tesla V100-SXM2-32GB

Table 4.3: Table of what GPU each model was trained on.

5

Results

In this chapter, we will present the result of the thesis. The optimization of hyperparameters for the models is explored and presented in Section 5.1. In Section 5.2, the model's ability to learn from a smaller data set is evaluated. The performance of the four different models are presented in Section 5.3, evaluated on metabolic test set. The best performing model is then evaluated on the GLORYx test data set and compared to other methods previously created by other authors and presented in Section 5.4.

5.1 Finding the best hyperparameters

Optuna was used to find the most suitable training framework and model parameters. Two studies were performed using Optuna (see Figure 5.1). These two studies had identical hyperparameter ranges, except for their scheduler parameters. The most important hyperparameter for both studies was the top learning rate, with a significantly higher importance for the exponential scheduler than the linear. The start learning rate was a close second.

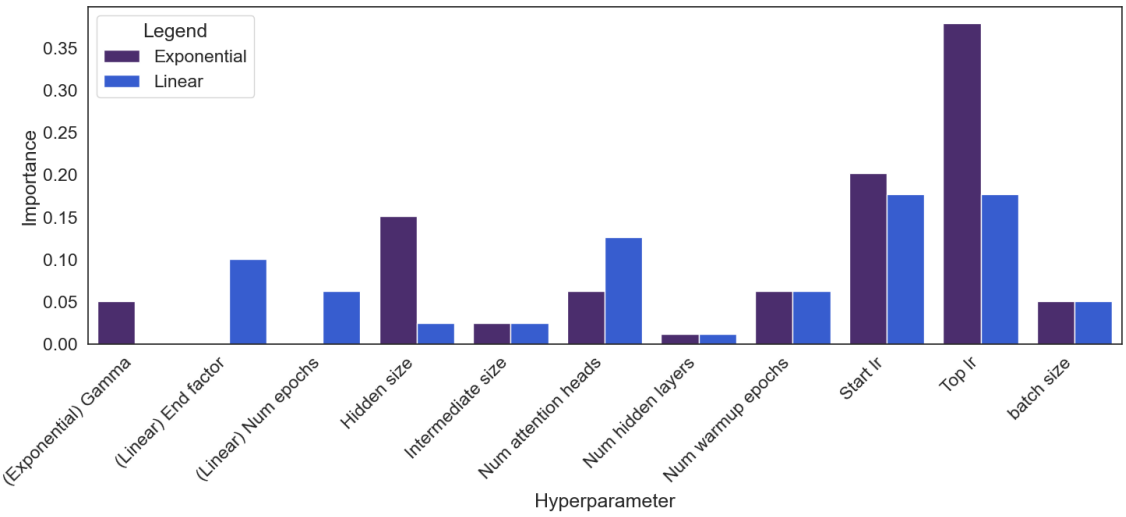


Figure 5.1: Hyperparameter importance calculated by Optuna for two studies: one with a fixed exponential scheduler, and another with a fixed linear scheduler.

The top 10 best-performing models originating from the exponential and linear Optuna studies can be found in Figure 5.2. The curves of the validation loss were approximately similar for different trials, where all reached minimum loss early, followed by a sharp increase after 10-15 epochs.

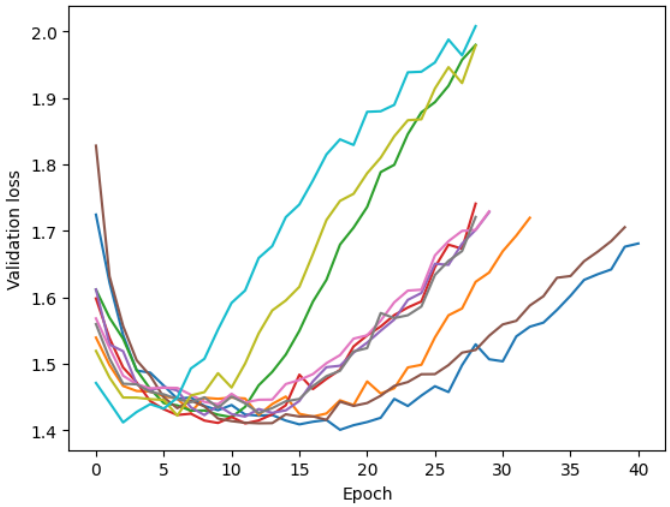


Figure 5.2: Validation loss of the top 10 models, originating from the exponential and linear studies.

5.2 Results from overtraining

A model containing all parent-metabolite pairs was allowed to train for an extensive period of time to enable maximum loss decrease. Another model was overtrained on the metabolic data set, but with only one randomly chosen metabolite per parent. These experiments show the difference in performance when the data set contain a one-to-one mapping. The training and validation results for two trials are depicted in Figure 5.3. The models were both quickly overtrained, after around 20 epochs.

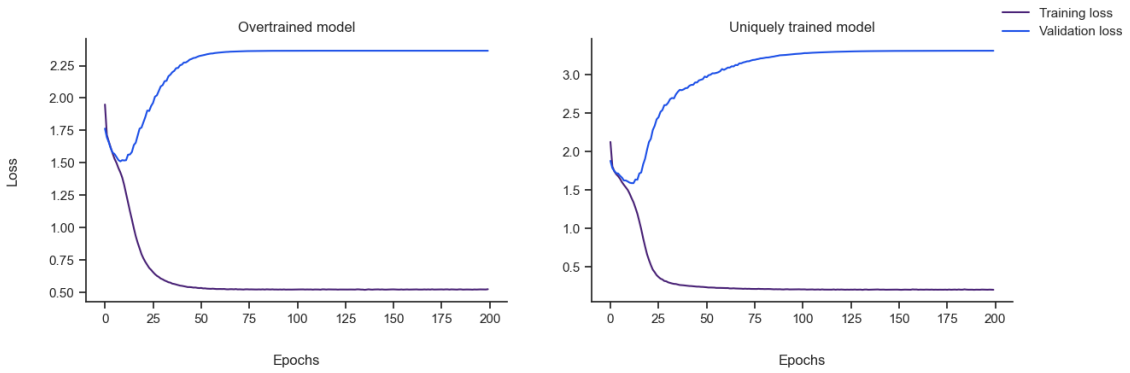


Figure 5.3: Observed training and validation loss for a model trained on the metabolic set (left) and with a unique metabolic data set containing one metabolite per parent drug (right).

The validity of the two models can be found in Figure 5.4. The predictions in the beginning are "CCCCC..." or similar outputs, which results in a higher validity.

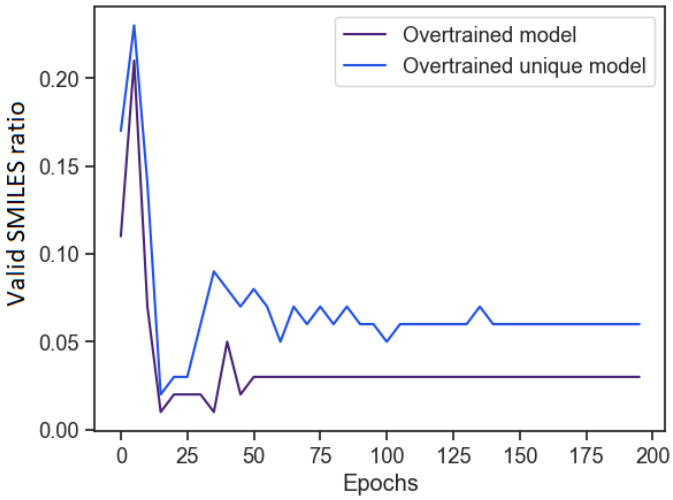


Figure 5.4: SMILES validity for two overtrained models. Validity was calculated using the training set.

The models were evaluated on a random sample of 300 data points from the respective model's training set. Note that part of the evaluation set could include datapoints from the validation set, meaning previously unseen data for the model. The results from evaluating the two overtrained models can be found in Table 5.1, where the unique model clearly outperformed the other overtrained model in all of the metrics.

	Top-1 accuracy	Top-10 accuracy	True positives	Validity	Mean string similarity
Overtrained model	9%	10%	8	15%	76%
Unique model	22%	30%	91	53%	89%

Table 5.1: Results from evaluating the two overtrained models. Each was evaluated on a sample of 300 data points from the respective model's training set.

5.3 Performance of the models

This section will present the performance of the four different models evaluated on the metabolic test set. These four models are: 1) the baseline model, 2) the augmented data model, 3) the finetuned model, and 4) the ensemble model.

In Figure 5.5, the training and validation losses of the models are shown. As one can see, the training and validation loss for the baseline model and augmented model reach around epoch 20 before diverging sharply, resulting in overfitting. The pretrained model also experiences overfitting, although not as sharp as finetuning.

For the finetuned and ensemble models, the observed loss when transitioning from pretraining into finetuning starts at a similar value as the start loss but shows more direct overfitting when the finetuning begins. Finetuning on models that was pretrained, showed a similar validation loss trend as those not pretrained.

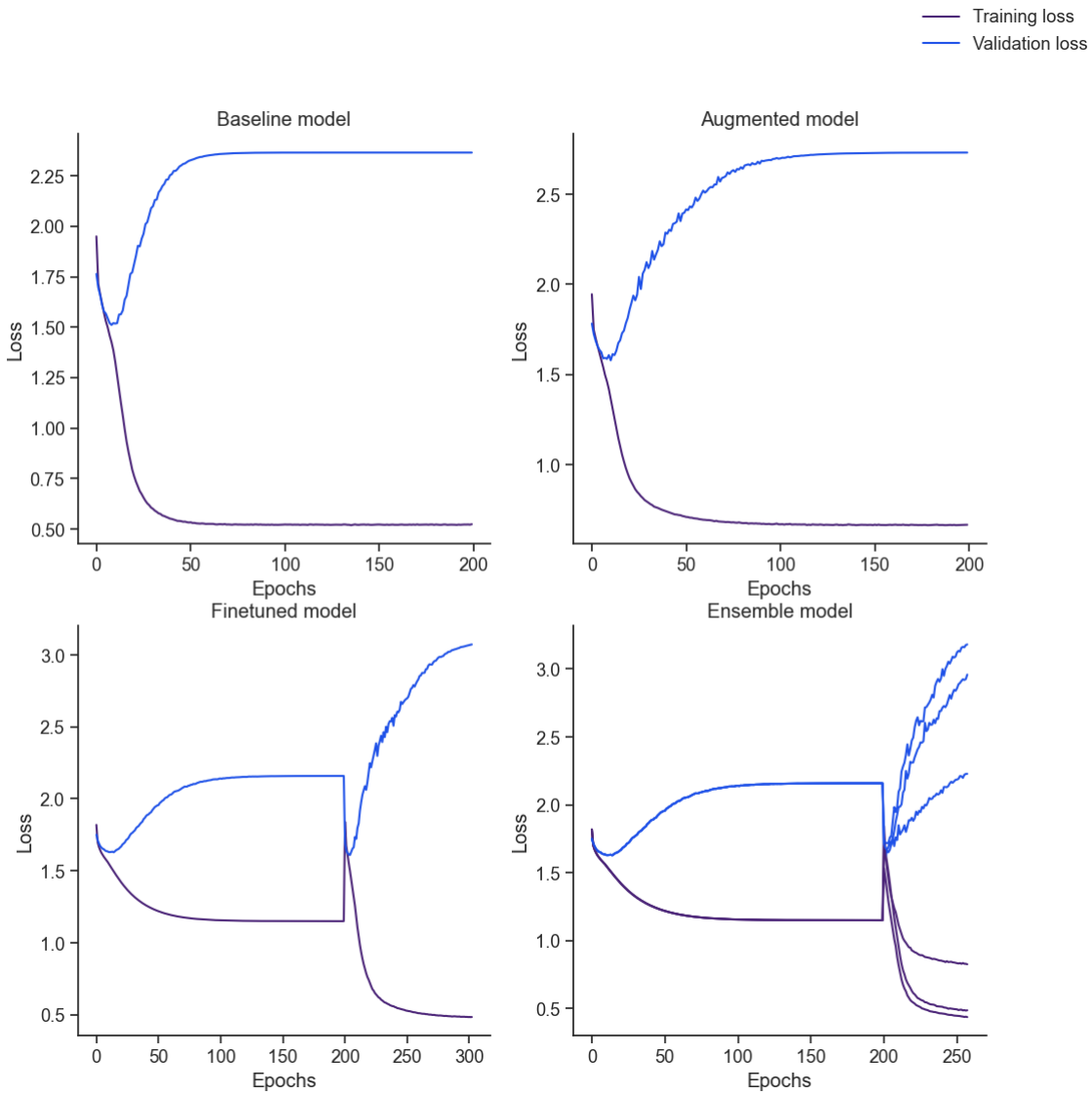


Figure 5.5: Training and validation loss for the training of the baseline, augmented, finetuned, and ensemble model.

Figure 5.6 and 5.7 show the change in validity of samples during the training. The initial validity during the first epochs is high for all four models. The models produce the same tokens indefinitely during the early training, i.e. 'CCCC' or 'OOOOO', which are valid SMILES. As the model continues to train, its ability to learn more token relations increases and it learns to produce more complex SMILES.

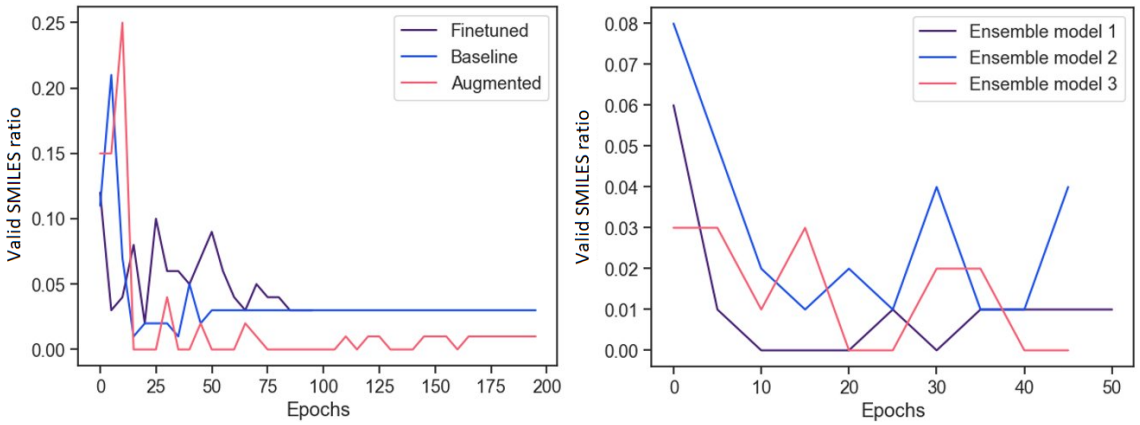


Figure 5.6: The validity of predictions on samples from validation set during training for, the baseline, the augmented, finetuned, and the ensemble model.

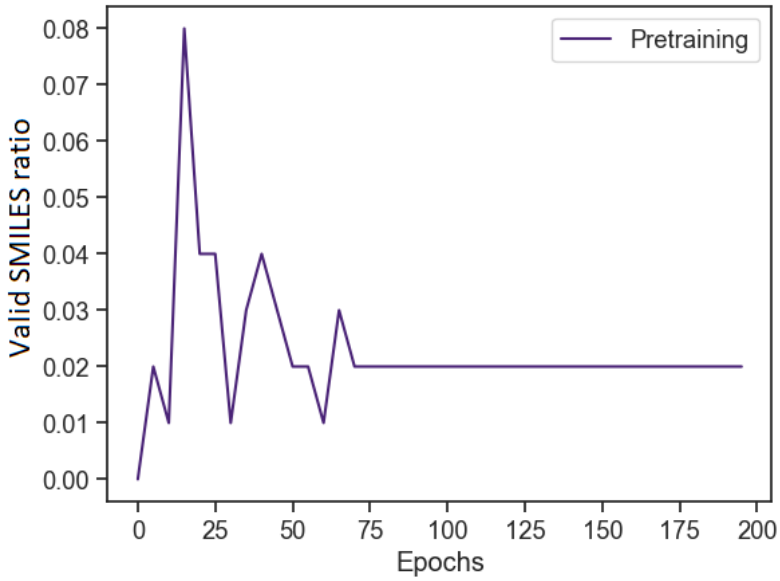


Figure 5.7: The validity of predictions on samples from the validation set during the training of the pretrained model, used for the finetuned model and the ensemble model.

The performance of the four models is summarized in Table 5.2, where the beam size and the n-best is 10. For the baseline model, the mean string similarity percentage between the top prediction and the closest metabolite is, on average, 63%. The same average was calculated for parent and child pairs, with low and high fingerprint similarity. Pairs that had a fingerprint similarity of $\leqslant 0.70$ and $\geqslant 0.90$, yielded a similarity average of 60% and 63% respectively. There is no pattern that can be found for true positives in relation to catalyzing enzymes.

	Top-1 accuracy	Top-10 accuracy	True Positives	Validity
Baseline model	0%	0.6%	1	4.5%
Augmented model	0%	0%	0	3.4%
Finetuned model	0%	0%	0	2.8%
Ensemble model	0%	0%	0	1.7%

	At least one metabolite	All metabolites	Mean string similarity
Baseline model	0.6%	0.6%	63%
Augmented model	0%	0%	57%
Finetuned model	0%	0%	55%
Ensemble model	0%	0%	48%

Table 5.2: The performance of the models. Top-1 accuracy used a beam size of 1, whereas the others had a beam size of 10 for their calculations.

The true positive of the baseline model is shown in Figure 5.8. One can see that the metabolite and parent are almost the same in this case, with a difference of one token.

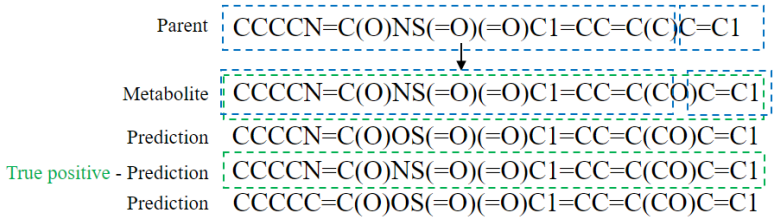


Figure 5.8: Baseline model's true positive. The parent molecule had one metabolite and got one true positive and four valid SMILES.

You can see an example of a molecule with several metabolites in Figure 5.9. We see here that as long as the parent and metabolite are the same, the prediction is correct. When they differ, the predictions are not as close. Figure 5.10 shows a similar example, but with several metabolites. Figure 5.11 shows an example of when the parent and metabolite differ more from each other, with a different starting token.

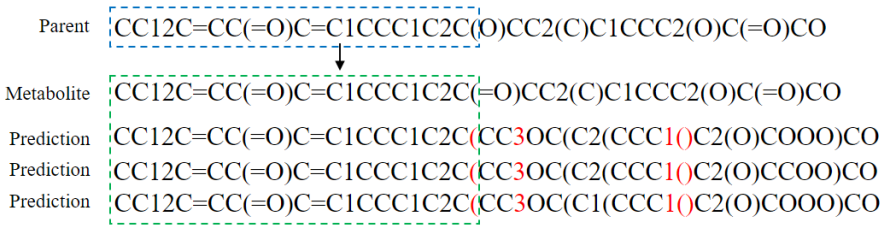


Figure 5.9: Baseline model's predictions of a molecule with one metabolite, resulting in no valid SMILES. The green square indicates correctly predicted tokens, and the blue square shows the tokens that reflect the parent.

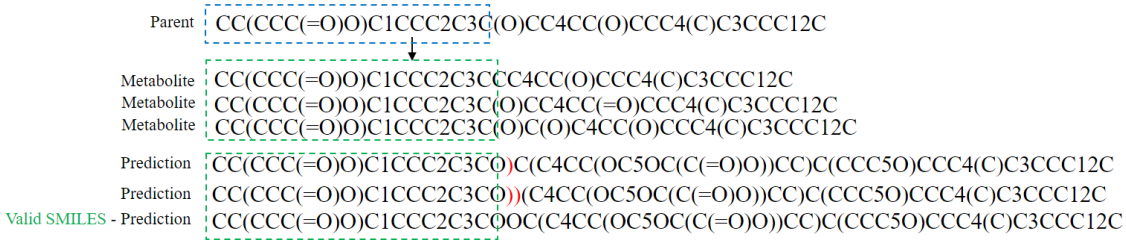


Figure 5.10: Baseline model's predictions of a molecule with several metabolites, with no valid SMILES and no true positives. The green square indicates correctly predicted tokens, and the blue square shows the tokens that reflect the parent. Tokens highlighted in red are notable examples of being incorrectly placed.

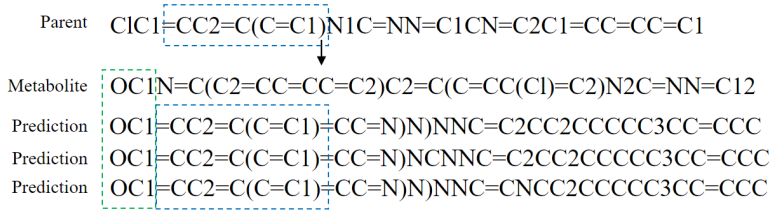


Figure 5.11: Baseline model's predictions of a molecule with one metabolite that differ from each other. The green square indicates correctly predicted tokens, and the blue square shows the tokens that reflect the parent

For the pretrained model, after the pretraining and before the finetuning, the model was evaluated on the metabolic test set. At that point, the model had a validity of 4%, and a top-n accuracy of 0%, and the total amount of valid SMILES was 21. One example that emulates how the predictions typically looked can be seen in Figure 5.12. The predictions were closely resembled to the format of the parent, at which point the metabolite could not be correctly predicted. This model also had several examples of predictions that are "CCCCC22CC11" or end with "111111....". These predictions can be compared to the evaluation of the model after being trained one epoch on the metabolic data set, when the predictions are "CCC..." or "====...".

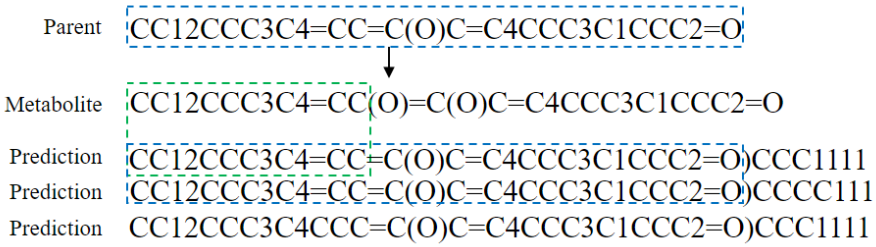


Figure 5.12: Pretrained model's predictions for a molecule with one metabolite, before finetuning, using a beam size of 10 and n-best 3. The blue square marks the identical token sequence.

5.4 Comparison to other methods

The best performing model, the baseline model, was compared to previous work in Table 5.3. Of the existing methods, the Biotransformer had the higher recall (54.8%), compared to only 0.7% recall by the best model in this thesis.

	Recall	Precision	Total no. predictions	No. correct
Baseline model	0.7%	0.0%	1244	1
Biotransformer	54.8%	12.7%	582	74
MetaTrans	52.6%	9.7%	721	70
GLORYx	77%	6.1%	1724	105
SyGMa	68%	12%	800	93

	At least one metabolite	All metabolites
Baseline model	2.7%	0%
Biotransformer	75.7%	21.6 %
MetaTrans	88.9%	16.7%
GLORYx	N/A	N/A
SyGMa	N/A	N/A

Table 5.3: Evaluation of other metabolite prediction models. The lower table indicates the percentage of parent molecules where A) at least one metabolite and B) all metabolites are accurately predicted.

6

Discussion

In this section we discuss the results and the choices made throughout the project, including data curation and model design.

6.1 Choices in data curation

Since the project's focus is understanding metabolism in the human body to aid drug development, the work does not contain the application of the model to non-human environments, and the proposed data set does not contain this type of data.

To ensure the model works as intended, the data should be representative of the aimed task. Since drugs are largely exogenous substances, compared to endogenous molecules produced by the body, the chemical compounds which were attributed with this feature were proceeded forward [15]. The non-organic molecules were filtered out as most pharmaceutical products are organic compounds. The molecular weight filtering was done to make sure that the focus was retained on small-molecule drugs. Too small molecules (MW $<$ 100 Da) were removed as these would not be optimal for a natural language model.

The data for the model was chosen to be in the SMILES format, since it is a well-used format of representing molecules on a computer. The SMILES also have clear information on the elements making up the molecules, where "C" means a carbon and does not change depending on what other atoms are in the molecule.

To make sure that there is a molecule-to-molecule mapping, any salts or mixtures were removed from the data set. The canonicalization, removal of stereochemistry, and removal of explicit aromaticity representation ("kekulization") were done to standardize the molecules and focus on the standardized syntax to enable model learning.

After removing stereochemistry and aromaticity, there were instances with parents and metabolites that had the same SMILES. These were identified using the Tanimoto fingerprint similarity score. To mitigate noise in the metabolic training set, these were removed. Data points in the DrugBank set with a score lower than 0.15 were also removed to mitigate too large transformations. This was not done in the MetXBioDB set since that data set has been rigorously curated and used in other works.

6.2 Choices in implementation of model

There are several variants of the Focused Transformer, which each serves different purposes. We chose the *LongLlamaForCausalLM* since it was designed for sequence generation, relating closely to this project. It allowed the generation of a prediction for each position in a sequence which we could then use to generate full SMILES using beam search. The training framework was designed to input the tokenized parent sequence along with the tokenized metabolite target. The idea behind it was to let the model learn the relation between the parent and metabolite. Thus, when prompted with an unseen drug, the model would have an understanding of how a possible metabolite could be built.

We chose the loss function, PyTorch implementation of the *CrossEntropyLoss*, for its suitability. The reason was that the model outputs a token for each position in the sequence, and the *CrossEntropyLoss* function compares the token of every position to the true value, computing the loss. *Adam* optimizer taken from Pytorch was used for its common use. Learning rate scheduling was implemented, to decrease the learning rate as the model loss nears its minimum. Using this allowed for speeding up the learning during early training and, as the weights become more tuned, lowering the learning rate to reach the local optimum. The two schedulers, the *linear* and *exponential*, were used for their simple structure. Adding a linear warmup period was also done due to its common use for deep learning models, ramping up the learning rate for faster loss convergence before it lowers to find the optimum.

6.2.1 Data split

We choose to split the data so that the parent molecules with all their metabolites belonged to the same split. This was done to let the data simulate a true ”out of distribution” data set, and give the model the possibility to train on several aspects and possible metabolites for each parent molecule. This let the validation and test set give a more unbiased result.

6.2.2 The generation strategy: beam search

We chose the generation strategy beam search since it is most commonly applied in similar tasks. It explores the model more than a greedy search, but it does not need the same amount of resources as exhaustive search. The beam search also gives the opportunity to return several predictions for each parent, which enables prediction of several metabolites. It also gives the model a higher number of chances to find the correct metabolite.

6.2.3 Pretraining

We made the choice to explore pretraining as a strategy, since the metabolic data set was smaller than the data sets commonly investigated in the natural language processing field, hence the pretraining would enable the model to learn the chemical transformation syntax better. The finetuning, where the model learns from the

metabolic data set, would then hone in on the metabolic transformations, without having to learn the chemical transformations at the same time.

The metabolic data set represents the metabolic transformation with a one-to-one molecule relation and it would be beneficial to use a data set that simulates the same one-to-one relation. The matched molecular pairs data set were chosen with this in mind, since a data point would be a one-to-one relation. The molecules in the data points have a difference in part of the molecule while the rest of the molecule are the same, resulting in a high molecular similarity, which is similar to the majority of metabolic data. There were also several analogs to each chemical compound, which also matches the metabolic data set.

6.2.4 Ensemble model

The ensemble model was chosen as a strategy to let the different ensemble members discover different aspects of metabolic transformations and, as such, find different metabolites for a molecule. The data split for the ensemble model was done in a way to let the models have a data set with different data sources; MetXBioDB, DrugBank, and augmented data. The idea was that the ensemble members could learn different aspects of the data.

We still performed the generating of the result with beam search with the same motivation as above. This, in combination with the ranking of the predictions based on the probability, was performed. A question we discussed was if a metabolite is included in several ensemble members' results, should the probability from the ensemble members be combined or not. The choice was to pick the highest probability of that metabolite and let the ranking be based on that. The reason was that there could be a risk if a metabolite originates from less common catalyzing enzymes, that metabolite could be disfavored compared to CYP-catalyzing reactions (that were the most abundant).

6.3 Ethics

The biochemical data being used are publicly available, so there is no issue with privacy or restrictions in any similar way. The data contains no information regarding the patient information (e.g., gender, ethnicity or age), therefore an artificial intelligence (AI) trained on this data should not pose any ethical concern compared to projects investigating other medical data.

An aspect to consider would be that a tool that predicts metabolites gives all kinds of metabolites, including potentially toxic ones. This could technically be misused, which should be taken into consideration.

6.4 Evaluation of performance

From the analysis on the hyperparameter importance by Optuna(see Figure 5.1), we noted that *top_lr* and *start_lr* were the most important ones. This was expected

as they affect how the weights adjust themselves to the data. Finding the suitable rate is, therefore, crucial so that the weights change at an appropriate rate. The hyperparameters related to the structure for the model were not as important which was surprising as the size of the model should affect its ability to learn.

During training, the validation loss seems to be biased toward a low loss early followed by a quick increase. The setting of the objective function could affect this behavior as it seeks to take the minimum value from the observed validation loss, which disregards any trend followed by it. It could be worth looking into alternative variables to minimize.

When we analyzed the validity of the two overtrained models, we noted that the model trained on a unique-parent set had a higher validity compared to the one with several metabolites per parent. We chose the ending epoch with the models producing the lowest training loss and evaluated them. The results showed that the uniquely trained model had better performance of the two in all of the metrics, which reveals that the model finds it hard to handle multiple metabolite structures per parent, thus making it difficult to generate valid SMILES (see Table 5.1).

As seen in Figure 5.5, all of the models overfit early when trained on metabolic data, as their training and validation loss diverge. We theorize this is due to the size of the metabolic data set that includes ~ 3000 pairs. When discussing language models, 3000 is a small number that does not allow the models to fully learn all of the features present in metabolic reactions. There might only exist a finite number of features the model could make use of in the training set before it starts learning too specific patterns only present in the training data. This reflects the evaluation as well, as 10% of the complete data set was reserved for evaluation resulting in ~ 300 data points. Due to its size, noise present in the test set can potentially become more prevalent. Therefore, further expanding the metabolic data set is something worth looking into.

For the four models created; baseline, augmented, pretrained, and ensemble model, it is noticeable that they are all struggling to generate valid SMILES and true positives (see Table 5.2). The baseline model, which was the best at generating valid SMILES, had several examples where it had a good start in generating valid SMILES but as the sequence continued to expand, errors started appearing (see Figure 5.9). It also appears that the predictions often resemble the parent more than the metabolite. A possible reason for the low SMILES validity and poor accuracy could be that during the training the model's loss was calculated for a pair of parent and metabolite and not for the complete set of metabolites. This could cause the model to perform misleading calculations of the loss when exposed to new pairs of data where it instead attempts to predict the prior seen metabolite for the same parent. Thus, the model would be considered wrong even though it might have made a correct output. This problem can be seen in table 5.1 where the model trained on unique pairs significantly outperforms the regular model. Figure 5.10 shows an example where every true metabolite shares the same start sequence, which the prediction was able to follow. Though, as soon as the three metabolites starts to diverge in structure, the predictions starts generating errors. The current loss calculations

might therefore make the model confused and could be further investigated.

Focusing more on the data set the model is trained and tested on, the model's performance in generating the true metabolite shows only a slight difference when comparing parent-metabolite pairs with high fingerprint similarity to those with low similarity. This suggests that fingerprint similarity between the parent and metabolite was not the key factor for achieving true positives. Additionally, the parent's enzyme family does not appear to correlate with the accuracy of metabolite predictions.

An insight we gained from the result is that the beam search did not seem to consider the prior tokens when predicting the next one. This results in the model generating invalid SMILES, as one can see in Figure 5.10, where one of the issues was a closing parenthesis predicted without an opening parenthesis present in the previous sequence. We believe that the issue arises because the model produces a single probability matrix on the input sequence. Beam search computes the next tokens based on this matrix, without considering the previous generated token sequence. This behavior allows beam search to select the closing parenthesis, since for that position and input, that token could be the most probable one. If the current generated sequences had been passed into the model as well, then its context would have been considered, thus increasing the probability of selecting the correct token. In the way the model was trained, this was not possible. Since the model is a decoder-only model, it can only receive one input at a time, and receiving the generated sequence would remove the context of the parent molecule. We trained the model such that the decoder input and output were the parent SMILES and its metabolite SMILES respectively. To combat this issue and make it possible to both consider the context of the parent and the generated token sequence, a different data handling for the model could be beneficial. A data handling where the model can incrementally give predictions based on parent and generated tokens, which we detail more in Section 6.5 *Future works*.

The baseline was, with a slim margin, the best-performing model on the metabolic train set. While evaluated on the GLORYx data set, it was only able to find one true metabolite (see Figure 5.3). The Biotransformer and MetaTrans were evaluated in an identical way as the baseline and performed better on all metrics while also requiring fewer predictions. Although the baseline model performed subpar on the OOD, it showed some promise, as it had the ability to predict drug metabolites from an unseen OOD data set. More methods could be explored which would improve its ability to generate valid SMILES and also correct metabolites.

6.4.1 Performance of strategies

We explored several strategies to improve the performance of the model. Using an augmented data set, on top of the metabolic data set, resulted in a slightly lower performance than the baseline model with no true positives (see Table 5.2). This was unexpected since exposing the model to more training data, augmented from the original training set, should be beneficial since it allows the model to learn more about valid SMILES. As previously mentioned, there could be an issue with training

the model with multiple metabolites. Since the augmented data set is constructed to have more metabolites per parent, it could be related.

As one can see in the results, the finetuned model performed slightly worse than the baseline and augmented models. With the lowest validity and still no true positives, the finetuned model with the current configuration did not achieve the result that we had expected. As we can see in Figure 5.5, the model overfitted almost directly after the model started finetuning on the metabolic data set. The reason could be that the model has already learned the chemical format and the metabolic data set is too small for the model.

We noticed slightly better performance previously, with another hyperparameter set, where the finetuned model was able to produce true positives. This indicated that the optimization of the hyperparameters on solely the baseline model did not fully translate to the pretraining. It could be beneficial to have a different learning rate for the pretraining than for the finetuning. The finetuning may also need different hyperparameters than when solely trained on the metabolic data set.

As one can see in Figure 5.12, the predictions of the pretrained model prior to finetuning are clearly better compared to the untrained model at finding some resemblance of molecules. It could not find the exact metabolite, but it was capable of finding valid SMILES and giving predictions that were within the right vicinity. It seems that the model was able to partly learn the chemistry format in SMILES. We came to the conclusion that the strategy of pretraining using the MMP data set was promising, but that it did not perform fully in the current approach.

The ensemble model had a similar result compared to the pretrained model. Most of the predictions that were valid were also less correct. For example, it would include a long string of carbon atoms ("CCCCCCC") or other parts that showed a clear structural difference from the metabolites. The issues previously explained regarding the pretrained model were prevalent in the ensemble model as well. The optimization of the hyperparameters could be performed on the entire ensemble model as well, to find a better set more optimal for that set-up.

A path that could be explored for an ensemble model is dividing the data set based on the catalyzing enzymes, to create a chance for one base model honing in on the aspects of reactions catalyzed by less common enzymes. To explore this part, we reason that the data sets would need to be extended with more data from non-CYP catalyzing reactions.

6.5 Future work

Since the result of the deep learning models were based on the available data, the process of gathering more metabolic data would be one of the future steps to look into. There could be publicly available data that we did not consider, as well as proprietary data available within pharmaceutical companies that could be curated and used for the model. This would further strengthen the model for common

transformation reactions, but also expand the applicability domain for non-CYP-mediated metabolic conversions.

Another aspect of the study that we were not able to fully explore was the comparison between the Focused Transformer and the original implementation of the transformer. The tool MetaTrans does use a transformer, but the authors considered a different metabolic data set than us. So, exploring differences in performance between a transformer and the FoT with our data set would add more insights into the effects of the FoT and the aspects that could be improved and better adapted to the domain.

A more in-depth analysis of the available hyperparameters for the framework could bring further insights. Although the Focused Transformer was investigated, certain hyperparameters were set to default values. This project centered on parameters affecting the model's architecture, though other parameters influencing additional aspects of the model may also be considered. More effort in tuning the hyperparameters could also lead to better outcomes, hence future efforts could focus on exploring Optuna's objective function. As of now, Optuna performs independent training sessions, where only the hyperparameters are varied across runs. Optimization based on validation loss serves a purpose, as a lower validation loss generally indicates that the model learns effectively and can generalize to unseen data. However, finding a more stable model would be beneficial and achieving a low validation loss does not necessarily mean that the model is optimizing for its end goal, which is generating valid SMILES strings and predicting correct metabolites for a drug. A potential improvement would be to incorporate additional metrics that assess the quality and accuracy of SMILES generation and metabolite prediction, alongside validation loss.

Additionally to the new settings for Optuna, performing optimization of parameters on pretraining combined with finetuning could be beneficial to find hyperparameters that better fit that case. In this regard, one could look into having different hyperparameters between finetuning and pretraining relating to the learning rate.

A future path to explore is to treat the input and output data in a different way. In the current implementation, the parents are treated as the input and expect the metabolites as the output. Instead, we could treat the data in the way one would treat text generating for both training and result gathering, where the parents are given as the prompt and the metabolite is the answer. To illustrate this, the parent and the starting token would be given to the model and then the first token from the metabolite would be expected; then the parent, starting token and the first token would be treated as the input and the next token would be expected. This is performed in steps, so that when the generation of predicted metabolites for a parent is performed, the model yields predictions based on the parent and the previously chosen token. This could potentially solve the issue with the performance of the model and give a better result. This aspect would be our first step to explore if there was an opportunity to continue this work.

7

Conclusion

In the field of drug metabolite prediction, there exists a gap in *in silco* techniques, where many tools are enzyme-focused and dependent on derived transformation rules. Deep learning models have the potential to overcome these dependencies by identifying unseen connections through a complex network with trainable parameters. This project aimed to build a deep neural network capable of predicting drug metabolites based on the deep learning model *Focused Transformer* derived from Tworkowski, Staniszewski, Pacek, *et al.* [10]. The project resulted in a well-curated metabolic data set, available for metabolic predictions. The second result were several models, where the best one was able to generate one true positive and had a chemical validity of 4.5%.

The Focused Transformer has the potential for handling molecular strings despite displaying worse accuracy than existing methods for metabolite prediction. The approach shows promise in learning chemical syntax from metabolic data and therefore predict valid SMILES. This was demonstrated via overfitting experiments, where the Focused Transformer was able to generate valid metabolite predictions for seen molecules, and in some cases even correct predictions. Additionally, it was observed that pretraining the model using the matched molecular pairs was feasible and displayed considerable potential. Moreover, as matched molecular pair analogy was reminiscent of the metabolic data set where many metabolites were structurally similar to parent drugs. Therefore, we also discussed future directions for this work and potential improvements that could aid in achieving more predictive machine translation models for the prediction of drug metabolites.

Bibliography

[1] Merriam-Webster, *Metabolite*, https://www.merriam-webster.com/dictionary/metabolite, Accessed: 2024-05-03.

[2] T. E. of Encyclopaedia Britannica, *Reduction (chemistry)*, https://www.britannica.com/science/reduction-chemistry, [Online; accessed 7-May-2024], 2022.

[3] Study.com, *Polar molecule*, Website, URL: https://study.com/learn/lesson/polar-molecule.html, Year.

[4] S. Susa, A. Hussain, and C. Preuss, "Drug metabolism," *StatPearls [Internet]*, Jan. 2024, [Updated 2023 Aug 17]. Available from: https://www.ncbi.nlm.nih.gov/books/NBK442023/.

[5] J. Kirchmair, A. H. Göller, D. Lang, *et al.*, "Predicting drug metabolism: Experiment and/or computation?" *Nature Reviews Drug Discovery*, vol. 14, no. 6, pp. 387–404, 2015.

[6] D. A. Smith, Ed., *Metabolism, Pharmacokinetics and Toxicity of Functional Groups: Impact of the Building Blocks of Medicinal Chemistry on ADMET* (Drug Discovery). Royal Society of Chemistry, 2010, p. 544, ISBN: 978-1-84973-016-7. DOI: 10.1039/9781849731102.

[7] G. A. N. Gowda and D. Djukovic, "Overview of mass spectrometry-based metabolomics: Opportunities and challenges," in *Mass Spectrometry in Metabolomics: Methods and Protocols*, D. Raftery, Ed. Springer New York, 2014, ISBN: 978-1-4939-1258-2. DOI: 10.1007/978-1-4939-1258-2_1. [Online]. Available: https://doi.org/10.1007/978-1-4939-1258-2_1.

[8] Djoumbou-Feunang, Yannick, J. Fiamoncini, *et al.*, "Biotransformer: A comprehensive computational tool for small molecule metabolism prediction and metabolite identification," *Journal of Cheminformatics 11 (1): 2*, 2019.

[9] E. E. Litsa, P. Das, and L. E. Kavraki, "Prediction of drug metabolites using neural machine translation," *Chemical Science:12777-12788*, 2020.

[10] S. Tworkowski, K. Staniszewski, M. Pacek, Y. Wu, H. Michalewski, and P. Mio, *Focused transformer: Contrastive training for context scaling*, 2023. arXiv: 2307.03170 [cs.CL].

[11] C. de Bruyn Kops, M. Sícho, A. Mazzolari, and J. Kirchmair, "Gloryx: Prediction of the metabolites resulting from phase 1 and phase 2 biotransformations of xenobiotics," *Chemical Research in Toxicology*, vol. 34, no. 2, pp. 286–299, 2021. [Online]. Available: https://doi.org/10.1021/acs.chemrestox.0c00224.

[12] M. Sícho, C. Stork, A. Mazzolari, *et al.*, "Fame 3: Predicting the sites of metabolism in synthetic compounds and natural products for phase 1 and phase 2 metabolic enzymes," *Journal of Chemical Information and Modeling*, vol. 59, no. 8, pp. 3400–3412, 2019. [Online]. Available: https://doi.org/10.1021/acs.jcim.9b00376.

[13] S. Phang-Lyn and V. A. Llerena, "Biochemistry, biotransformation," *StatPearls*, 2020, Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/31335073.

[14] A. Blanco and G. Blanco, *Medical Biochemistry*. Elsevier, 2022, pp. 1–892. [Online]. Available: https://doi.org/10.1016/C2020-0-02932-4.

[15] S. Phang-Lyn and V. Llerena, "Biochemistry, biotransformation," *StatPearls [Internet]*, Jan. 2024, [Updated 2023 Aug 14]. Available from: https://www.ncbi.nlm.nih.gov/books/NBK544353/.

[16] Editors of Encyclopaedia Britannica, "Enzyme," *Encyclopedia Britannica*, Jan. 2024. [Online]. Available: https://www.britannica.com/science/enzyme.

[17] Editors of Encyclopaedia Britannica, *Anabolism*, https://www.britannica.com/science/anabolism, Jul. 2019.

[18] T. E. of Encyclopaedia Britannica, *Hydrolysis*, https://www.britannica.com/science/hydrolysis, [Online; accessed 7-May-2024], 2022.

[19] S. M. Kerwin, *Medicinal Chemistry: Principles and Practice*, 2nd, F. D. King, Ed. Cambridge: Royal Society of Chemistry, 2002, pp. xxvii + 448, Second Edition Edited by Frank D. King (GlaxoSmithKline, UK), ISBN: 0-85404-631-3. [Online]. Available: https://doi.org/10.1021/ja025346j.

[20] K. Shankar and H. Mehendale, "Cytochrome p450," in *Encyclopedia of Toxicology (Third Edition)*, P. Wexler, Ed., Third Edition, Oxford: Academic Press, 2014, pp. 1125–1127, ISBN: 978-0-12-386455-0. DOI: https://doi.org/10.1016/B978-0-12-386454-3.00299-2. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780123864543002992.

[21] M. Zhao, J. Ma, M. Li, *et al.*, "Cytochrome p450 enzymes and drug metabolism in humans," *International journal of molecular sciences*, vol. 22, no. 23, p. 12 808, 2021. DOI: 10.3390/ijms222312808.

[22] M. A. Malik. "Introduction to organic and biochemistry," Open Education Resource (OER) LibreTexts Project. (2024), [Online]. Available: https://chem.libretexts.org/Bookshelves/Introductory_Chemistry/Introduction_to_Organic_and_Biochemistry_(Malik)/00%3A_Front_Matter/03%3A_Table_of_Contents.

[23] F. A. Carey, *Aromatic compound*, https://www.britannica.com/science/aromatic-compound, Encyclopedia Britannica, Mar. 2024.

[24] D. Weininger, "Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules," *Journal of Chemical Information and Computer Sciences*, vol. 28, no. 1, pp. 31–36, 1988. DOI: 10.1021/ci00057a005. eprint: https://doi.org/10.1021/ci00057a005. [Online]. Available: https://doi.org/10.1021/ci00057a005.

[25] I. Daylight Chemical Information Systems. "Simplified molecular input line entry system (smiles)." (2024), [Online]. Available: https://www.daylight.com/dayhtml/doc/theory/theory.smiles.html.

[26] S. Fang, Y. Liu, and S. Liu, "Mfgb: Molecular properties prediction leveraging self-supervised morgan fingerprint representation learning," in *2023 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2023, pp. 3004–3011. DOI: 10.1109/BIBM58861.2023.10385471.

[27] S. Ghosh, *Understanding molecular similarity*, https://medium.com/@santuchal/understanding-molecular-similarity-51e8ebb38886, Accessed: 2024-05-03.

[28] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 6000–6010. DOI: https://doi.org/10.48550/arXiv.1706.03762.

[29] R. Merritt. "What is a transformer model?" (Mar. 2022), [Online]. Available: https://blogs.nvidia.com/blog/what-is-a-transformer-model/.

[30] W. baeldung. "From rnns to transformers." Baeldung on Computer Science. (2024), [Online]. Available: https://www.baeldung.com/cs/rnns-transformers-nlp (visited on 02/09/2024).

[31] H. Touvron, T. Lavril, G. Izacard, *et al.*, *Llama: Open and efficient foundation language models*, 2023. arXiv: 2302.13971 [cs.CL].

[32] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with gpus," *CoRR*, vol. abs/1702.08734, 2017. [Online]. Available: http://arxiv.org/abs/1702.08734.

[33] P. Khosla, P. Teterwak, C. Wang, *et al.*, "Supervised contrastive learning," in *Advances in Neural Information Processing Systems*, Neural Information Processing Systems Foundation, vol. 2020-December, 2020.

[34] T. Wolf, L. Debut, V. Sanh, *et al.*, *Huggingface's transformers: State-of-the-art natural language processing*, 2020. arXiv: 1910.03771 [cs.CL].

[35] T. Vykruta, "Understanding causal llms, masked llms, and seq2seq: A guide to language model training approaches," *Medium*, 2023. [Online]. Available: https://medium.com/@tomasvykruta/understanding-causal-llms-masked-llms-and-seq2seq-a-guide-to-language-model-training-approaches-7b6e1e85ea23 (visited on 10/15/2023).

[36] US Army Corps of Engineers, *Ensemble modeling*, Website, URL: https://hec.usace.army.mil/confluence/hmsdocs/hmstrm/ensemble-modeling, Year.

[37] JavaTPoint, *Define beam search*, https://www.javatpoint.com/define-beam-search, Accessed: 2024-05-03.

[38] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, "Dive into deep learning," *d2l.ai*, 2020, Chapter: Recurrent Neural Networks, Section: Beam Search. [Online]. Available: https://d2l.ai/.

[39] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *KDD (arXiv)*, 2019.

[40] Y. Djoumbou-Feunang, E. Schymanski, J. Zhang, and D. S. Wishart, *S73 | metxbiodb | metabolite reaction database from biotransformer (norman-sles73.0.0.1)*, Data set, 2020. DOI: 10.5281/zenodo.4056561. [Online]. Available: https://doi.org/10.5281/zenodo.4056561.

[41] D. Wishart, Y. Feunang, A. Guo, *et al.*, "Drugbank 5.0: A major update to the drugbank database for 2018," *Nucleic Acids Res*, vol. 46, no. D1, pp. D1074–D1082, 2017. DOI: 10.1093/nar/gkx1037.

[42] M. Whirl-Carrillo, R. Huddart, L. Gong, *et al.*, "An evidence-based framework for evaluating pharmacogenomics knowledge for personalized medicine," *Clinical Pharmacology & Therapeutics*, 2021, Online ahead of print. DOI: 10.1002/cpt.2350.

[43] P. Banerjee, M. Dunkel, E. Kemmler, and R. Preissner, "Supercypspred- a web server for the prediction of cytochrome activity," *NAR-webserver issue*, 2020.

[44] *The metabolomics innovation centre*, https://www.tmicwishartnode.ca/, Accessed: 2024-03-08.

[45] D. S. Wishart, D. Tzur, C. Knox, *et al.*, "Hmdb: The human metabolome database," *Nucleic Acids Res*, vol. 35, no. Database issue, pp. D521–D526, Jan. 2007.

[46] D. Mendez, A. Gaulton, A. P. Bento, *et al.*, "Chembl: Towards direct deposition of bioassay data," *Nucleic Acids Res*, vol. 47, no. D1, pp. D930–D940, 2019. DOI: 10.1093/nar/gky1075.

[47] D. Dimova and J. Bajorath, *Systematic design of analogs of active compounds covering more than 1000 targets*, Zenodo, Feb. 2016. DOI: 10.5281/zenodo.45807.

[48] D. Dimova and J. Bajorath, "Systematic design of analogs of active compounds covering more than 1000 targets," *Med. Chem. Commun.*, vol. 7, pp. 859–863, 5 2016. DOI: 10.1039/C5MD00585J. [Online]. Available: http://dx.doi.org/10.1039/C5MD00585J.

[49] G. Landrum. "Rdkit: Open-source cheminformatics." (2024), [Online]. Available: http://www.rdkit.org.

[50] D. S. Wishart, A. C. Guo, E. Oler, *et al.*, "Hmdb 5.0: The human metabolome database for 2022," *Nucleic Acids Research*, vol. 50, no. D1, pp. D622–D631, Jan. 2022. DOI: 10.1093/nar/gkac323.

[51] L. Buitinck, G. Louppe, M. Blondel, *et al.*, "API design for machine learning software: Experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.

[52] P. Team, "Pytorch 2.1.2," in 2024. [Online]. Available: https://pytorch.org/.

[53] P. S. Foundation, *Python standard library: Difflib helpers for computing deltas*, Accessed: 2024-04-26, 2020. [Online]. Available: https://docs.python.org/3.9/library/difflib.html#difflib.SequenceMatcher.