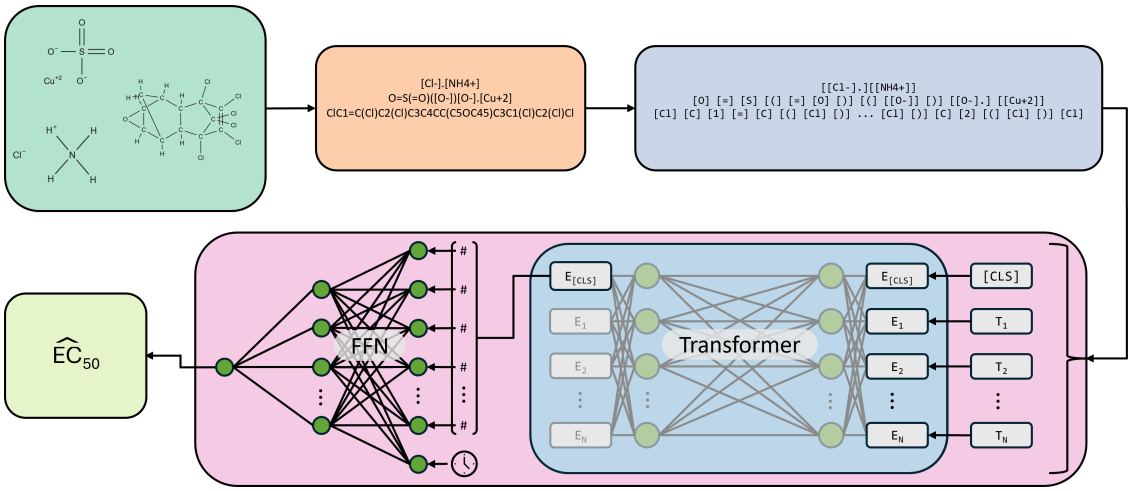
CHALMERS
UNIVERSITY OF TECHNOLOGY



Using Transformers for Chemical Toxicity Prediction

The Impact of Model Architecture on Performance

Master's Thesis in Complex Adaptive Systems

Clara Berglund Hiltunen

DEPARTMENT OF MATHEMATICAL SCIENCES

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

Master's thesis 2025

Using Transformers for Chemical Toxicity Prediction

The Impact of Model Architecture on Performance

CLARA BERGLUND HILTUNEN

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Division of Applied Mathematics and Statistics
Chalmers University of Technology
Gothenburg, Sweden 2025

Using Transformers for Chemical Toxicity Prediction
The Impact of Model Architecture on Performance
CLARA BERGLUND HILTUNEN

Supervisor: Prof. Erik Kristiansson, Department of Mathematical Sciences, Doctoral Student Styrbjörn Käll, Department of Mathematical Sciences
Examiner: Prof. Erik Kristiansson, Department of Mathematical Sciences

Master's Thesis 2025
Department of Mathematical Sciences
Division of Applied Mathematics and Statistics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Workflow from chemical formula to toxicity prediction constructed in PowerPoint. Further details on each step are provided in Sections 2.1, 3.2, and 3.3.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2025

Using Transformers for Chemical Toxicity Prediction
The Impact of Model Architecture on Performance
CLARA BERGLUND HILTUNEN
Department of Mathematical Sciences
Chalmers University of Technology

Abstract

Pollution from toxic chemicals threatens both biodiversity and human health, resulting in significant costs for society. To mitigate these impacts, chemical emission regulations, such as EC_{50}, are employed. These regulations typically establish environmentally safe concentrations of chemicals based on data from *in vivo* experiments, which are usually time-consuming, expensive, and sometimes ethically problematic to conduct. As alternative means to predict chemical toxicity, previous studies have proposed computational methods (e.g., QSAR) and machine learning approaches, including transformer-based models. In one of these previous studies, a pre-trained transformer-based model was employed to predict the EC_{50} values of chemicals for fish. The chemical structures were represented using SMILES notation and served as the input to the model, which consisted of a RoBERTa component followed by a fully connected feed-forward neural network. The present master's thesis builds upon this study by using the same dataset and model framework. It aims to compare the toxicity prediction performance of fine-tuned-only models with different model hyperparameters related to the model architecture and analyze the influence of these hyperparameters. In addition, the thesis aims to evaluate the impact of pre-training by comparing these models with different model hyperparameters to a pre-trained and fine-tuned ChemBERTa model. The effect of model architecture was examined only for the RoBERTa component by varying the following model hyperparameters: embedding size, number of encoder layers, and number of attention heads. The results indicated that increasing the embedding size and the number of encoder layers improved prediction performance. In contrast, no clear pattern regarding the impact of the number of attention heads on prediction performance was observed. Additionally, pre-training appeared to be necessary since the ChemBERTa-based model outperformed all non-pretrained models. These findings contribute to the development of transformer-based machine learning models for chemical toxicity prediction by indicating the optimal directions regarding model architecture and pre-training approaches. Thus, future research may include evaluating whether these findings hold for larger model hyperparameter values, as well as for other chemical representations, toxicity endpoints, and species beyond EC_{50} and fish.

Keywords: toxicity prediction, chemical toxicity, EC_{50}, transformers, ChemBERTa, SMILES, pre-training, machine learning, deep learning, neural networks

Acknowledgements

I would like to offer my deepest appreciation to my supervisor, Prof. Erik Kristiansson, for his continuous support, guidance, and encouragement, as well as to Doctoral Student Styrbjörn Käll for his invaluable mentorship, support, and thoughtful reassurance. I also wish to thank Researcher Mikael Gustavsson. Without the foundational research conducted by him, Prof. Erik Kristiansson, Doctoral Student Styrbjörn Käll, and their colleagues, this master's thesis would not have been possible.

Clara Berglund Hiltunen, Gothenburg, June 20225

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AE	Absolute Error
BERT	Bidirectional Encoder Representations from Transformers
CLS	Classification Token
E	Number of Encoder Layers
EC_{50}	Effective Concentration 50%
Em	Embedding Size
FFN	Feed-Forward Neural Network
H	Number of Attention Heads
LLM	Large language Model
MAE	Mean Absolute Error
QSAR	Quantitative Structure-Activity Relationship
ReLU	Rectified Linear Unit
SMILES	Simplified Molecular Input Line Entry System

Contents

List of Figures

List of Tables

1
Introduction

Biodiversity is globally threatened [3]. A decline in biodiversity may affect ecosystem health and lead to alterations, which, depending on the species that go extinct, can impact productivity and decomposition [4]. Among the multiple contributors to biodiversity loss, chemical pollution is recognized as one of the major ones [3]. For example, chemical pollution is considered a key factor in the steep drop in vulture numbers in India [5], and in the broad decline of bees in Western countries [6].

Humans are also affected by chemical pollution. In the European Union, diseases linked to chemical pollution are estimated to cost €157 billion annually [7], while the corresponding cost in the United States is estimated to be $109 billion [8]. Furthermore, exposure to chemical pollution in air, water, and soil was estimated to have caused between 8.4 and 13 million premature deaths in 2012 [9].

In order to protect the public from hazardous chemicals, the European Union established the Dangerous Substances Directive (DSD) in the 1960s, which was designed to regulate industrial chemicals through legislation [10]. Throughout the years, regulations aiming to protect human health as well as the environment have expanded, and their enforcement has become more rigorous [11, 12, 13, 14].

The environmentally safe concentrations of chemicals are often determined based on *in vivo* experiments, where organisms are exposed to various concentrations of the chemical of interest [2, 15]. The final value is obtained by dividing the concentration at which effects are observed by a safety factor. A commonly studied metric is the half maximal effective concentration (EC_{50}), denoting the concentration at which a chemical elicits 50% of its maximum effect after a predefined exposure time [16].

Moreover, obtaining data through *in vivo* experiments is usually time-consuming, expensive, and sometimes ethically problematic [17, 18]. As a faster and cheaper approach, computational methods have been proposed. One of these is the prominent Quantitative Structure–Activity Relationship (QSAR) modeling approach. In conventional QSAR modeling, the molecular structures of chemicals are used by regression methods to predict toxicological measurement values. Due to the difficulty of predicting the outcomes of larger changes in chemical structure, the development of QSAR models often uses data that is highly stratified by, for instance, chemical structure, exposure conditions, and toxicological effects. Thus, to make predictions for structurally diverse chemicals, multiple models are required due to the limited application domain of a single model.

Furthermore, because of the recent increase in the availability of toxicity data and improvements in computational power, machine learning models have become a popular approach for toxicity prediction [19, 20]. Among these models, a transformer-based model proposed by Gustavsson et al. [2] has shown promising predictive performance, owing to the transformer's ability to identify the structural features of a chemical which are most associated with toxic effects.

This feature-identification capability of the transformer model is enabled by the self-attention mechanism, which creates representations of sequences by capturing the relationships between different positions within them [1]. Hyperparameters that influence the self-attention mechanism include embedding size, number of attention heads, and number of encoder layers. These model hyperparameters collectively define the overall transformer model architecture, thereby affecting the number of trainable parameters. Generally, there is a substantial number of trainable parameters, which results in time-consuming training of the transformer model.

One way to address the tedious model training is pre-training, during which the transformer model is trained on large amounts of unlabeled data in order to learn the overall structure of it [21]. Although pre-training is time-consuming and computationally intensive, it is typically performed only once for a model before it can be fine-tuned, i.e., adapted through additional, relatively inexpensive training to perform a specific task. Pre-trained transformer models have shown improved performance in natural language processing tasks [21, 22], but the impact of pre-training on chemical toxicity prediction performance is still unknown.

Since the promising transformer-based model developed by Gustavsson et al. for chemical toxicity prediction was pre-trained on a dataset containing chemical structures and employed a fixed set of model hyperparameters [2], this master's thesis seeks to build upon these findings by exploring how the model hyperparameters that define the model architecture affect chemical toxicity prediction performance, and to examine the necessity of pre-training for toxicity prediction tasks.

1.1 Aims

In line with the title, *Using Transformers for Chemical Toxicity Prediction: The Impact of Model Architecture on Performance*, this thesis aims to fulfill the following objectives:

- To compare the toxicity prediction performance of models configured with different model hyperparameter values.
- To analyze the influence of different hyperparameters on the performance of toxicity prediction.
- To investigate the impact of pre-training by comparing the toxicity prediction performance of non-pre-trained models and a pre-trained model.

2
Theory

In this chapter, a theoretical background on the key concepts used in this thesis is provided. It starts with an overview of representations of chemical structures, followed by an introduction to tokenization and artificial neural networks. The chapter ends with a section on transformer-based models, including BERT, RoBERTa, and ChemBERTa. While BERT provides the foundation for RoBERTa and ChemBERTa, the latter two play a central role in the methods employed for predicting chemical toxicity in this thesis.

2.1 Chemicals

In today's society, chemicals are utilized across a wide range of applications, including medicine, cosmetics and agriculture [3, 23, 24]. There are both natural and synthetic chemicals, and the usage of the latter has led to significant improvements in food production and living standards [23]. The number of registered chemicals and chemical mixtures exceeded 350,000 in 2022, and this number is expected to grow as new chemicals are likely to be discovered and added in the future [3].

However, as stated previously, some chemicals may have negative effects on the environment and human health. The potential harmful effects, as well as the physical and chemical properties of these chemicals, are largely determined by their molecular structure, which describes the arrangement of the constituent atoms in space [25, 26].

This spatial arrangement can be described in two dimensions by a structural formula, which shows how the atoms, represented by their chemical symbols, are connected or bonded by lines representing the chemical bonds. A chemical bond involves sharing or transferring electrons between atoms to achieve a lower total energy state [27]. Two atoms sharing an electron pair form a single bond, whereas two and three shared electron pairs create double and triple bonds, respectively. In the structural formula, the number of lines between two atoms represents the number of bonds, that is, one line equals a single bond, two lines a double bond, and three lines a triple bond [26]. Figure 2.1a shows the structural formula of ethanol as an example.

Furthermore, if the bonding characteristics and spatial arrangement are disregarded, the constituent atoms of a chemical can be described using a molecular formula [26]. Similar to structural formulas, a molecular formula represents the constituent elements of a chemical using their chemical symbols, but the quantity of each element

is indicated as a subscript following its symbol. For example, the molecular formula of ethanol is C_2H_6O, as seen in Figure 2.1b. This formula conveys that an ethanol molecule consists of two carbon atoms (C), six hydrogen atoms (H), and one oxygen atom (O), but it does not show the spatial arrangement of these atoms.

2.1.1 SMILES

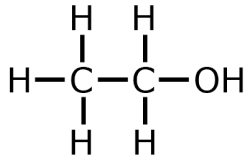
The Simplified Molecular Input Line Entry System (SMILES) is a chemical notation language introduced in the 1980s, designed to make chemical tasks more efficient for computational applications [28]. It represents molecular structures as graphs, such as structural formulas, using linear sequences of characters. This means that SMILES strings specify which atoms are bonded and how, but do not account for the three-dimensional structure. As an example, an ethanol molecule represented by a SMILES string is shown in Figure 2.1c.

When representing molecular structures using SMILES, several rules must be followed [28]. To begin with, as with structural and molecular formulas, SMILES strings denote atoms using their chemical symbols, except that they are enclosed in square brackets. However, exceptions are made for B, C, N, O, P, S, F, Cl, Br, and I, which are written without square brackets when they exhibit their usual number of bonds. Some chemical structures also have branches, which are side chains or groups of atoms connected to the main chain, i.e., the longest continuous chain of atoms. Branches are put in parentheses, which can be stacked or nested when needed.

Hydrogen atoms are typically omitted, i.e., H is not explicitly included in the SMILES strings unless necessary [28]. One such necessary case is when hydrogen atoms are present in ions, i.e., atoms with an electric charge. This is because ions are denoted by their constituent atomic symbols, followed by '+' if positively charged, or '-' if negatively charged, all enclosed in square brackets. Regarding bond representations in SMILES strings, single bonds are omitted, whereas double and triple bonds are denoted by '=' and '#', respectively [28]. In addition, disconnected structures, such as two ions interacting due to opposite charges, are represented by a dot between the entities involved.

In order to represent cyclic chemical structures as SMILES strings, ring closure digits are used to specify the conceptual start and end of the ring [28]. The numbering of atoms can be described as conceptually breaking the ring between two atoms, and then listing the atoms in sequence, starting from one of them. To indicate the atoms where the ring closes, the same digit is placed immediately after them in the SMILES string. Consequently, there may be multiple SMILES strings describing the same cyclic chemical structure.

While SMILES string notations include additional rules [28, 29], these are beyond the scope of this thesis and not essential for the reader's understanding. Nonetheless, for the interested reader, more comprehensive descriptions are available in [28, 29].



(a) Structural formula. **(b)** Molecular formula. **(c)** SMILES string.

Figure 2.1: Ethanol as an example, shown with (a) structural formula, (b) molecular formula, and (c) SMILES string.

2.2 Tokenization

A computer interprets text as a sequence of characters and the division of this sequence into smaller subunits, known as tokens, is called tokenization [30]. This process is performed by a tokenizer, which prepares the text for further use and analysis by partitioning it, typically at layout symbols such as spaces and newline characters. As a result, tokens often correspond to individual words, but tokenization can be refined by introducing additional rules for partitioning. Consequently, the more knowledge there is of the language being tokenized, the better the prerequisites for achieving a more detailed tokenization result.

2.3 Artificial Neural Networks

Artificial neural networks are machine learning methods inspired by the dynamics and architecture of biological neural networks in the mammalian brain [31]. They utilize simplified computational representations of neurons to learn by adjusting the strengths of the connections between them, known as weights. Each weight connects two neurons, and weights can be positive, negative, or zero, in which case zero indicates no connection.

For instance, artificial neural networks can classify and identify structures in data. They are also able to generalize that knowledge to some extent and therefore, they are utilized in natural language processing. Depending on the intended application, the network architecture employed varies.

2.3.1 Feed-Forward Neural Network

A feed-forward neural network (FFN) consists of an input layer, typically followed by one or more hidden layers, and an output layer [31]. Each layer is composed of neurons, and each neuron is connected to all neurons in the succeeding layer, resulting in fully connected layers. These connections are one-way, meaning information can only be passed forward in the network, meaning there are no backward connections to previous layers. Furthermore, there are also no connections within a layer, nor any connections that skip layers. The general architecture of a feed-forward neural network is shown in Figure 2.2.

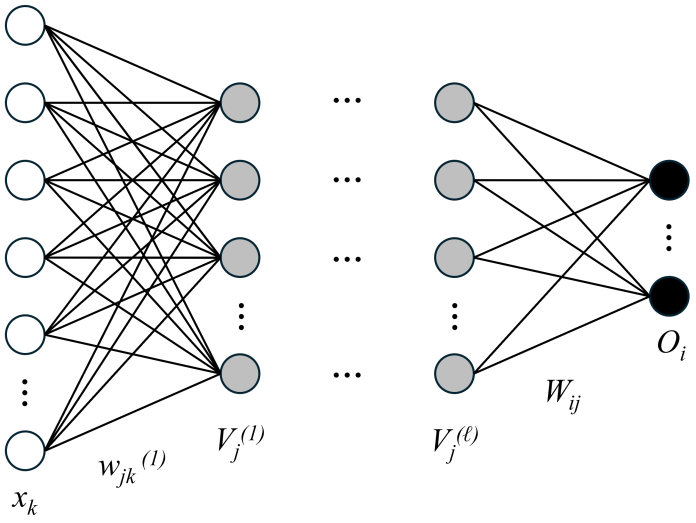


Figure 2.2: General architecture of a feed-forward neural network. White circles represent input neurons in the input layer, gray circles represent neurons in the ℓ hidden layers, and black circles compose the output layer.

The N input values are fed into the neurons of the input layer, which passes the values to the first hidden layer [31]. All neurons in the hidden layer perform the computation described in equation (2.1), where x_k is either the input fed into the input neuron k or the output from the previous hidden layer in the case of multiple hidden layers. Moreover, w_{jk} is the weight between neuron k and hidden neuron j, θ_j is the threshold (or bias) of neuron j, and V_j is the output from neuron j. Also, $g(b)$ is an activation function that typically introduces non-linearity to the network, and may vary depending on the intended task of the network. In addition, a layer index is sometimes indicated as a superscript on the variables to indicate the layer to which they belong.

$$V_j^{(l)} = g(b_j) \quad \text{with} \quad b_j = \sum_{k=1}^{N} w_{jk} x_k - \theta_j \tag{2.1}$$

The input is propagated through the neurons of the hidden layers, where each neuron performs the computation described in equation (2.1) [31]. Upon reaching the output layer, the computation in equation (2.2) is used to compute the final output of the network. In this equation, W_{ij} is the weight between hidden neuron j and output neuron i, and Θ_i is the threshold of output neuron i. O_i is the output of output neuron i and V_j is the output of neuron j in the last hidden layer. The outputs of the output neurons are the final outputs of the feed-forward neural network.

$$O_i = g(B_i) \quad \text{with} \quad B_i = \sum_{j} W_{ij} V_j - \Theta_i \tag{2.2}$$

The aim of a feed-forward neural network is to produce the desired output, typically $O_i^{(\mu)} = t_i^{(\mu)}$ for all output neurons i and all inputs μ, by adjusting its weights and thresholds [31]. By introducing a task-appropriate loss function, which depends on the weights and computes the error of the prediction in relation to the input, the

optimal weights and thresholds are reached when the loss function is minimized. Gradient descent is often used for this minimization, resulting in the update rules for the weights and thresholds seen in equation (2.3) and (2.4). In these equations, w' and θ' are the updated parameter values for both hidden and output neurons, η is the learning rate controlling the magnitude of the updates of these parameter values, and $\mathcal{L}$ is the loss function measuring the model's prediction performance.

$$w'_{mn} = w_{mn} + \delta w_{mn} \quad \text{with} \quad \delta w_{mn} = -\eta \frac{\partial \mathcal{L}}{\partial w_{mn}} \tag{2.3}$$

$$\theta'_{m} = \theta_{m} + \delta \theta_{m} \quad \text{with} \quad \delta \theta_{m} = -\eta \frac{\partial \mathcal{L}}{\partial \theta_{m}} \tag{2.4}$$

2.4 Transformers

Large language models (LLMs) are typically large neural network models, specialized for generating and processing human language [32]. Their architecture is based on a particular neural network model called the Transformer, which was introduced in 2017 [33]. The original Transformer is primarily composed of encoder and decoder layers [1]. In the encoder layers, representations of the contexts and relationships between the input tokens are created, whereas the decoder layers use those representations to generate output sequences [1, 34]. The use of encoder layers, decoder layers, or a combination of both in LLMs enables the models to be applied to a variety of language-related tasks after training on extensive text corpora [33]. Notable examples of LLMs include OpenAI's GPT-3 as well as Google's Gemini and BERT.

2.4.1 BERT

BERT stands for Bidirectional Encoder Representations from Transformers, and utilizes only the encoder component from the original Transformer architecture [1, 21]. BERT is pre-trained using masked language modeling and next sentence prediction, and can be fine-tuned in order to be applied in downstream tasks [21]. The general structure of BERT, as part of the Transformer architecture, is shown in Figure 2.3.

2.4.1.1 Model Architecture

Similar to the original Transformer model, BERT takes sequences of tokens as input [1]. However, in BERT's input sequences, the first token is a classification token, referred to as the `[CLS]` token. It is an additional token that can be considered an aggregated representation for the entire sequence, and is therefore used, for instance, in classification tasks. Moreover, since the Transformer operates using numerical values, the tokens are converted into numerical representations, called embedding vectors, which have a predetermined embedding size of d_{model}. This conversion is done by mapping each token to its corresponding embedding vector in a learned embedding matrix [1, 35]. The fact that the embedding matrix is learned means its parameter values are optimized during model training [36].

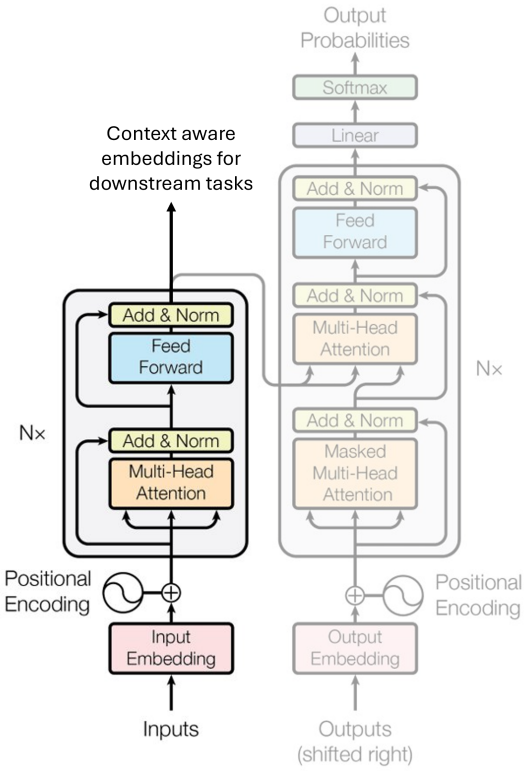


Figure 2.3: The overall Transformer model design adapter from Vaswani et al. [1]. The unfaded left part corresponds to the encoder layers, which correspond to BERT, and the faded part to the right corresponds to the decoder layers.

In order to provide the model with information of relative or absolute positions of the tokens in the input sequences, numerical positional encodings of dimension d_{model} are added to the token embedding vectors [1]. There are different ways of computing positional encodings, but two approaches are sinusoidal functions and learned positional embeddings. The BERT model is also able to take sentence pairs as input [21]. In such cases, a learned embedding called the segment embedding, is added to the embedding vectors to indicate which sentence each token belongs to.

These input representation construction steps are followed by the first encoder layer, whose first sublayer performs multi-head attention [1]. Each attention head can be considered as an independent, parallel attention mechanism. Since each attention head focuses on its own representation subspace, multi-head attention enables the model to simultaneously attend to different parts of and features in the input sequences. Thus, various patterns and relationships within the input sequences are captured by the model [34].

In practice, the vectors resulting from the summation of the token embeddings, positional encodings, and any segment embeddings are stacked such that they form the rows in the so-called embedding matrix, denoted $X \in \mathbb{R}^{L \times d_{model}}$, where L is the length of the input sequence. This matrix is then passed on to the multi-

head attention mechanism in the first encoder layer. There, three projections of X, denoted the Query (Q), the Key (K), and the Value (V) are created, typically by multiplying X with one learned weight matrix for each projection, W^Q for the Query, W^K for the Key, and W^V for the Value. These weight matrices contain different trainable parameters and $W^Q, W^K \in \mathbb{R}^{d_{model} \times d_k}$, whereas $W^V \in \mathbb{R}^{d_{model} \times d_v}$. Here, d_k is the dimension of the queries and keys, and d_v is the dimension of the values, which all three depend on the number of attention heads denoted h. The relationship between these dimensions is shown in equation (2.5).

$$d_k = d_v = \frac{d_{model}}{h} \tag{2.5}$$

Since d_k and d_v are smaller than d_{model} unless there is only one attention head, each head obtains a projection of the Query, Key, and Value matrices with reduced dimensionality compared to the original embedding matrix [1]. Each attention head then independently performs the scaled dot-product attention described in equation (2.6), resulting in h attention outputs, each of dimension $L \times d_v$.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.6}$$

As a final step, the outputs of all attention heads are concatenated and projected once again using $W^O \in \mathbb{R}^{hd_v \times d_{model}}$, as seen in equation (2.7), in order to obtain the final output from the multi-head attention mechanism.

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O \tag{2.7}$$

Thereafter, the sum of the output and input of the multi-head attention sublayer is computed, followed by layer normalization [1]. The output of this computation is then fed into the second sublayer of the encoder, which is a fully connected feed-forward neural network with one hidden layer. The feed-forward network processes all token representations individually using the same weights, and its output is added to its input before applying layer normalization, creating the final output of the encoder layer.

This final output of the encoder layer is passed on as the input to the succeeding encoder layer, where the multi-head attention and the feed-forward network mechanisms are applied again [1]. Then, the output of the last encoder layer, i.e. the final representation of the input sequence, can be used for various applications and tasks [21].

2.4.1.2 Pre-training

The purpose of pre-training is to train a model on large amounts of unlabeled data in order to learn its overall structure. Regarding BERT, it is pre-trained using the pre-training methods masked language modeling and next sentence prediction [21].

Masked language modeling selects 15% of the tokens in the input sequences randomly. These selected tokens, referred to as masked tokens, may be modified according to the following rule, given that the i-th token has been selected. In 80% of

the cases, the i-th token will be replaced by an actual `[MASK]` token, in 10% of the cases it is replaced by a randomly selected token from the corpus, and in the remaining 10% of the cases it is left unchanged. In addition, the random token selection enables the model to utilize context both to the left and to the right of the modified tokens, resulting in the model learning bidirectional context representations.

Moreover, with the aim of learning sentence relationships, the pre-training method, next sentence prediction, is applied [21]. In next sentence prediction, pairs of sentences are selected from the corpus and the two sentences in each pair are denoted `A` and `B`. For 50% of the pairs, sentence `B` is selected so it is the sentence that immediately follows sentence `A` in the original text. In the other 50% of the sentence pairs, sentence `B` is randomly sampled from the corpus.

2.4.2 RoBERTa

RoBERTa stands for Robustly optimized BERT approach and shares the same model architecture as BERT [37]. RoBERTa can be seen as an improved version of BERT, which, in comparison, can be considered undertrained. This improvement in Roberta results from modifications in its pre-training relative to BERT's.

These modifications include the omission of next sentence prediction in RoBERTa's pre-training, causing it to rely solely on masked language modeling, in which the masking pattern changes dynamically each time a given sentence is used during training [37]. In contrast, the masked tokens in BERT are selected once and then fixed throughout the pre-training. Furthermore, RoBERTa is trained on larger datasets and for more training steps than BERT, meaning RoBERTa undergoes more parameter update iterations during pre-training. In addition, RoBERTa is pre-trained on longer input sequences and with larger batch sizes, meaning that a greater number of samples is processed before the model's trainable parameters are updated.

2.4.3 ChemBERTa

Based on RoBERTa, ChemBERTa is a transformer-based model specialized in molecular property prediction [38]. Such as RoBERTa, ChemBERTa was pre-trained using masked language modeling, where 15% of the tokens in the input sequence were masked. However, instead of being pre-trained on English-language corpora like RoBERTa, ChemBERTa was pre-trained on a dataset comprising 77 million unique SMILES strings collected from PubChem, an open-source chemistry database that includes, among other things, chemical structures and their properties [37, 38, 39]. This allows ChemBERTa to capture the relationships in chemical space, which then can be utilized in downstream applications [37].

3
Methods

The purpose of this chapter is to describe the dataset used in this thesis, along with the model architectures, training procedure, post-processing of the model outputs, and evaluation methods. The rationale behind the selected training configurations and evaluation approaches is also provided.

3.1 Dataset

The dataset used in this thesis contains information on the toxicity of chemicals, expressed as EC_{50} concentrations in mg/L, measured at different exposure times for fish. In addition, it includes each chemical's SMILES notation, CAS Registry Number[1], chemical name, and the fish species for which the EC_{50} value was reported.

The data were collected by Gustavsson et al. [2] from the Registration, Evaluation, Authorisation and Restriction of Chemicals (REACH) dossiers in August 2020, as well as from both the United States Environmental Protection Agency (U.S. EPA) ECOTOX database and the European Food Safety Authority (EFSA) pesticide registration data in November 2020. To ensure sufficiently high data quality and avoid outliers, all reported concentrations greater than 500 mg/L were removed by Gustavsson et al. [2].

In total, the dataset contained 52,666 measurements. For some chemicals, there were multiple measurements, but at different exposure times, and in some cases, there were multiple measurements of the same chemical with the same exposure time. In total, there were 3,542 unique chemicals in the dataset and 8,973 unique measurements with respect to chemical and exposure time.

Since some chemical structures can be represented by multiple SMILES, Gustavsson et al. [2] used the open-source Python library `RDKit` version 2024.03.2 to generate canonical SMILES. This resulted in a given chemical always being represented by the same SMILES string, even when multiple valid representations existed. Moreover, the exposure times and EC_{50} values were log-transformed using base 10, and the resulting transformed values were used by the chemical toxicity prediction models in this thesis.

[1] A numerical identifier assigned to chemical substances by the Chemical Abstracts Service [40].

3.2 Tokenization

The canonical SMILES strings were tokenized using the `SmilesTokenizer` module from the open-source Python library `DeepChem`. The vocabulary file used for tokenization was downloaded from the official `DeepChem` GitHub repository [2]. The tokenizer primarily divides the SMILES strings into their smallest meaningful components. If the number of tokens for a single SMILES string exceeded the maximum sequence length of 100 tokens, the sequence was truncated. Conversely, if a SMILES string was shorter, padding was applied to ensure that all sequences had the same length. In order to keep track of which positions in a sequence correspond to padding or SMILES string components, attention masks were introduced.

3.3 Model Architectures

The models used for the chemical toxicity predictions consisted of a RoBERTa component followed by a fully connected feed-forward neural network with one hidden layer, as shown in Figure 3.1. The idea behind this setup was that the transformer learns to capture the chemical structures from the input sequences of tokenized SMILES strings, while the feed-forward network maps these learned representations to toxicity predictions.

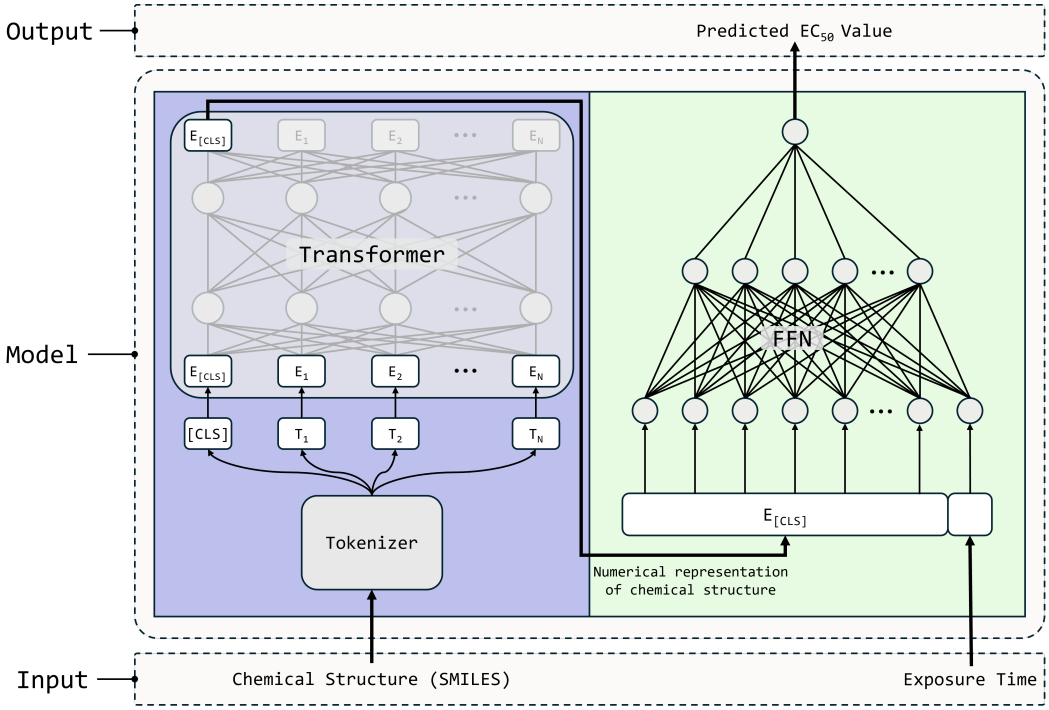


Figure 3.1: Overview of the model achitechture with the transformer and the following FFN adapted from Gustavsson et al. [2].

[2] `https://github.com/deepchem/deepchem/blob/master/deepchem/feat/tests/data/vocab.txt`

Multiple models with unique model architectures were built by changing the following model hyperparameters: number of attention heads, number of encoder layers, and embedding size. The explored model hyperparameter values are listed in Table 3.1 and all possible combinations of these values were evaluated, resulting in a total of 144 unique models. When referring to a model in subsequent sections, a model with a unique combination of these model hyperparameter values is implied.

Table 3.1: Hyperparameter values for embedding size, number of attention heads and number of encoder layers used to build toxicity prediction models. All possible combinations of these hyperparameter values were explored, resulting in a total of 144 unique models.

Model Hyperparameter	Value
Embedding Size	{64, 256, 768}
Number of Attention Heads	{1, 4, 8, 16, 32, 64}
Number of Encoder Layers	{1, 3, 6, 8, 10, 12, 14, 16}

Furthermore, the feed-forward network architecture was identical across all models, except for the input layer, which was adjusted to match the embedding size used in the RoBERTa component. This difference is due to the input to the feed-forward neural network being a concatenation of the exposure time and the corresponding feature vector. The feature vector refers to the numerical embedding of the `[CLS]` token extracted from the RoBERTa component's output for each input sequence. Therefore, the input layer consisted of one more neuron than the embedding size, to accommodate the exposure time. Moreover, the hidden layer had 512 neurons, and since the EC_{50} value was the only output, the output layer consisted of a single neuron. A Rectified Linear Unit (ReLU) activation function was also applied between the hidden layer and the output layer.

3.4 Training Procedure

The batch size and learning rate were optimized by exploring different settings; for more details, see Section 3.4.3. Once the batch size and the learning rate were established, all non-pre-trained models, each with a unique architecture, were fine-tuned using the same set of hyperparameter values. Since no prior pre-training was conducted, all models were fine-tuned only. An overview of these hyperparameters with corresponding values is shown in Table 3.1.

Table 3.2: Overview of the final hyperparameters used in model training.

Hyperparameter	Value
Number of Epochs	100
Batch size	128
Learning rate	$1 \cdot 10^{-5}$
Max sequence length	100
Optimizer	Adam

3.4.1 Loss Function and Optimizer

The loss function used for all models was the Mean Absolute Error (MAE), also known as the L1 loss function, which is shown in equation (3.1), where S denotes the number of samples, $\hat{y}_i$ is the predicted value, and y_i is the true value for sample i [41]. MAE was used because of its ease of interpretation and robustness to outliers, as it penalizes errors in direct proportion to their magnitude. In addition, all models were optimized using the Adam optimizer from the open-source library `PyTorch` version 2.1.2.

$$\text{MAE} = \frac{1}{S}\sum_{i=1}^{S}|\hat{y}_i - y_i| \tag{3.1}$$

3.4.2 Cross-Validation

In order to maximize the usage of the dataset, the models' prediction performance were evaluated by implementing K-fold cross-validation. In this method, the dataset is split into K subsets. K - 1 of these subsets are used to train the model, thus composing the training data, and the remaining subset is used to evaluate the model, thereby making it the validation data [42]. This process is repeated K times, ensuring that each subset is used exactly once as validation data.

In this thesis, K is set to 5. Since multiple measurements exist for some chemicals, the five subsets were created by random sampling based on unique chemicals, i.e., SMILES strings. As a result, all measurements corresponding to a given chemical were assigned to the same subset to prevent data leakage.

3.4.3 Optimization of Batch Size and Learning Rate

The batch size and learning rate were optimized by training multiple separate models with different values for these hyperparameters. The evaluated batch sizes were 32, 128, and 256, whereas the tested learning rates were $1{\cdot}10^{-3}$, $1{\cdot}10^{-4}$, and $1{\cdot}10^{-5}$.

The models used in this hyperparameter optimization of batch size and learning rate all had an embedding size of 768, 12 attention heads, and 6 encoder layers. This architecture was selected with the expectation that the resulting optimal hyperparameter values would generalize well to both smaller and larger model configurations.

The combination of batch size 128 and learning rate $1{\cdot}10^{-5}$ resulted in the lowest prediction error. Therefore, these values were used for all subsequent fine-tuning of the models. In addition, all models were trained for 100 epochs during both the hyperparameter tuning phase and the later fine-tuning phase for model evaluation, since the loss was observed to have stabilized by then.

3.4.4 Weighted Batch Sampling

The batches were resampled for every epoch. Since some chemicals were more abundant in the dataset than others, weighted sampling was used. The weights for each unique SMILES string were calculated according to equation (3.2), where f_i is the number of occurrences of SMILES string i, and ω_i is the weight assigned to all measurements with SMILES string i. The purpose of this weight function was to assign higher weights to less common chemicals in order to make them more likely to be sampled, thereby mitigating overrepresentation of frequently occurring chemicals. The square root of the inverse frequency was applied in order to prevent excessively large weights for very rare chemicals.

$$\omega_i = \frac{1}{\sqrt{f_i}} \tag{3.2}$$

3.4.5 Training Environment

All models were implemented in Python 3.11.3 using the open-source libraries `Hugging Face Transformers` version 4.39.3 and `PyTorch` version 2.1.2. The full code implementation is publicly available on GitHub[3].

Most models were trained on Nivida A40-based compute nodes, each equipped with 4 × NVIDIA Tesla A40 GPUs (48 GB VRAM each), 2 × Intel Xeon Gold 6338 CPUs (32 cores each, 64 cores total) running at 2 GHz and 256 GiB DDR4 RAM. However, due to limited memory capacity on the A40-based nodes, some of the larger models were trained on Nivida A100-based compute nodes. These nodes were each equipped with 4 x NVIDIA Tesla A100 HGX GPU with 80GB RAM, 2 x 32 core Intel(R) Xeon(R) Gold 6338 CPU @ 2GHz (total 64 cores) and 1024GiB DDR4 RAM.

The models trained on these A100-based nodes included models with embedding size 768, 64 attention heads and 10 to 16 encoder layers, and models the same embedding size, but 32 attention heads and 14 to 16 encoder layers. Also, models with embedding size 256, 64 heads and 14 to 16 encoder layers were trained on A100-based nodes, while the remaining ones were trained on A40-based nodes.

3.5 Evaluation

The outputs of the models during validation were predicted toxicity values for the chemicals in the validation set. For each chemical, the absolute error (AE) was calculated as defined in equation (3.3).

$$\mathrm{AE}_i = |y_i - \hat{y}_i| \tag{3.3}$$

The aggregated absolute error values served as the basis for most model performance evaluations. The following subsections provide detailed descriptions of the aggre-

[3] `https://github.com/tessa116cbh/Master-s_Thesis_Using_Transformers_for_Chemical_Toxicity_Prediction.git`

gation procedure as well as various evaluation methods that were used, including those beyond absolute error.

3.5.1 Aggregation of Model Output

As mentioned berfore, some chemicals had multiple experimental measurements, occasionally even for the same exposure time, which resulted in multiple predictions and, consequently, multiple absolute errors for a single chemical. This overrepresentation can introduce bias in the evaluation. With this in mind, and since the focus was on accurately estimating the toxicity of individual chemicals rather than optimizing performance across the dataset, aggregation steps were applied to produce a single predicted value and a single absolute error per chemical.

The aggregation was executed in two steps. In the first step, all predictions and absolute errors for the same combination of chemical and exposure time were aggregated using the mean, resulting in one mean prediction and one mean absolute error per chemical-exposure time pair. In the second step, these mean values were further aggregated over the exposure times for each chemical, resulting in one aggregated prediction and one aggregated absolute error per unique chemical.

3.5.2 Performance Assessment

For each of the 144 models, the aggregations of the output predictions and absolute errors were computed per epoch and fold, resulting in a total of 500 aggregated values per metric, since the model was trained for 100 epochs across 5 folds. To determine each model's best performance, the mean of the fold-wise medians was calculated for every epoch. This metric is referred to as median $\log_{10}$ prediction error, and the lowest value of this metric, regardless of which epoch it was achieved in, was considered to represent that model's best performance. Only the aggregated model outputs from the epoch attaining the best performance were used for further model evaluation.

3.5.2.1 Pearson's Correlation coefficient

Since Pearson's correlation coefficient (r) measures the direction and strength of a linear relationship between two variables, it was used to evaluate potential linear associations between either the model hyperparameters or the number of trainable parameters, and the median $\log_{10}$ prediction error [43]. Pearson's correlation coefficient is defined in equation (3.4), where $\hat{y}_\nu$ and y_ν denote paired values of the two variables for instance ν, and S is the total number of instances.

$$r = \frac{S \sum_{\nu=1}^{S} \hat{y}_\nu y_\nu - \sum_{\nu=1}^{S} \hat{y}_\nu \sum_{\nu=1}^{S} y_\nu}{\sqrt{\left(S \sum_{\nu=1}^{S} \hat{y}_\nu^2 - \left(\sum_{\nu=1}^{S} \hat{y}_\nu\right)^2\right)\left(S \sum_{\nu=1}^{S} y_\nu^2 - \left(\sum_{\nu=1}^{S} y_\nu\right)^2\right)}} \tag{3.4}$$

Additionally, Pearson's correlation coefficient ranges from –1 to 1 ($-1 \leq r \leq 1$), where $r = 1$ indicates a perfect positive linear relationship, and $r = -1$ indicates a

perfect negative linear relationship [43]. $r = 0$ indicates no linear relationship, while values between 0 and ± 1 imply some linear relationship, albeit not a perfect one.

3.5.2.2 Spearman's rank correlation coefficient

Spearman rank correlation coefficient (ρ) is used to assess the monotonic relationship between two variables, i.e., the extent to which the relationship between two variables can be described with an monotonically increasing or decreasing function based on their ranks [44]. Due to this property, Spearman's rank correlation coefficient was used to assess any monotonic relationship between either the model hyperparameters or the number of trainable parameters, and the median $\log_{10}$ prediction error.

Furthermore, the ranks used in Spearman's rank correlation coefficient are obtained by ranking the instances of each variable separately, with the largest values assigned rank S and the smallest rank 1 [45]. For each variable pair ν, the difference of their ranks, denoted D_ν, is computed before ρ is calculated as described in equation (3.5).

$$\rho = 1 - \frac{6 \sum_{\nu=1}^{S} D_\nu^2}{S(S^2 - 1)} \tag{3.5}$$

The range of Spearman rank correlation coefficient is $-1 \leq \rho \leq 1$, where $\rho < 0$ means a negative relationship between the examined variables and $\rho > 0$ implies a positive one [45]. The closer to -1 and 1 ρ is, the more perfect the relationship is, and if the variables are uncorrelated, then $\rho = 0$.

3.5.2.3 90^{th} Percentile

For each model, the 90^{th} percentile of the aggregated absolute errors was computed within each of the five folds from the 5-fold cross-validation. The mean of these five percentile values was then calculated, resulting in the final 90^{th} percentile of the $\log_{10}$ performance error. By computing this performance metric, model performance on the 10% most difficult-to-predict chemicals can be assessed, since 90% of the chemicals have prediction errors below the 90^{th} percentile. Thus, the 90^{th} percentile provides insights into the models' reliability and their distribution of prediction errors.

3.5.2.4 Fold Change

Fold change is a measurement to quantify the relative change between two values, reflecting the extent to which one value is increased or decreased relative to a reference value [46]. Thus, fold change was used to assess the number of unique chemicals for which the models under- and overestimated toxicity values. It was calculated as the ratio between the aggregated predicted and aggregated true toxicity values, with the latter serving as the reference value. Consequently, only one ratio calculation was performed for each unique chemical. Moreover, to identify any substantial deviations, a 1000-fold change threshold was applied, such that only aggregated pre-

dicted values exceeding or falling below the reference value by a factor of 1000 were deemed significant.

3.5.3 Benchmarking

As a benchmark for the examined models, the pre-trained ChemBERTa model `seyonec/PubChem10M_SMILES_BPE_450k`, available via the HuggingFace Transformers library, was used. It had 6 encoder layers, 12 attention heads, and an embedding size of 768. In addition, to ensure compatibility with the pre-trained ChemBERTa model, the tokenizer was switched from `SmilesTokenizer` from `DeepChem` to the one associated with the pre-trained model, as tokenization and the model's expected input format must align. This new tokenizer was loaded using HuggingFace's `AutoTokenizer` and splits SMILES strings into common subtokens, i.e., typically longer segments of the strings than the smallest meaningful components used by the tokenizer from `DeepChem`.

The pre-trained ChemBERTa model was fine-tuned on the same chemical toxicity prediction task as the other non-pre-trained models. In order to make these models as comparable as possible, the batch size and learning rate for the pre-trained model were the same as for the non-pre-trained ones, i.e., 128 and $1 \cdot 10^{-5}$, respectively. The output results were aggregated and evaluated in the same way as the output of the other models.

4
Results

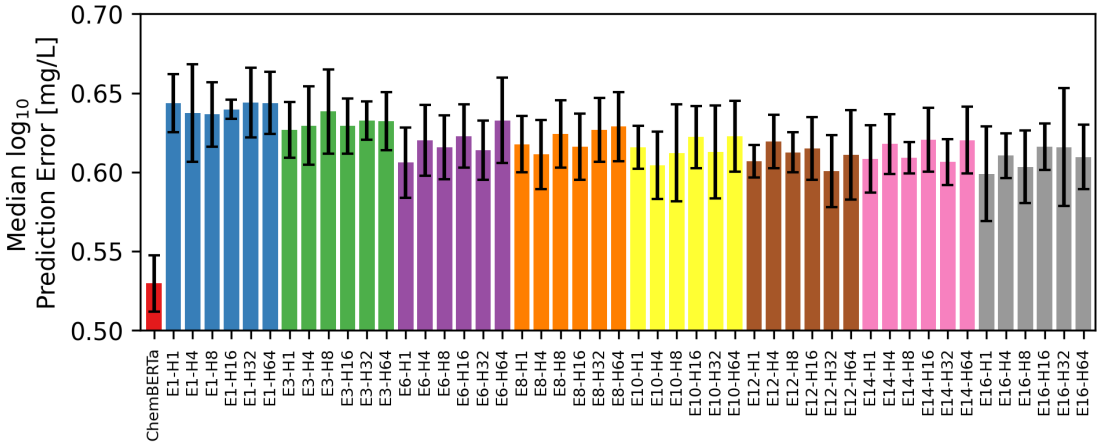
This chapter presents the results, that have been post-processed (i.e., best epoch selection and aggregation), and evaluated based on the framework outlined in Section 3.5 and its subsections. Moreover, the observations emphasized in the figures and tables in the following sections are highlighted in relation to the aims specified in Section 1.1.

4.1 Model Performance Comparison

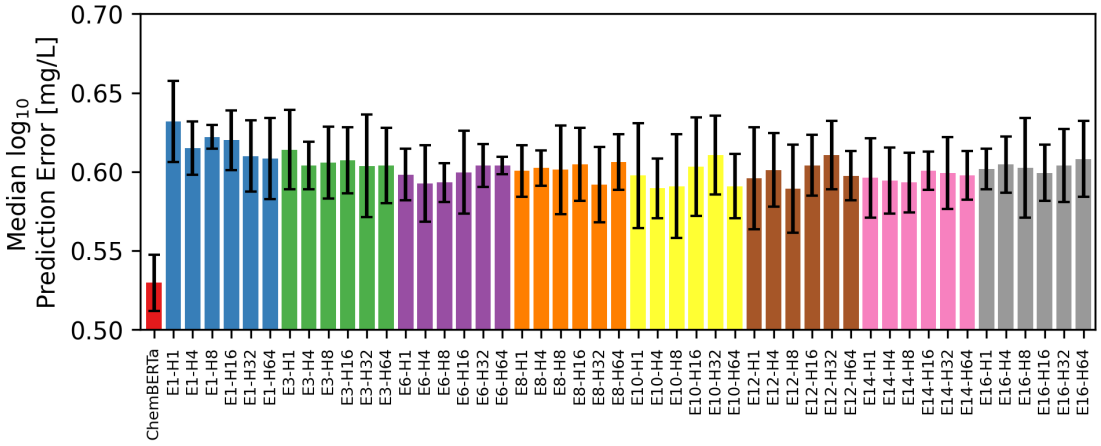
In accordance with the first and third aims, the chemical toxicity prediction performance of models with different architectures, determined by varying the following model hyperparameters: embedding size (Em), number of encoder layers (E), and number of attention heads (H), is compared. The performance of these non-pretrained models is shown in Figure 4.1. In this figure, each bar represents a model with a unique architecture, and the performance metric is the median $\log_{10}$ prediction error computed as described in Section 3.5.2. Additionally, the reported prediction errors are the means of the five median absolute errors, one from each fold in the 5-fold cross-validation, and the error bars correspond to the standard deviation among these five folds. The subfigures are grouped according to the embedding size used in each model configuration. Consequently, models in subfigure 4.1a use an embedding size of 64, those in 4.1b use 256, and those in 4.1c use 768. The exact values shown in these subfigures are listed in Table A.1, located in Appendix A.1.

Moreover, Figure 4.1 shows that the fine-tuned ChemBERTa outperforms all other non-pretrained models, regardless of their architecture, because it achieves a lower median $\log_{10}$ prediction error. Specifically, ChemBERTa's median $\log_{10}$ prediction error is 0.5297, which is 0.0504 lower on the log scale than the best-performing non-pretrained model, E10-H32 (i.e., 10 encoder layers and 32 attention heads) with an embedding size of 768 and a median $\log_{10}$ prediction error of 0.5801.

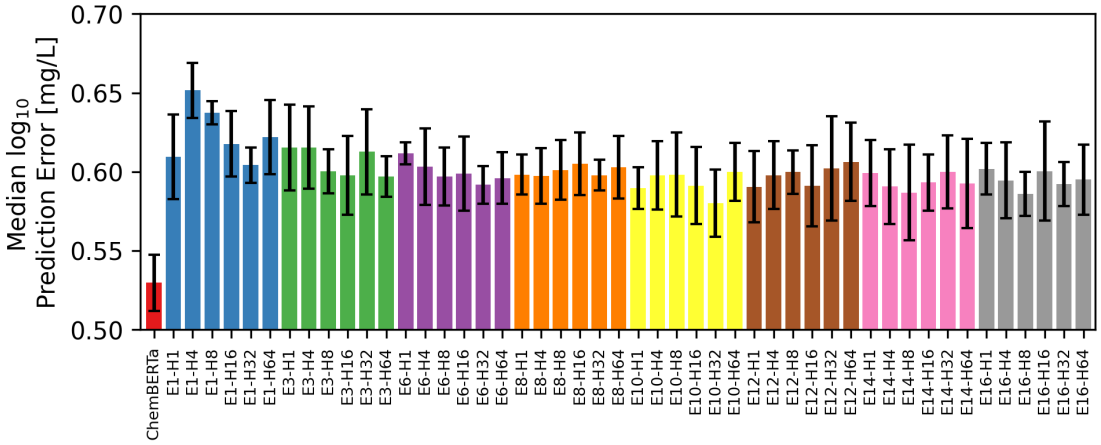
In addition, the figure shows that multiple models perform similarly. In fact, there are five models whose performance is within 0.01 $\log_{10}$ units of that of E10-H32 with an embedding size of 768, and an additional 42 models whose performance is within 0.02 $\log_{10}$ units of the best-performing model. Notably, the worst-performing model, E1-H4 with an embedding size of 768, achieves a median $\log_{10}$ prediction error of 0.6516, resulting in a performance range of 0.0715 $\log_{10}$ units.



(a) Embedding size 64.



(b) Embedding size 256.



(c) Embedding size 768.

Figure 4.1: Median $\log_{10}$ prediction performance, computed for all models as the mean of median performances across the five cross-validation folds, with error bars indicating variability between folds. The fine-tuned ChemBERTa model is shown in red at the left for benchmarking.

4.2 Impact of Model Hyperparameters on Performance

To address the second aim, this section emphasizes the evaluation of the impact of model hyperparameters on performance. In Figure 4.1, which shows the median $\log_{10}$ prediction errors for all models, it can be observed that models with embedding sizes of 256 and 768 generally perform better, i.e., exhibit lower prediction errors, compared to models with an embedding size of 64. This observation is further supported by the mean median $\log_{10}$ prediction errors across all models with the same embedding size. Models with embedding sizes of 256 and 768 have similar mean values, 0.6025 ($\sigma = 0.0085$) and 0.6012 ($\sigma = 0.0123$), respectively. These values are lower than the mean value of the models with an embedding size of 64, which have a value of 0.6200 ($\sigma = 0.0118$). All these observations suggest that larger embedding sizes are generally associated with lower prediction errors.

This trend is further supported by the Pearson and Spearman correlation coefficients between model hyperparameters and the median $\log_{10}$ prediction error, as listed in Table 4.1. In this table, it is noted that correlation coefficients between embedding size and median $\log_{10}$ prediction error are r = -0.605 and ρ = -0.464. This indicates a moderately negative relationship between embedding size and median $\log_{10}$ prediction error, confirming that increasing the embedding size tends to improve predictive performance.

Additionally, Figure 4.1 suggests that, despite some variations, increasing the number of encoder layers generally leads to a decrease in the median $\log_{10}$ prediction error. This is supported by computing the mean of the median $\log_{10}$ prediction errors across all models sharing the same number of encoder layers. The results show that increasing the number of encoder layers from 1 to 16 leads to a decrease in $\log_{10}$ prediction error from 0.6275 to 0.6025, although this reduction is not strictly monotonic across all configurations, as shown in the extended results provided in Table A.3 in Appendix A.3.1. Nevertheless, the general observation that a higher number of encoder layers overall leads to lower prediction error is further supported by the Pearson and Spearman correlation coefficients in Table 4.1. Specifically, $r = -0.443$ and $\rho = -0.521$, which indicate a moderate negative correlation between the number of encoder layers and the prediction error across all models.

Regarding the number of attention heads, no clear relationship with prediction error is evident in Figure 4.1. This is further supported by the mean of median $\log_{10}$ prediction errors averaged over models with the same number of attention heads, as shown in Table A.5 in Appendix A.3.2, where only minor differences between the values are observed. Additionally, the Pearson's and Spearman's correlation coefficients are both close to zero, indicating no significant correlation, as presented in Table 4.1. However, Figure 4.1 also suggests that models using 1 or 64 attention heads rarely achieve the lowest prediction error within a given combination of embedding size and number of encoder layers.

Table 4.1: Pearson's (r) and Spearman's (ρ) correlation coefficients, with p-values in parentheses, quantifying the relationship between both model hyperparameters or number of trainable parameters and median $\log_{10}$ prediction error.

Model Property	Pearson r (p-value)		Spearman ρ (p-value)	
Embedding Size	-0.605	$(9.11 \cdot 10^{-16})$	-0.464	$(4.53 \cdot 10^{-9})$
# Encoder Layers	-0.443	$(2.60 \cdot 10^{-8})$	-0.512	$(4.27 \cdot 10^{-11})$
# Attention Heads	0.051	$(5.43 \cdot 10^{-1})$	0.041	$(6.24 \cdot 10^{-1})$
# Trainable Parameters	-0.578	$(3.33 \cdot 10^{-14})$	-0.787	$(1.29 \cdot 10^{-31})$

4.3 Impact of Number of Trainable Parameters on Performance

Since the number of trainable parameters is related to the model hyperparameters, the impact of the number of trainable parameters on chemical prediction performance was evaluated as an extension of the second aim. The total number of trainable parameters for the various model architectures is displayed in Figure 4.2. Since the number of attention heads does not affect the number of trainable parameters given a set of E and Em, it is not included. Moreover, the figure also shows that the number of trainable parameters increases significantly with embedding size, and that it increases approximately linearly with the number of encoders when the embedding size is fixed.

Regarding chemical toxicity prediction performance, Table 4.1 shows that there is a moderate to strong negative correlation between the number of trainable parameters and median $\log_{10}$ prediction error, because r = -0.578 and ρ = -0.787. This implies that models with more trainable parameters tend to achieve lower prediction errors.

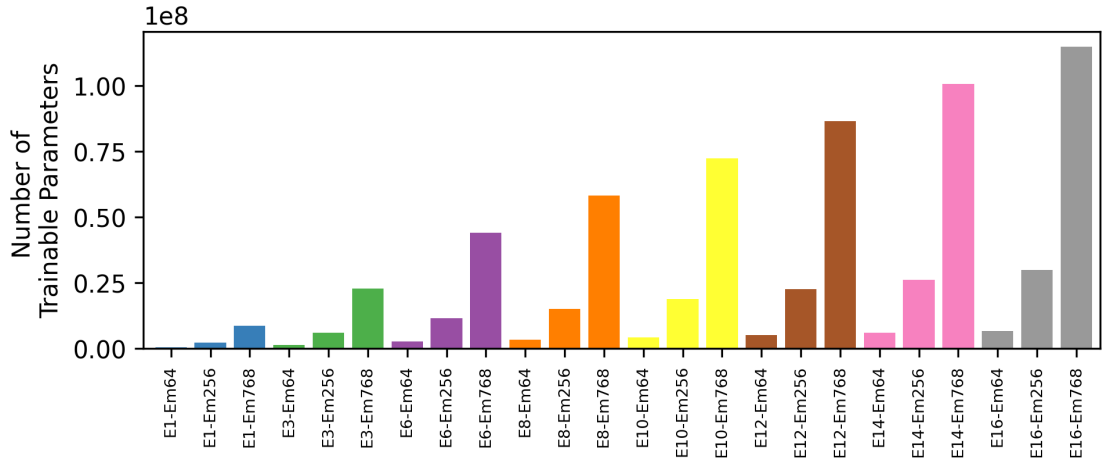


Figure 4.2: Number of trainable parameters of the models. Number of attention heads is not included, as it does not affect trainable parameters for fixed H and Em.

4.4 Model Performance at the 90th Percentile

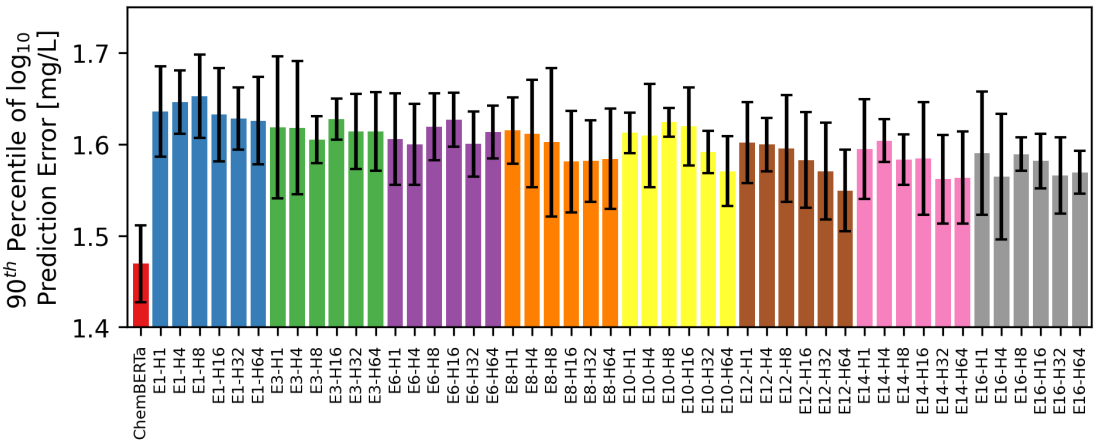
To address all three aims, the performance of the models at the 90th percentile of $\log_{10}$ prediction error was evaluated. This metric, computed for all models as the mean of the 90th percentile values across the 5 cross-validation folds, is presented in Figure 4.3. The error bars show the standard deviations across these folds and the exact values in this figure are listed in Table A.2 in Appendix A.2.

Furthermore, in Figure 4.3, it can be observed that the fine-tuned ChemBERTa model achieves a lower 90th percentile of the $\log_{10}$ prediction error compared to all non-pre-trained models, indicating superior performance in predicting toxicity for the 10% most challenging chemicals. The 90th percentile for ChemBERTa is 1.4694, outperforming the best-performing model, E12-H64 with an embedding size of 768, which achieves 1.5297.

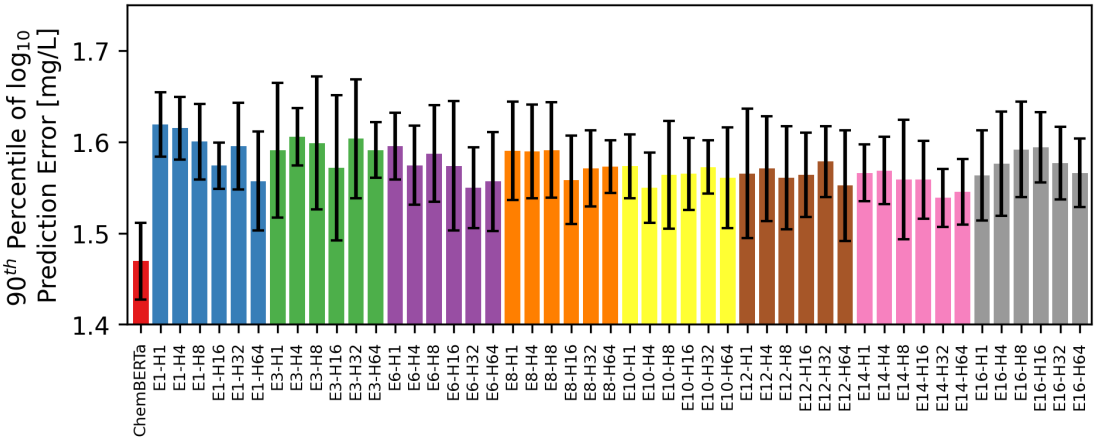
Figure 4.3 also shows that models with an embedding size of 768 generally yield lower 90th percentile prediction errors than those with an embedding size of 256, which in turn have lower prediction errors than models with an embedding size of 64. This visually discernible trend is supported by the corresponding means of the 90th percentile $\log_{10}$ prediction errors, averaged across all encoder layer and attention head configurations for each embedding size. The resulting means are 1.6009 ($\sigma = 0.0234$) for embedding size 64, 1.5754 ($\sigma = 0.0183$) for 256, and 1.5640 ($\sigma = 0.0255$) for 768. Thus, these results indicate that, overall, a larger embedding size contributes to improved prediction performance, even for the 10% most difficult-to-predict chemicals.

In addition, Figure 4.3 suggests that increasing the number of encoder layers tends to reduce the 90th percentile of the prediction error to some extent. The same general pattern is observed among the means of the 90th percentile of $\log_{10}$ prediction error computed for each unique number of encoder layers value, as they are decreasing from 1.6124 for 1 encoder layer to 1.5737 for 16 encoder layers, albeit with some minor fluctuations, as shown in Table A.4 in Appendix A.3.1.

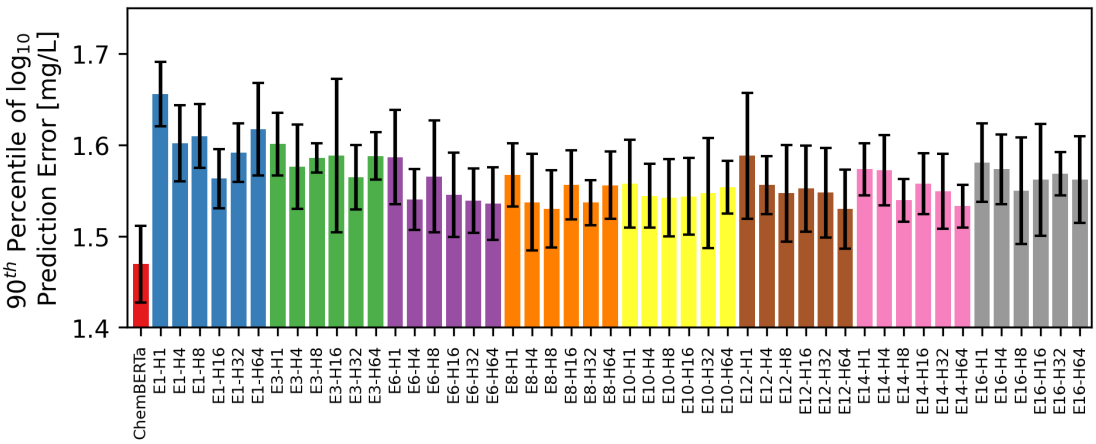
Moreover, although Figure 4.3 shows that the number of attention heads does influence model performance on the 10% most difficult-to-predict chemicals, no consistent trend can be observed either in that figure. However, the mean 90th percentile $\log_{10}$ prediction errors obtained by averaging over models with the same number of attention heads, shown in Table A.6 in Appendix A.3.2, implies that increasing the number of attention heads reduces the 90th percentile of the prediction error from 1.5938 for 1 attention head to 1.5696 for 64 attention heads.



(a) Embedding size 64.



(b) Embedding size 256.



(c) Embedding size 768.

Figure 4.3: 90$^{\text{th}}$ percentile of $\log_{10}$ prediction error, computed for all models as the mean of the 90$^{\text{th}}$ percentile values across the 5 cross-validation folds. Error bars indicate variability between folds. The fine-tuned ChemBERTa model is shown in red on the left for benchmarking.

4.5 Frequency of Over- and Under-Predictions at the 1000-Fold Change Threshold

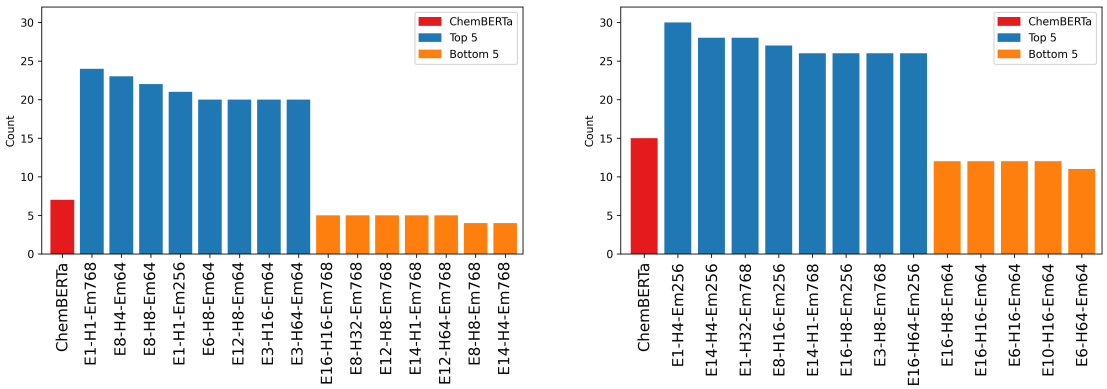
To complement the addressing of all three aims, fold changes with a 1000-fold change threshold are presented in Figure 4.4 for the five models with the most and the fewest under-predictions (Figure 4.4a) and over-predictions (Figure 4.4b) of chemical toxicity values for unique chemicals. The corresponding results for all trained models are provided in Figures A.1 and A.2 in Appendix A.4.

The model with the most under-predictions of toxicity values, E1-H1-Em768, underestimates the toxicity values of 24 unique chemicals. In contrast, the lowest number of under-predictions of toxicity values, 4 values, is achieved by E8-H8-Em768 and E14-H4-Em768. Thus, they outperform the fine-tuned ChemBERTa, which underpredicts the toxicity values of 7 unique chemicals.

Additionally, Figure 4.4a shows that the five models with the fewest under-predictions of chemical toxicity values all share two model hyperparameter prperties: an embedding size of 768 and at least 8 encoder layers. Though, the number of attention heads varies from 1 to 64. In contrast, models with the highest number of underpredictions of toxicity values predominantly exhibit an embedding size of 64 and 12 or fewer encoder layers. However, similar to the group with fewest underestimation of toxicity values, the number of attention heads among these models also ranges from 1 to 64.

In terms of over-prediction of toxicity values, in Figure 4.4b, it can be observed that E1-H14-Em256 overestimates the toxicity values of 30 unique chemicals, which is the highest among all models. In contrast, E6-H64-Em64 has the lowest number of over-predictions of toxicity values, over-predicting toxicity values for 11 chemicals. The fine-tuned ChemBERTa overestimates the toxicity values of 15 chemicals.

It is also notable that all models with the fewest number of over-predictions of toxicity values have an embedding size of 64, between 6 and 10 encoder layers, and at least 8 heads. Conversely, the models with the most over-predictions of toxicity values have embedding sizes of either 256 or 768, whereas the number of attention heads varies between 1 and 64. Also, the encoder layers counts range from 1 to 16 for embedding size 256, and between 1 and 14 for embedding size 768.



(a) Predicted toxicity value < true toxicity value.

(b) Predicted toxicity value > true toxicity value.

Figure 4.4: Fold change with a 1000-fold change threshold for the five models with the most and the fewest under-predictions (a) and over-predictions (b) of chemical toxicity values. Due to tied values, some groups may include more than five models. The performance of the fine-tuned ChemBERTa model is shown in red for comparison.

5

Discussion and Conclusion

After presenting the findings of this thesis in Chapter 4, the focus now shifts to addressing their implications in relation to the aims, namely to compare the toxicity prediction performance of different model architectures, to analyze the influence of hyperparameters on predictive performance, and to investigate the impact of pre-training. This chapter also covers the thesis's limitations, relation to prior research, and relevance to the field, and concludes with suggestions for future research.

5.1 Discussion of Results in Relation to the Aims

As noted in Section 4.1, the models exhibited similar chemical toxicity prediction performance, varying by only 0.0715 in median $\log_{10}$ prediction error. This suggests that the choice of model architecture may have a limited impact on predictive accuracy for this task, but also that the choice of tokenization method, training hyperparameters or feed-forward network architecture, which were common across all models, were not optimal. Given that the intended purpose of the feed-forward network is to predict toxicity based on chemical representations and exposure time, it is possible that network architecture or training hyperparameters or both may have limited its ability to fully leverage the chemical structural representations provided by the RoBERTa component of each model.

In addition, dividing the SMILES into their smallest meaningful components may have made it difficult for the RoBERTa component of the models to effectively learn the chemical structures associated with toxicity, although it should, in theory, be capable of doing so. A possible explanation for this is that sequences composed of many small tokens introduce more irrelevant or less informative individual tokens, which may act as noise. Thus, the attention mechanism must process more elements to infer spatial relationships and toxicity-related patterns, possibly diluting attention across less relevant elements.

Concerning the model architecture, although the models exhibit only minor variations in predictive performance, patterns within this variation are observed in Sections 4.2 and 4.4. Based on visualizations of prediction performance using bar plots, as well as analyses of correlation coefficients and means averaged with respect to model hyperparameters, it is noted that increasing the embedding size or the number of encoder layers tends to reduce the median $\log_{10}$ prediction error and the 90^{th} percentile prediction error.

A possible explanation for the observed trend related to embedding size is that a larger embedding size allows a model to capture and store more information for each token, including the `[CLS]` token. Thereby, the feed-forward network is provided with more comprehensive information about the chemical structures, enabling more accurate toxicity predictions. In addition, the representations created by an encoder layer are passed on to the next, allowing the model to gradually learn deeper and more complex characteristics of the SMILES strings, which may help explain the observed trend related to the number of encoder layers and increased performance.

On the other hand, the fold changes with a 1000-fold change threshold in Section 4.5 show that the most over-predictions of toxicity values were exhibited by models with an embedding size of 256 or 768 and between 1 and 16 encoder layers. Over-prediction of toxicity values is more concerning for real-world applications than under-prediction in this context, since the toxicity values are measured as EC_{50}. This means that chemicals whose toxicity values are over-predicted by the models are actually more toxic than predicted, since a lower true EC_{50} indicates higher toxicity. Thus, increasing the embedding size and number of encoder layers to reduce the prediction error and the 90^{th} percentile prediction error can sometimes lead to an underestimation of actual toxicity. Therefore, careful analysis of the results is necessary to address the potential increase in toxicity underestimation for chemicals.

Furthermore, as observed in Sections 4.2, the number of attention heads appeared to affect the prediction error, although no clear trend could be identified. One possible explanation is that the embedding size and number of encoder layers already provide sufficient capacity for predicting chemical toxicity values, thereby reducing the contribution of additional attention heads. Another explanation could be the detailed tokenization, where small tokens limit the extraction of structural patterns by the attention heads. Thus, the benefit of multi-head attention is reduced, possibly causing the parallel attention heads to focus on similar features. This suggests that the underlying issue may originate from factors other than the attention mechanism and the chosen number of attention heads. On the other hand, Section 4.4 shows that the mean of 90^{th} percentile prediction error averaged over models with the same number of attention heads is reduced when increasing the number of attention heads. Thus, it is implied that more attention heads have a positive effect on prediction of the 10% most difficult-to-predict chemicals, suggesting that the models nonetheless have some benefit from the multi-head attention.

Regarding the pre-trained and fine-tuned ChemBERTa model, it outperformed all non-pre-trained models in terms of median $\log_{10}$ prediction error and 90^{th} percentile of prediction error, as described in Sections 4.1 and 4.4, showing that pre-training significantly enhances the model's ability to generalize and accurately predict chemical toxicity. This may be due to its advantage in learning generalized representations of chemical data during the pre-training phase, whereas the non-pre-trained models lack this prior knowledge and must learn these representations solely from the limited labeled data during fine-tuning, see Section 3.4.

5.2 Limitations

Some limitations in this thesis include performance metrics, technical limitations, external validity, and the utilization of SMILES strings as representations for chemical structures. For measuring the performance of the models, the output values and their corresponding absolute errors were aggregated in two steps using the mean at each stage. This was done to ensure a single measurement per chemical, since some chemicals had multiple entries in the dataset, as described in Section 3.5.1. While this aggregation simplified the evaluation process, it may have reduced the variability in model performance, thereby making the models appear more consistent than they actually are. Additionally, these aggregations may result in chemicals with few entries having the same weight as chemicals with multiple entries when computing the median or the 90^{th} percentile, consequently disproportionately influencing these metrics used to evaluate model performance.

Technical limitations due to memory capacity on the computing nodes prevented the evaluation of larger batch sizes and combinations of model hyperparameters, such as larger embedding sizes and more encoder layers, resulting in more trainable parameters. Thus, whether further increases in embedding size and number of encoder layers would continue to improve chemical toxicity prediction performance remains unexplored.

Additionally, external validity is somewhat limited since the dataset used consisted of EC_{50} values for fish. As a consequence, whether the findings and conclusions of the models examined in this thesis are applicable to other species or toxicity endpoints remains uncertain and this requires further investigation. Moreover, since SMILES strings are linear representations of three-dimensional chemical structures, they may not fully represent the stereochemistry of a chemical, potentially affecting the accuracy and predictive performance of the models.

5.3 Connection to Previous Research

As outlined in the Introduction, a previous study employed a transformer-based model pre-trained on chemical structure data, using a fixed set of model hyperparameters [2]. This master's thesis builds upon that work by investigating how changes in model hyperparameters affect prediction performance and by examining whether pretraining is necessary for toxicity prediction tasks. The results showed that certain model hyperparameters, i.e., embedding size and number of encoder layers, had a clear impact on prediction performance, and that models without pretraining showed decent performance, but generally did not outperform the pretrained benchmark. Due to differences in evaluation metrics between the studies, direct comparison with the results of the previous study is difficult. However, the results of both studies indicate that transformer-based models hold potential for chemical toxicity prediction.

5.4 Relevance and Outlook

The findings in this thesis help to highlight promising directions for model architecture and the role of pre-training when it comes to transformer-based machine learning models for toxicity prediction for chemicals. By advancing these models, faster and more cost-effective toxicity assessments can be achieved, thereby reducing the need for extensive and sometimes unethical animal testing. This creates a foundation for a rapid and efficient generation of data to support chemical emission regulation. Given the continuous discovery and development of new chemicals, ensuring timely toxicity assessment is increasingly important to reduce the harmful effects of chemicals on the environment and human health.

In addition, further research could help clarify the impact of using more extreme values for model hyperparameters, particularly embedding size and the number of encoder layers, and also investigate whether these findings generalize to other chemical representations, such as molecular graphs, as well as to different toxicity endpoints and species beyond fish and EC_{50}.

Bibliography

[1] Vaswani A, Brain G, Shazeer N, Parmar N, Uszkoreit J, Jones L, et al. Attention Is All You Need. In: 31st Conference on Neural Information Processing Systems (NIPS 2017). Long Beach, CA, USA; 2017. Available from: https://arxiv.org/pdf/1706.03762.

[2] Gustavsson M, Käll S, Svedberg P, Inda-Diaz JS, Molander S, Coria J, et al. Transformers enable accurate prediction of acute and chronic chemical toxicity in aquatic organisms. Science Advances. 2024 3;10(10):6669. Available from: /doi/pdf/10.1126/sciadv.adk6669?download=true.

[3] Sigmund G, Ågerstrand M, Antonelli A, Backhaus T, Brodin T, Diamond ML, et al. Addressing chemical pollution in biodiversity research. Global Change Biology. 2023 6;29(12):3240-55. Available from: https://doi.org/10.1111/gcb.16689.

[4] Hooper DU, Adair EC, Cardinale BJ, Byrnes JEK, Hungate BA, Matulich KL, et al. A global synthesis reveals biodiversity loss as a major driver of ecosystem change. Nature. 2012 6;486(7401):105-8. Available from: https://www.nature.com/articles/nature11118.

[5] Oaks JL, Gilbert M, Virani MZ, Watson RT, Meteyer CU, Rideout BA, et al. Diclofenac residues as the cause of vulture population decline in Pakistan. Nature. 2004 2;427(6975):630-3. Available from: https://www.nature.com/articles/nature02317.

[6] Sgolastra F, Medrzycki P, Bortolotti L, Maini S, Porrini C, Simon-Delso N, et al. Bees and pesticide regulation: Lessons from the neonicotinoid experience. Biological Conservation. 2020 1;241:108356. Available from: https://www.sciencedirect.com/science/article/pii/S0006320719310912#ab0005.

[7] Trasande L, Zoeller RT, Hass U, Kortenkamp A, Grandjean P, Myers JP, et al. Estimating Burden and Disease Costs of Exposure to Endocrine-Disrupting Chemicals in the European Union. The Journal of Clinical Endocrinology & Metabolism. 2015 4;100(4):1245-55. Available from: https://dx.doi.org/10.1210/jc.2014-4324.

[8] Attina TM, Hauser R, Sathyanarayana S, Hunt PA, Bourguignon JP, Myers JP, et al. Exposure to endocrine-disrupting chemicals in the USA: a population-based disease burden and cost analysis. The Lancet Diabetes and Endocrinology. 2016 12;4(12):996-1003. Available from: https://www.thelancet.com/action/showFullText?pii=S2213858716302753https://www.thelancet.com/action/showAbstract?pii=S2213858716302753https://www.thelancet.com/journals/landia/article/PIIS2213-8587(16)30275-3/abstract.

[9] Naidu R, Biswas B, Willett IR, Cribb J, Kumar Singh B, Paul Nathanail C, et al. Chemical pollution: A growing peril and potential catastrophic risk to humanity. Environment International. 2021 11;156:106616. Available from: https://www.sciencedirect.com/science/article/pii/S0160412021002415#b0080.

[10] Council of the European Communities. Council Directive 67/548/EEC of 27 June 1967 on the approximation of laws, regulations and administrative provisions relating to the classification, packaging and labelling of dangerous substances. Official Journal of the European Communities; 1967. 196. Available from: https://eur-lex.europa.eu/eli/dir/1967/548/oj.

[11] Whittaker MH. Risk Assessment and Alternatives Assessment: Comparing Two Methodologies. Risk Analysis. 2015 12;35(12):2129-36. Available from: /doi/pdf/10.1111/risa.12549https://onlinelibrary.wiley.com/doi/abs/10.1111/risa.12549https://onlinelibrary.wiley.com/doi/10.1111/risa.12549.

[12] Christensen FM, Eisenreich SJ, Rasmussen K, Sintes JR, Sokull-Kluettgen B, Van De Plassche EJ. European experience in chemicals management: Integrating science into policy. Environmental Science and Technology. 2011 1;45(1):80-9. Available from: /doi/pdf/10.1021/es101541b?download=true.

[13] Botos A, Graham JD, Illes Z. Industrial chemical regulation in the European Union and the United States: a comparison of REACH and the amended TSCA#. Journal of Risk Research. 2019 10;22(10):1187-204. Available from: https://www.tandfonline.com/doi/pdf/10.1080/13669877.2018.1454495.

[14] European Commission. Chemicals Strategy for Sustainability Towards a Toxic-Free Environment. Brussels; 2020. Available from: https://circabc.europa.eu/ui/group/8ee3c69a-bccb-4f22-89ca-277e35de7c63/library/dd074f3d-0cc9-4df2-b056-dabcacfc99b6/details?download=true.

[15] European Comission. Aquatic toxicity;. Available from: https://joint-research-centre.ec.europa.eu/projects-and-activities/reference-and-measurement/european-union-reference-laboratories/eu-reference-laboratory-alternatives-animal-testing-eurl-ecvam/alternative-methods-toxicity-testing/validated-test-methods-health-effects/aquatic-toxicity_en.

[16] Arvidsson R. Life Cycle Assessment and Risk Assessment of Manufactured Nanomaterials. Nanoengineering: Global Approaches to Health and Safety Issues. 2015 6:225-56.

[17] Cherkasov A, Muratov EN, Fourches D, Varnek A, Baskin II, Cronin M, et al. QSAR modeling: Where have you been? Where are you going to? Journal of Medicinal Chemistry. 2014 6;57(12):4977-5010. Available from: /doi/pdf/10.1021/jm4004285.

[18] Muratov EN, Bajorath J, Sheridan RP, Tetko IV, Filimonov D, Poroikov V, et al. QSAR without borders. Chemical Society Reviews. 2020 6;49(11):3525-64. Available from: https://pubs.rsc.org/en/content/articlehtml/2020/cs/d0cs00098ahttps://pubs.rsc.org/en/content/articlelanding/2020/cs/d0cs00098a.

[19] Cavasotto CN, Scardino V. Machine Learning Toxicity Prediction: Latest Advances by Toxicity End Point. ACS Omega. 2022 12;7(51):47536-46. Available from: /doi/pdf/10.1021/acsomega.2c05693.

[20] Guo W, Liu J, Dong F, Song M, Li Z, Khan MKH, et al. Review of machine learning and deep learning models for toxicity prediction. Experimental Biology and Medicine. 2023 11;248(21):1952. Available from: https://pmc.ncbi.nlm.nih.gov/articles/PMC10798180/.

[21] Devlin J, Chang MW, Lee K, Google KT, Language AI. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv preprint. 2018 10. Available from: https://github.com/tensorflow/tensor2tensor.

[22] Gururangan S, Marasovic A, Swayamdipta S, Lo K, Beltagy I, Downey D, et al. Don't stop pretraining: Adapt language models to domains and tasks. Proceedings of the Annual Meeting of the Association for Computational Linguistics. 2020 4:8342-60. Available from: https://arxiv.org/pdf/2004.10964.

[23] Johnson AC, Jin X, Nakada N, Sumpter JP. Learning from the past and considering the future of chemicals in the environment. Science. 2020 1;367(6476):384-7. Available from: https://doi.org/10.1126/science.aay6637.

[24] Panico A, Serio F, Bagordo F, Grassi T, Idolo A, De Giorgi M, et al. Skin Safety and Health Prevention: an Overview of Chemicals in Cosmetic Products. Journal of Preventive Medicine and Hygiene. 2019 4;60(1):E50-0. Available from: https://www.jpmh.org/index.php/jpmh/article/view/1080.

[25] Barratt MD. Prediction of toxicity from chemical structure. Cell Biology and Toxicology. 2000;16(1):1-13. Available from: https://link.springer.com/article/10.1023/A:1007676602908.

[26] Encyclopædia Britannica. Chemical formula | Definition, Types, Examples, & Facts | Britannica; 2022. Available from: https://www.britannica.com/science/chemical-formula.

[27] Britannica Academic. chemical bonding – Britannica Academic;. Available from: https://academic-eb-com.eu1.proxy.openathens.net/levels/collegiate/article/chemical-bonding/110108.

[28] Weininger D. SMILES, a Chemical Language and Information System: 1: Introduction to Methodology and Encoding Rules. Journal of Chemical Information and Computer Sciences. 1988 2;28(1):31-6. Available from: /doi/pdf/10.1021/ci00057a005.

[29] Apodaca R, O'Boyle N, Dalke A, van Drie J, Ertl P, Hutchison G, et al.. OpenSMILES specification; 2016. Available from: http://opensmiles.org/opensmiles.pdf.

[30] van Halteren H, editor. Syntactic Wordclass Tagging. vol. 9 of Text, Speech and Language Technology. 1st ed. Dordrecht: Kluwer Academic Publishers; 1999. Available from: http://link.springer.com/10.1007/978-94-015-9273-4.

[31] Mehlig B. Machine Learning with Neural Networks. Padstow: Cambridge University Press; 2021.

[32] Campesato O. Large language models : an introduction. Boston: Mercury Learning and Information; 2024.

[33] Kamath U, Keenan K, Somers G, Sorenson S. Large Language Models: A Deep Dive. 1st ed. Springer Cham; 2024.

[34] Mswahili ME, Jeong YS. Transformer-based models for chemical SMILES representation: A comprehensive literature review. Heliyon. 2024 10;10(20):e39038. Available from: https://www.sciencedirect.com/science/article/pii/S2405844024150694.

[35] Blanco-Justicia A, Domingo-Ferrer J, Martínez S, Sánchez D, Flanagan A, Tan KE. Achieving security and privacy in federated learning systems: Survey, research challenges and future directions. Engineering Applications of Artificial Intelligence. 2021 11;106:104468. Available from: https://www.sciencedirect.com/science/article/pii/S095219762100316X.

[36] Blanco-Justicia A, Domingo-Ferrer J, Martínez S, Sánchez D, Flanagan A, Tan KE. Achieving security and privacy in federated learning systems: Survey, research challenges and future directions. Engineering Applications of Artificial Intelligence. 2021 11;106.

[37] Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv preprint. 2019. Available from: https://github.com/pytorch/fairseq.

[38] Chithrananda S, Grand G, Deepchem BR. ChemBERTa: Large-Scale Self-Supervised Pretraining for Molecular Property Prediction. arXiv preprint. 2020.

[39] National Center for Biotechnology Information. About - PubChem;. Available from: https://pubchem.ncbi.nlm.nih.gov/docs/about.

[40] Chemical Abstracts Service. CAS REGISTRY | CAS; 2025. Available from: https://www.cas.org/cas-data/cas-registry.

[41] Stewart K. Mean squared error (MSE) | Definition, Formula, Interpretation, & Facts | Britannica; 2025. Available from: https://www.britannica.com/science/mean-squared-error.

[42] Hastie T, Tibshirani R, Friedman J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd ed. New York: Springer Science+Business Media, LLC; 2009.

[43] Stewart K. Pearson's correlation coefficient | Definition, Formula, & Facts | Britannica; 2025. Available from: https://www.britannica.com/topic/Pearsons-correlation-coefficient.

[44] Haug MG. measure of association – Britannica Academic;. Available from: https://academic-eb-com.eu1.proxy.openathens.net/levels/collegiate/article/measure-of-association/627729#335773.toc.

[45] Dodge Y. Spearman Rank Correlation Coefficient. New York, NY: Springer, New York, NY; 2008. Available from: https://link.springer.com/rwe/10.1007/978-0-387-32833-1_379.

[46] Lötsch J, Kringel D, Ultsch A. Revisiting Fold-Change Calculation: Preference for Median or Geometric Mean over Arithmetic Mean-Based Methods. Biomedicines. 2024 8;12(8):1639. Available from: https://pmc.ncbi.nlm.nih.gov/articles/PMC11352044/.

A

Appendix A

In this Appendix, the data used to plot Figure 4.1 and 4.3 are listed. In addition, complete tables showing the means of the median $\log_{10}$ prediction error and the 90^{th} percentile of the $\log_{10}$ prediction error across embedding sizes, encoder layers, and attention heads are provided. Additionally, figures that illustrate the over- and underestimation of chemical toxicity values, based on a 1000-fold change threshold for all models, are provided

A.1 Median $\log_{10}$ Prediction Error and Standard Deviation for All Models

Table A.1: Median $\log_{10}$ prediction error and standard deviation for all 144 non-pre-trained only models covering all hyperparameter combinations.

Embedding Size	# Encoder Layers	# Attention Heads	Median $\log_{10}$ Prediction Error [$\log_{10}$ mg/L]	Standard Deviation
64	1	1	0.6437	0.0183
64	1	4	0.6374	0.0307
64	1	8	0.6365	0.0203
64	1	16	0.6397	0.0059
64	1	32	0.6438	0.0220
64	1	64	0.6438	0.0197
64	3	1	0.6268	0.0177
64	3	4	0.6295	0.0248
64	3	8	0.6383	0.0268
64	3	16	0.6291	0.0173
64	3	32	0.6327	0.0122
64	3	64	0.6321	0.0184
64	6	1	0.6060	0.0222
64	6	4	0.6202	0.0225
64	6	8	0.6157	0.0202
64	6	16	0.6229	0.0201
64	6	32	0.6139	0.0188
64	6	64	0.6327	0.0271
64	8	1	0.6176	0.0178
64	8	4	0.6112	0.0218

64	8	8	0.6241	0.0212
64	8	16	0.6160	0.0209
64	8	32	0.6268	0.0203
64	8	64	0.6288	0.0219
64	10	1	0.6157	0.0135
64	10	4	0.6043	0.0212
64	10	8	0.6122	0.0305
64	10	16	0.6223	0.0197
64	10	32	0.6129	0.0293
64	10	64	0.6226	0.0224
64	12	1	0.6070	0.0102
64	12	4	0.6194	0.0169
64	12	8	0.6125	0.0127
64	12	16	0.6152	0.0198
64	12	32	0.6006	0.0227
64	12	64	0.6109	0.0282
64	14	1	0.6084	0.0214
64	14	4	0.6178	0.0190
64	14	8	0.6090	0.0099
64	14	16	0.6206	0.0202
64	14	32	0.6065	0.0144
64	14	64	0.6203	0.0212
64	16	1	0.5990	0.0300
64	16	4	0.6105	0.0142
64	16	8	0.6034	0.0229
64	16	16	0.6160	0.0148
64	16	32	0.6158	0.0372
64	16	64	0.6096	0.0203
256	1	1	0.6320	0.0257
256	1	4	0.6150	0.0167
256	1	8	0.6221	0.0076
256	1	16	0.6200	0.0189
256	1	32	0.6099	0.0225
256	1	64	0.6085	0.0257
256	3	1	0.6140	0.0250
256	3	4	0.6040	0.0149
256	3	8	0.6059	0.0228
256	3	16	0.6072	0.0209
256	3	32	0.6038	0.0325
256	3	64	0.6039	0.0239
256	6	1	0.5983	0.0164
256	6	4	0.5925	0.0243
256	6	8	0.5932	0.0122
256	6	16	0.5997	0.0261
256	6	32	0.6040	0.0134
256	6	64	0.6041	0.0055

256	8	1	0.6005	0.0164
256	8	4	0.6024	0.0112
256	8	8	0.6013	0.0281
256	8	16	0.6047	0.0230
256	8	32	0.5918	0.0238
256	8	64	0.6062	0.0177
256	10	1	0.5977	0.0332
256	10	4	0.5896	0.0189
256	10	8	0.5909	0.0329
256	10	16	0.6032	0.0311
256	10	32	0.6106	0.0250
256	10	64	0.5909	0.0204
256	12	1	0.5960	0.0323
256	12	4	0.6012	0.0234
256	12	8	0.5895	0.0279
256	12	16	0.6041	0.0193
256	12	32	0.6105	0.0216
256	12	64	0.5976	0.0155
256	14	1	0.5961	0.0251
256	14	4	0.5943	0.0209
256	14	8	0.5932	0.0188
256	14	16	0.6007	0.0122
256	14	32	0.5993	0.0228
256	14	64	0.5978	0.0156
256	16	1	0.6017	0.0129
256	16	4	0.6046	0.0177
256	16	8	0.6024	0.0316
256	16	16	0.5993	0.0178
256	16	32	0.6039	0.0231
256	16	64	0.6081	0.0240
768	1	1	0.6096	0.0267
768	1	4	0.6516	0.0173
768	1	8	0.6373	0.0073
768	1	16	0.6177	0.0208
768	1	32	0.6042	0.0113
768	1	64	0.6221	0.0235
768	3	1	0.6153	0.0270
768	3	4	0.6153	0.0260
768	3	8	0.6004	0.0140
768	3	16	0.5978	0.0251
768	3	32	0.6127	0.0270
768	3	64	0.5969	0.0129
768	6	1	0.6116	0.0070
768	6	4	0.6033	0.0243
768	6	8	0.5971	0.0185
768	6	16	0.5988	0.0233

768	6	32	0.5917	0.0118
768	6	64	0.5960	0.0164
768	8	1	0.5982	0.0127
768	8	4	0.5974	0.0176
768	8	8	0.6012	0.0189
768	8	16	0.6052	0.0198
768	8	32	0.5979	0.0097
768	8	64	0.6028	0.0199
768	10	1	0.5896	0.0132
768	10	4	0.5977	0.0217
768	10	8	0.5983	0.0267
768	10	16	0.5912	0.0244
768	10	32	0.5801	0.0214
768	10	64	0.6000	0.0184
768	12	1	0.5905	0.0225
768	12	4	0.5979	0.0215
768	12	8	0.5999	0.0137
768	12	16	0.5913	0.0257
768	12	32	0.6020	0.0330
768	12	64	0.6063	0.0248
768	14	1	0.5993	0.0210
768	14	4	0.5906	0.0236
768	14	8	0.5869	0.0303
768	14	16	0.5933	0.0179
768	14	32	0.6000	0.0230
768	14	64	0.5925	0.0283
768	16	1	0.6020	0.0163
768	16	4	0.5946	0.0241
768	16	8	0.5860	0.0139
768	16	16	0.6005	0.0313
768	16	32	0.5922	0.0141
768	16	64	0.5951	0.0222

A.2 90^{th} Percentile of $\log_{10}$ Prediction Errors and Standard Deviations for All Models

Table A.2: Median $\log_{10}$ prediction error and standard deviation for all 144 non-pre-trained only models covering all hyperparameter combinations.

Embedding Size	# Encoder Layers	# Attention Heads	90^{th} Percentile of $\log_{10}$ Prediction Error [$\log_{10}$ mg/L]	Standard Deviation
64	1	1	1.6357	0.0495
64	1	4	1.6463	0.0346
64	1	8	1.6527	0.0457

64	1	16	1.6324	0.0509
64	1	32	1.6284	0.0339
64	1	64	1.6259	0.0478
64	3	1	1.6185	0.0779
64	3	4	1.6180	0.0728
64	3	8	1.6051	0.0259
64	3	16	1.6275	0.0226
64	3	32	1.6143	0.0410
64	3	64	1.6141	0.0428
64	6	1	1.6059	0.0500
64	6	4	1.6000	0.0445
64	6	8	1.6192	0.0365
64	6	16	1.6268	0.0296
64	6	32	1.6004	0.0358
64	6	64	1.6134	0.0290
64	8	1	1.6152	0.0363
64	8	4	1.6117	0.0586
64	8	8	1.6022	0.0812
64	8	16	1.5812	0.0557
64	8	32	1.5817	0.0448
64	8	64	1.5842	0.0549
64	10	1	1.6125	0.0220
64	10	4	1.6096	0.0562
64	10	8	1.6241	0.0158
64	10	16	1.6196	0.0429
64	10	32	1.5914	0.0232
64	10	64	1.5707	0.0382
64	12	1	1.6018	0.0443
64	12	4	1.5997	0.0293
64	12	8	1.5956	0.0583
64	12	16	1.5829	0.0523
64	12	32	1.5707	0.0531
64	12	64	1.5495	0.0446
64	14	1	1.5949	0.0547
64	14	4	1.6041	0.0232
64	14	8	1.5832	0.0276
64	14	16	1.5847	0.0616
64	14	32	1.5619	0.0486
64	14	64	1.5636	0.0503
64	16	1	1.5904	0.0674
64	16	4	1.5646	0.0687
64	16	8	1.5893	0.0184
64	16	16	1.5817	0.0300
64	16	32	1.5660	0.0418
64	16	64	1.5693	0.0235
256	1	1	1.6193	0.0351

256	1	4	1.6152	0.0342
256	1	8	1.6004	0.0415
256	1	16	1.5740	0.0253
256	1	32	1.5956	0.0477
256	1	64	1.5572	0.0545
256	3	1	1.5910	0.0740
256	3	4	1.6060	0.0315
256	3	8	1.5990	0.0728
256	3	16	1.5715	0.0795
256	3	32	1.6036	0.0652
256	3	64	1.5912	0.0303
256	6	1	1.5956	0.0368
256	6	4	1.5745	0.0432
256	6	8	1.5874	0.0529
256	6	16	1.5738	0.0709
256	6	32	1.5500	0.0442
256	6	64	1.5567	0.0540
256	8	1	1.5904	0.0538
256	8	4	1.5898	0.0515
256	8	8	1.5911	0.0525
256	8	16	1.5585	0.0484
256	8	32	1.5711	0.0415
256	8	64	1.5729	0.0287
256	10	1	1.5734	0.0352
256	10	4	1.5501	0.0387
256	10	8	1.5640	0.0589
256	10	16	1.5650	0.0395
256	10	32	1.5724	0.0292
256	10	64	1.5609	0.0552
256	12	1	1.5656	0.0711
256	12	4	1.5708	0.0572
256	12	8	1.5608	0.0567
256	12	16	1.5640	0.0460
256	12	32	1.5786	0.0389
256	12	64	1.5523	0.0607
256	14	1	1.5661	0.0312
256	14	4	1.5688	0.0369
256	14	8	1.5588	0.0653
256	14	16	1.5586	0.0426
256	14	32	1.5389	0.0318
256	14	64	1.5455	0.0358
256	16	1	1.5633	0.0495
256	16	4	1.5761	0.0571
256	16	8	1.5917	0.0523
256	16	16	1.5943	0.0385
256	16	32	1.5767	0.0399

256	16	64	1.5662	0.0377
768	1	1	1.6559	0.0352
768	1	4	1.6019	0.0419
768	1	8	1.6098	0.0351
768	1	16	1.5632	0.0323
768	1	32	1.5916	0.0320
768	1	64	1.6173	0.0505
768	3	1	1.6010	0.0344
768	3	4	1.5763	0.0461
768	3	8	1.5858	0.0160
768	3	16	1.5882	0.0841
768	3	32	1.5648	0.0355
768	3	64	1.5879	0.0259
768	6	1	1.5868	0.0515
768	6	4	1.5402	0.0335
768	6	8	1.5655	0.0612
768	6	16	1.5453	0.0462
768	6	32	1.5391	0.0354
768	6	64	1.5359	0.0399
768	8	1	1.5672	0.0346
768	8	4	1.5373	0.0529
768	8	8	1.5300	0.0424
768	8	16	1.5564	0.0378
768	8	32	1.5368	0.0249
768	8	64	1.5559	0.0369
768	10	1	1.5575	0.0481
768	10	4	1.5443	0.0352
768	10	8	1.5421	0.0425
768	10	16	1.5436	0.0420
768	10	32	1.5472	0.0602
768	10	64	1.5538	0.0291
768	12	1	1.5882	0.0692
768	12	4	1.5561	0.0318
768	12	8	1.5470	0.0530
768	12	16	1.5522	0.0471
768	12	32	1.5477	0.0491
768	12	64	1.5297	0.0432
768	14	1	1.5735	0.0285
768	14	4	1.5723	0.0387
768	14	8	1.5394	0.0234
768	14	16	1.5575	0.0331
768	14	32	1.5493	0.0411
768	14	64	1.5330	0.0234
768	16	1	1.5805	0.0430
768	16	4	1.5734	0.0384
768	16	8	1.5499	0.0584

768	16	16	1.5619	0.0612
768	16	32	1.5685	0.0237
768	16	64	1.5621	0.0475

A.3 Median and 90th Percentile Prediction Errors Grouped by Model Hyperparameters

A.3.1 Prediction Errors by Number of Encoder Layers

Table A.3: Mean and standard deviation of median $\log_{10}$ prediction errors grouped by encoder layers, i.e., averaged over embedding sizes and attention head counts for each encoder layer configuration.

Encoder Layers	Mean of Median $\log_{10}$ Prediction Error [$\log_{10}$ mg/L]	Standard Deviation
1	0.6275	0.0148
3	0.6148	0.0134
6	0.6057	0.0116
8	0.6075	0.0109
10	0.6016	0.0122
12	0.6029	0.0086
14	0.6015	0.0102
16	0.6025	0.0078

Table A.4: Mean and standard deviation of 90th percentile of $\log_{10}$ prediction errors grouped by encoder layers, i.e., averaged over embedding sizes and attention head counts for each encoder layer configuration.

Encoder Layers	Mean of 90th Percentile of $\log_{10}$ Prediction Error [$\log_{10}$ mg/L]	Standard Deviation
1	1.6124	0.0287
3	1.5980	0.0173
6	1.5787	0.0295
8	1.5741	0.0249
10	1.5723	0.0274
12	1.5674	0.0203
14	1.5641	0.0194
16	1.5737	0.0123

A.3.2 Prediction Errors by Number of Attention Heads

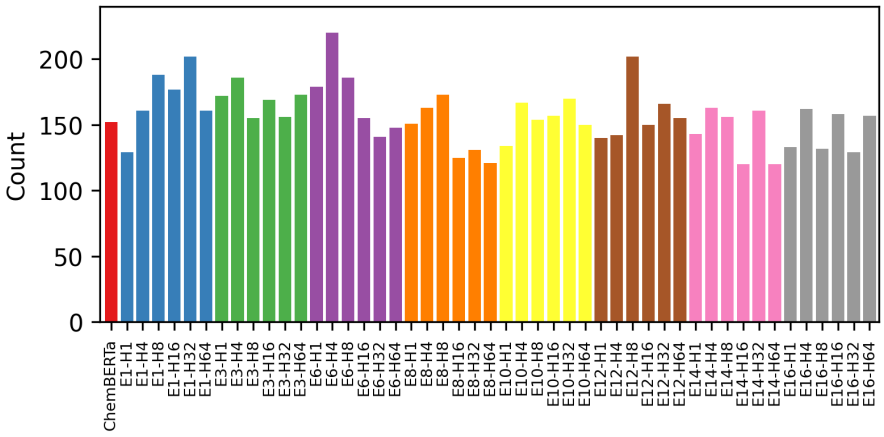
Table A.5: Mean and standard deviation of median $\log_{10}$ prediction errors grouped by attention heads, i.e., averaged over embedding sizes and encoder layer counts for each attention head configuration.

Attention Heads	Mean of Median $\log_{10}$ Prediction Error [$\log_{10}$ mg/L]	Standard Deviation
1	0.6074	0.0131
4	0.6084	0.0154
8	0.6066	0.0156
16	0.6090	0.0127
32	0.6070	0.0137
64	0.6096	0.0143

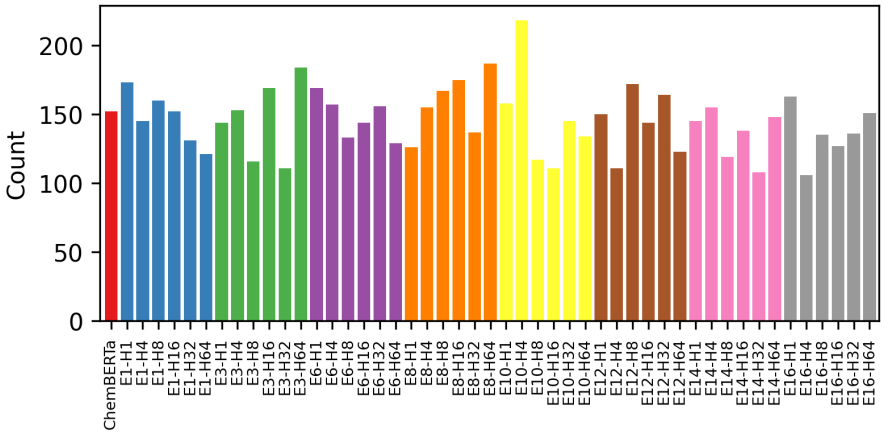
Table A.6: Mean and standard deviation of 90^{th} percentile of $\log_{10}$ prediction errors grouped by attention heads, i.e., averaged over embedding sizes and encoder layer counts for each attention head configuration.

# Attention Heads	Mean of 90^{th} Percentile of $\log_{10}$ Prediction Error [$\log_{10}$ mg/L]	Standard Deviation
1	1.5938	0.0243
4	1.5836	0.0280
8	1.5831	0.0301
16	1.5777	0.0259
32	1.5728	0.0247
64	1.5696	0.0269

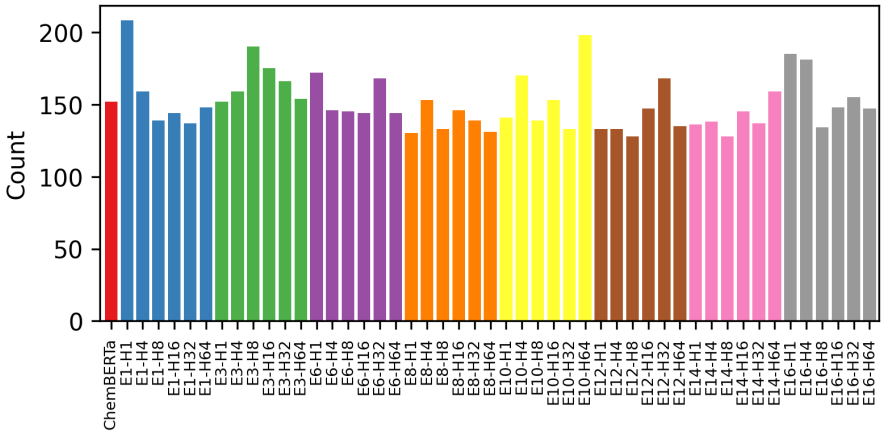
A.4 Fold Change Plots with 1000-Fold Threshold for All Models



(a) Embedding size 64.

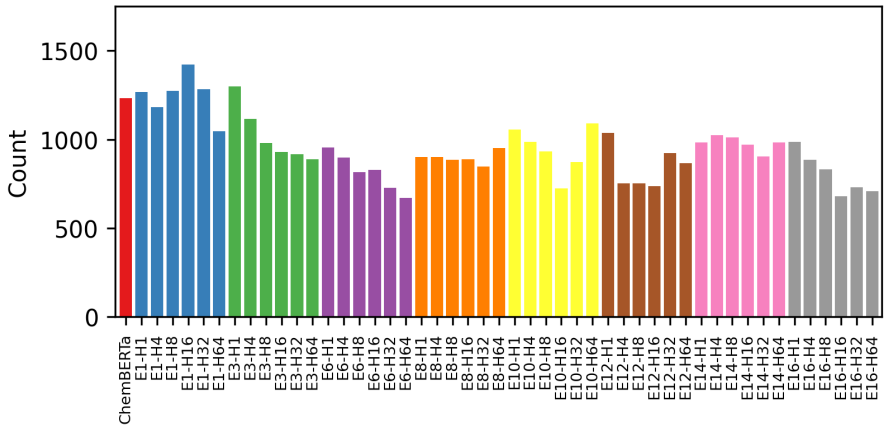


(b) Embedding size 256.

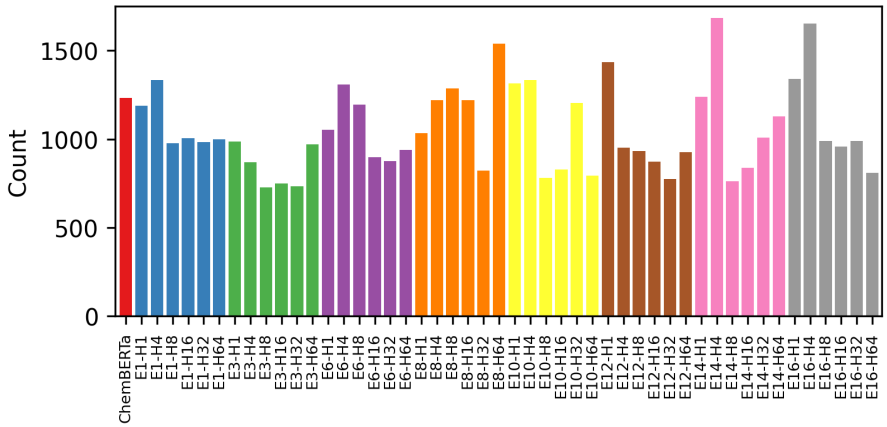


(c) Embedding size 768.

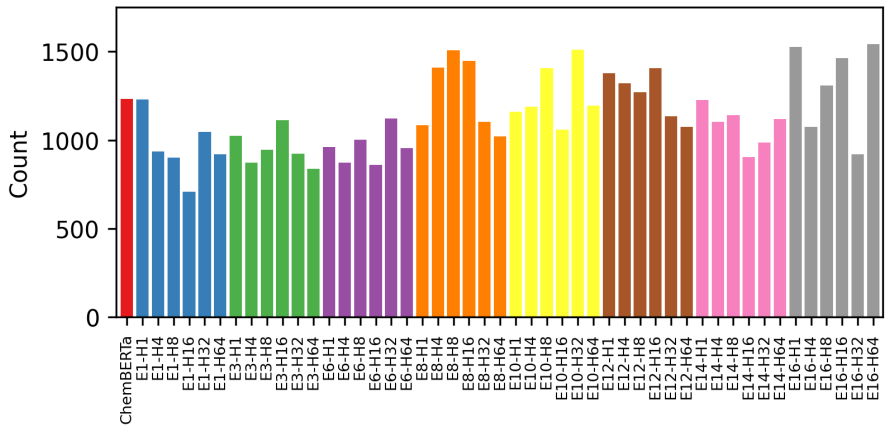
Figure A.1: Fold change with a 1000-fold change threshold for under-prediction of chemical toxicity values, i.e., predicted toxicity value $<$ true toxicity value, for all trained models.



(a) Embedding size 64.



(b) Embedding size 256.



(c) Embedding size 768.

Figure A.2: Fold change with a 1000-fold change threshold for over-prediction of chemical toxicity values, i.e., predicted toxicity value > true toxicity value, for all trained models.

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY