

Universidade do Minho
Escola de Engenharia

Diogo da Silva Rebelo

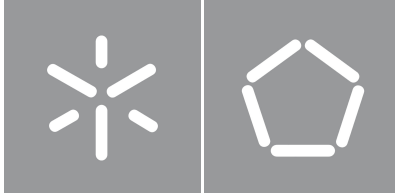
**Development of a machine
learning-based application to predict
the toxicity of chemical compounds**

**Development of a machine learning-based application
to predict the toxicity of chemical compounds**

Diogo da Silva Rebelo

UMinho | 2024

October 2024



University of Minho
School of Engineering

Diogo da Silva Rebelo

**Development of a machine
learning-based application to predict
the toxicity of chemical compounds**

Masters' Dissertation
Masters in Informatics Engineering

Dissertation supervised by
Miguel Francisco Almeida Pereira Rocha,
Associate Professor, Department of Informatics,
University of Minho

Copyright and Terms of Use for Third Party Work

This dissertation reports on academic work that can be used by third parties as long as the internationally accepted standards and good practices are respected concerning copyright and related rights.

This work can thereafter be used under the terms established in the license below.

Readers needing authorization conditions not provided for in the indicated licensing should contact the author through the RepositóriUM of the University of Minho.

License granted to users of this work:



Attribution-NonCommercial-ShareAlike 4.0 International

CC BY-NC-SA 4.0 DEED

<https://creativecommons.org/licenses/by-nc-sa/4.0/>

Acknowledgements

First and foremost, I would like to express my deep gratitude to Professor Miguel Rocha. His unwavering support, insightful feedback, and commitment to my success have been invaluable throughout this journey. This entire work would not have been possible without his guidance.

I extend my sincere thanks to OmniumAI spin-off, which has been my second home for over a year. I am particularly grateful to João Correia, whose close mentorship and expertise in the field have been crucial to my progress. I also would like to acknowledge Diogo, Marta, Leonardo and Guilherme, for all their help and contribution for a healthy and promising corporate environment.

I owe a great debt of gratitude to my family, especially my parents, who have always supported my academic pursuits by providing all the necessary resources and unwavering belief in me. To my brothers, André, Rodrigo, and Miguel, I extend my heartfelt thanks for their assistance, sharing their knowledge in programming and other areas, and offering the best advice. Your support has been one of the most important parts of this journey.

To my best friend, Lídia Sousa, my little monkey, I offer words of love and gratitude. Her life advice, support, and motivation have been essential, making her like the fourth sister I never had.

Lastly, I would like to acknowledge my dear friends, my *amigolas*, each of whom contributed in their unique way to make this journey more enjoyable and less challenging. Your friendship has made my experience richer and more memorable.

All of you will always hold a special place in my heart.

“Not everything that is faced can be changed, but nothing can be changed until it is faced.”

— James Baldwin

Statement of Integrity

I hereby declare having conducted this academic work with integrity.

I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

University of Minho, Braga, october 2024

Diogo da Silva Rebelo

Abstract

The prediction of the toxicity of chemical compounds is a crucial task in drug development and chemical safety assessment. In this study, are implemented and compared both *singletask* and *multitask* machine learning (ML) approaches to predict three toxicity endpoints: mutagenicity, carcinogenicity, and genotoxicity. The comparisons included models with and without cross-validation (CV), aiming to determine the most effective approach for predicting these biological phenomena. The training process was supported by the internal pipelines of *OmniumAI*, a spin-off of the University of Minho.

For hyperparameter optimization, the Tree-structured Parzen Estimator (TPE) algorithm was employed, with the primary goal of maximizing the *balanced accuracy* metric, given the imbalanced nature of the data. Various predictive models were explored, including neural networks and classical ML techniques, with a robust number of trials for each task. The optimization process was managed using an SQLite database, ensuring persistence of the best-performing models for subsequent analysis and testing.

The model was integrated into a final application that allows predictions to be generated using the trained models. This application was developed in parallel, utilizing React JavaScript and Python as the main technologies.

The results revealed that the *multitask* approach outperformed the *singletask* methods in the test set, indicating that toxicity information for one endpoint can be useful in predicting another. The inclusion of *featurizers* and *standardizers* in the models did not lead to significant performance gains, reinforcing the robustness of certain neural network architectures like convolutional neural networks (TextCNN and CNN) and graph-based models (D-MPNN and Pathgn), which do not heavily rely on complex preprocessing techniques.

As a final product of this project, we developed not only a set of robust models—with balanced accuracy reaching 95% (validation) and ROC AUC values of up to 92% (test)—but also an intuitive website.

Keywords AutoML, Machine Learning, Singletask and Multitask learning, Software Development, Toxicity

Resumo

A previsão da toxicidade de compostos químicos é uma tarefa fundamental no desenvolvimento e avaliação da segurança de fármacos. Neste âmbito, implementaram-se e compararam-se abordagens de *machine learning* (ML) *singletask* e *multitask* para prever três *endpoints* de toxicidade: mutagenicidade, carcinogenicidade e genotoxicidade. As comparações incluíram modelos com e sem validação cruzada (CV), com o objetivo de determinar a abordagem mais eficaz para a previsão destes fenómenos biológicos. Este treino foi auxiliado pelos *pipelines* internos da *OmniumAI*, uma *spin-off* da Universidade do Minho.

Para otimização dos hiperparâmetros, utilizou-se o algoritmo *Tree-structured Parzen Estimator* (TPE), com maximização da *balanced accuracy*, dada a natureza desbalanceada dos dados. Exploraram-se diversos modelos preditivos, incluindo redes neurais e técnicas clássicas de ML, com um número robusto de ensaios.

O modelo foi integrado numa aplicação final que permite a geração de previsões, utilizando os modelos treinados. Esta aplicação foi desenvolvida paralelamente e utiliza *React JavaScript* e *Python* como tecnologias principais.

Os resultados revelaram que a abordagem *multitask* superou a *singletask* nos dados de teste, indicando que a toxicidade para um *endpoint* pode ser utilizada para prever outro. Além disso, observou-se que a inclusão de *featurizers* e *standardizers* nos modelos não resultou em ganhos muito expressivos de desempenho, reforçando a robustez de certas arquiteturas de redes neurais (TextCNN e CNN) e à base de grafos (D-MPNN e Patgn), que não dependem fortemente de pré-processamentos complexos.

Como produto final do desenvolvimento deste projeto, resultou, assim, não só um conjunto de modelos robustos- com uma *balanced accuracy* a atingir 95% (validação) valores de *ROC AUC* de 92% (teste) -, mas também um *website* intuitivo.

Palavras-chave *AutoML*, Desenvolvimento de *Software*, Machine Learning, Singletask and Multitask learning, Toxicidade

Contents

1	Introduction	1
1.1	Context and motivation	1
1.2	Objectives	1
1.3	Thesis Structure	2
2	Toxicity in Toxicological Sciences	4
2.1	Determining factors	5
2.1.1	Intrinsic	5
2.1.2	Dose	5
2.1.3	Exposure conditions	6
2.1.4	Host dose-response	7
2.2	Toxicity testing	11
2.2.1	<i>In vivo</i> testing	11
2.2.2	<i>In vitro</i> testing	11
2.3	Toxic endpoints	12
2.3.1	Quantifying toxic endpoints	12
2.3.2	Describing major organ-specific toxicities	13
2.4	Databases	15
3	Chemoinformatics	17
3.1	Molecular Descriptors	18
3.1.1	Physicochemical descriptors	19
3.1.2	Structural descriptors	20
3.2	Molecular Representations	21
3.2.1	Line Notations	21
3.2.2	Molecular Graphs	27

3.2.3	Three-Dimensional Representations	28
3.2.4	Embeddings	29
3.3	Databases	30
3.4	Software, packages and tools	32
4	Machine Learning	33
4.1	Types of Machine Learning	34
4.1.1	Supervised Learning	34
4.1.2	Unsupervised Learning	34
4.1.3	Reinforcement Learning	35
4.1.4	Generative Learning	35
4.2	Basic principles	35
4.2.1	Concepts and Terms	35
4.2.2	Performance Measure	37
4.3	Traditional Machine Learning	40
4.3.1	Supervised Learning	40
4.3.2	Unsupervised Learning	44
4.4	Deep Learning	46
4.4.1	Architectures	46
4.5	Software, packages and tools	51
5	Machine Learning for Toxicity Prediction	52
5.1	<i>In silico</i> Modeling Methods	52
5.2	Existent approaches for endpoint prediction	55
5.2.1	Endpoint-specific	55
5.2.2	General Toxicity	60
6	Methods	64
6.1	Methodology	64
6.2	Toxicity of compounds in <i>OpenFoodTox</i> dataset	66
6.2.1	Data Collection and Processing using Medallion Architecture	67
6.2.2	Preliminary Data Visualization and Exploration	69
6.2.3	Split of dataset	73
6.2.4	Model training	74

6.2.5	Overcoming the impact of imbalanced data	76
7	Results and Discussion	77
7.1	Unsupervised Learning Analysis	77
7.1.1	Principal Component Analysis	77
7.1.2	T-Distributed Stochastic Neighbor Embedding	79
7.1.3	K-means clustering	80
7.2	Singletask Learning	82
7.2.1	Without cross-validation	82
7.2.2	With cross-validation	85
7.2.3	Comparative Analysis (with and without cross-validation)	88
7.3	Multitask Learning	89
7.3.1	With cross-validation	89
7.4	Comparative Analysis (Singletask vs. Multitask)	90
7.4.1	Comparative Analysis with Literature	90
8	Application Development	92
8.1	Concept and Features	92
8.2	Technological stack	93
8.3	Architecture and Design	93
8.3.1	System Overview	93
8.3.2	Data/work Flow and Interaction	96
9	Conclusion and Future work	98
9.1	Main Conclusions	98
9.2	Future Work	99

List of Figures

1	Dose-response curves of chemicals A and B.	9
2	Shape comparison of dose-response curve types.	10
3	Four ways of depicting propane.	22
4	Caffeine molecule and its various representations.	24
5	Example of primary layers within the standard InChI notation.	25
6	Molecular fingerprints of Salicylic acid.	26
7	Adjacency matrices and Connection Table (CT) of salicylic acid.	28
8	3D Cartesian model of ethanol.	29
9	Diagram of Machine Learning (ML) traditional classes and their goals.	33
10	Support vectors and margin of Support Vector Machine (SVM).	40
11	Example of decision tree for the ripe-target dataset.	41
12	The M-P neuron model.	43
13	Small Artificial Neural Network (ANN) example.	43
14	Basic Convolutional Neural Network (CNN) architecture.	47
15	Recurrent Neural Network (RNN)/Long Short-Term Memory (LSTM)/Gated Recurrent Units (GRU) blocks.	50
16	Conceptual tree of <i>in silico</i> technologies.	53
17	Phases of the adopted methodology.	65
18	Overview of the OpenFoodTox data	66
19	Medallion architecture stages.	67
20	Distribution of data for each endpoint.	72
21	Several study categories in the dataset.	73
22	Evolution of the cumulative variance percentage with the number of components.	78

23	2D plot between components 1 and 2 for each data.	78
24	Evolution of perplexity values with KL Divergence.	79
25	T-Distributed Stochastic Neighbor Embedding (t-SNE) first two components for each data.	80
26	K-Means Clustering (K-means) components 1 and 2 for each data, with 2 clusters.	80
27	Evolution of number of clusters with WCSS.	81
28	K-means components 1 and 2 for each data, 3 clusters	81
29	Parallel Diagram with top 3 trials without Cross-validation (CV) for each metric.	85
30	Parallel Diagram with top 3 trials with CV for each metric.	88
31	High-level system architecture	94
32	Compound insertion page.	95
33	Model selection page.	95
34	Export predictions page.	95
35	Activity Diagram that describes the flow between the user and the system.	96

List of Tables

1	Description of the principle dose types.	6
2	Classification of xenobiotics.	7
3	Main sources of toxic data.	16
4	Requirements and properties of molecular descriptors.	18
5	Simplified Molecular Input Line Entry System (SMILES) Specification Rules.	23
6	Literature database taxonomies.	30
7	Chemoinformatics Databases with Types.	31
8	Confusion/Contingency Matrix.	38
9	Common classification metrics.	39
10	Several types of models used in toxicity prediction.	62
11	Initial merged dataframe.	70
12	Presence of several studies for the same compound.	71
13	Number of missing values for each column.	71
14	Set of options for parameters of light and medium presets.	74
15	Singletask binary classification results from training with medium preset for mutagenic label.	83
16	Singletask binary classification results from training with medium preset for carcinogenic label.	83
17	Singletask binary classification results from training with medium preset for genotoxic label.	84
18	Singletask binary classification results from training with light preset for mutagenic label.	86
19	Singletask binary classification results from training with light preset for carcinogenic label.	87
20	Singletask binary classification results from training with light preset for genotoxic label.	87

21	Multitask binary classification results with light preset and CV.	89
22	Overall top configurations	90
23	Key features of the developed app.	92

Acronyms and Abbreviations

ACID Atomicity, Consistency, Isolation, Durability.

AI Artificial Intelligence.

ANN Artificial Neural Network.

ANNs Artificial Neural Networks.

AOT Acute Oral Toxicity.

CAS Chemical Abstracts Service.

CB CatBoost.

CNN Convolutional Neural Network.

CNNs Convolutional Neural Networks.

CRISP-DM Cross Industry Standard Process for Data Mining.

CT Connection Table.

CV Cross-validation.

DILI Drug induced Liver Injury.

DL Deep Learning.

DM Data Mining.

DNN Deep Neural Network.

DNNs Deep Neural Networks.

ECFP Extended Connectivity Fingerprints.

EDA Exploratory Data Analysis.

EFSA European Food Safety Authority.

EMBL European Molecular Biology Laboratory.

EPA Environmental Protection Agency.

ESPH Element-Specific Persistent Homology.

ET Extremely Randomized Trees.

FCFP Functional Connectivity Fingerprints.

GATs Graph Attention Networks.

GCN Graph Convolutional Network.

GCNs Graph Convolutional Networks.

GNN Graph Neural Network.

GNNs Graph Neural Networks.

GRU Gated Recurrent Units.

HPG Hypothalamic-Pituitary-Gonadal.

IUPAC International Union of Pure and Applied Chemistry.

K-means K-Means Clustering.

KIM-1 Kidney Injury Molecule-1.

KNN K-Nearest Neighbors.

KSPA Kernel-based Structural and Pharmacological Analoging.

LGB Light Gradient Boosting Machine.

LR Logistic Regression.

LSTM Long Short-Term Memory.

LSTMs Long Short-Term Memories.

MACCS Molecular Access System.

ML Machine Learning.

MLP Multilayer Perceptron.

MOE Molecular Operating Environment.

NB Naive-Bayes.

NCBI National Center for Biotechnology Information.

NCI National Cancer Institute.

NGAL Neutrophil Gelatinase-Associated Lipocalin.

NIH the National Institutes of Health.

NTP National Toxicology Program.

P3TQ Product, Price, Place, Time, Quantity.

PCA Principal Component Analysis.

QSAR Quantitative Structure Activity Relationship.

RF Random Forest.

RL Reinforcement Learning.

RNN Recurrent Neural Network.

RNNs Recurrent Neural Networks.

SAR Structure Activity Relationship.

SEMMA Sample, Explore, Modify, Model, and Assess.

SL Supervised Learning.

SMILES Simplified Molecular Input Line Entry System.

SVM Support Vector Machine.

t-SNE T-Distributed Stochastic Neighbor Embedding.

TPE Tree-structured Parzen Estimator.

UFORDAT Environment Research in Progress.

UL Unsupervised Learning.

XAI Explainable Artificial Intelligence.

XGB Extreme Gradient Boosting.

Chapter 1

Introduction

1.1 Context and motivation

The assessment of the toxicity of chemical compounds plays a fundamental role in several areas, including the pharmaceutical industry, chemical safety and environmental protection [1, 2]. Toxicity refers to the ability of a substance to cause adverse effects on living organisms [3], and its determination is essential to ensure the safety and effectiveness of new chemical products.

Traditional toxicity assessment is a time-consuming and expensive process, involving a series of animal tests and laboratory trials [3]. Furthermore, growing concerns about animal welfare and the ethics involved in animal testing have highlighted the need to develop an alternative, and more effective approaches to predict the toxicity of compounds, even before their synthesis.

The increasing availability of public toxicological data, coupled with advances in ML techniques and computational modeling [4, 5], offers a unique opportunity to address this challenge. The ability to predict the toxicity of compounds based on their molecular structure and physicochemical properties can speed up the process of screening new compounds and reduce dependence on animal testing and monetary resources.

1.2 Objectives

This thesis aims to explore advanced ML, Deep Learning (DL) and computational modeling approaches to predict the toxicity of compounds accurately and efficiently to a selected set of endpoints, following a variety of strategies [6].

By overcoming the challenges of predicting compound toxicity, this research aims to provide a significant contribution to the scientific community and industries that rely on toxicity assessment. Despite traditional approaches that rely on *in vitro* or *in vivo* testing and those based on advanced computational

tools, there is still room for exploring different ML architectures and structural representations of compounds. The use of ML models represents an important step towards faster, more accurate and ethical toxicity assessment, resulting in benefits for both human health and the environment.

To accomplish this project, a specific methodology was followed, focusing on literature review, design of an adequate solution, its implementation, and validation. The following objectives were pursued:

1. Review of the relevant literature, tools, and existing datasets to gain a clear understanding of the state of the art in the respective field;
2. Collection and integration of the identified relevant toxicity datasets for different end-points;
3. Exploration and optimization of different ML and DL models, as well as various computational representations of compounds, for the development of robust models capable of predicting distinct toxicity end-points;
4. Development of a software application, integrating the most effective developed models;
5. Writing of the dissertation and related publications.

1.3 Thesis Structure

This thesis is structured in two parts. The first part includes all the introductory and fundamental information, a literature review, and a general state-of-the-art overview. The second part focuses on the case study that was conducted, specifically including information related to both the ML tasks and the software development processes carried out. **Chapter 2** explores key factors influencing toxicity, including intrinsic properties, dose, and exposure conditions; discusses host dose-response relationships; examines methods for toxicity testing (both in vivo and in vitro); and details the quantification of toxic endpoints, alongside descriptions of major organ-specific toxicities such as carcinogenesis, hepatotoxicity, and renal toxicity. **Chapter 3** begins by defining important terms in chemoinformatics, proceeds with the description of molecular descriptors and representations, and concludes with the presentation of some relevant software tools in the field. **Chapter 4** presents the different types of ML and some basic principles. Traditional ML algorithms and relevant DL architectures are also introduced. The chapter concludes with relevant software tools in this domain. **Chapter 5** presents relevant ML approaches for toxicity. **Chapter 6** includes all the relevant information concerning the case study, specifically detailed information on each phase of the work, from data collection to the development of predictive models. **Chapter 7** presents

and discusses the results, synthesizing the main findings obtained, as well as providing a critical analysis of them. **Chapter 8** focuses on the app, particularly on the development environment, features, and architecture. **Chapter 9** presents the main conclusions of the work and suggestions for future research. Finally, the thesis ends with the presentation of the bibliographic references used to frame and support the entire investigation conducted.

Chapter 2

Toxicity in Toxicological Sciences

The definition of toxicity inevitably derives from the definition associated with toxicology. Therefore, to define the former, it is relevant to understand the latter. Toxicology is defined as the study of the adverse effects of chemicals or physical agents on living organisms, focusing, in a more modern context, on the toxic effects, whether chemical or biological, of a particular substance. This “Science of Poisons” has always sparked interest in fields such as chemistry, physiology, biochemistry, and molecular biology and can traditionally be divided into various branches, including clinical, forensic, occupational, environmental, regulatory, food, and ecosystem toxicology [7]. Consequently, in conjunction with the detrimental impact of a specific compound, the concept of “toxicity”, as precisely delineated by [1], signifies the extent or the potential of a given substance to elicit injurious or adverse effects within living organisms.

Some authors even define toxicity (T) mathematically, expressing it as a function defined in terms of exposure (E), which can be, in turn, defined by the dose (c) and the duration of exposure (t): $T = f(E(c, t))$ [8].

Toxicity can also be seen as an interaction that can manifest at the cellular level and can be perceived as a cause-and-effect relationship between the compound and the organism, and vice versa: what the organism does to the administered compound or what the compound induces in the organism. Both interactions can be defined as functions of dose and time, introducing new biological variables: the cause-and-effect relationship between organism and compound is associated with the processes of compound absorption and elimination, and the cause-and-effect relationship between compound and organism is linked to damage and recovery processes, proportionally. This interaction continues to evolve, and further breakdowns of each process can be performed, for example, at the level of the organism’s internal processes (excretion, distribution, bio-transformation, reversibility, repair, and adaptation), which depend on the molecular events of each [8].

2.1 Determining factors

The multitude of toxic effects discussed by [2] underscores the complexity of toxicity. However, the assessment of toxicity can be effectively conducted by considering determining factors, which play a pivotal role in evaluating the overall toxic impact of a substance. These factors aid in the interpretation of toxicity data, as they represent the variables that are systematically manipulated or taken into account during the assessment of toxicity. The primary determining factors are elaborated upon in the subsequent section.

2.1.1 Intrinsic

Intrinsic toxicity refers to the inherent and specific toxic potential of a chemical substance, which stems from its distinct combination of chemical and physical properties. These properties encompass the molecular structure, functional groups, solubility, volatility, stability, reactivity, and various other physical attributes [9]. This concept is chiefly characterized by its ability to manifest diverse modes of toxic action at the biological level, including narcosis, baseline toxicity, enzyme inhibition, receptor competition, and various reactive mechanisms. Furthermore, intrinsic toxicity is intricately linked to the concentration of the substance within an organism that precipitates these particular internal effects, with variations contingent upon the intrinsic properties of the substance.

2.1.2 Dose

Dose is a critical factor in determining toxicity, serving as a key parameter in toxicological considerations. Specifically, dose refers to the quantity of a substance administered at one time. In toxicology discussions, the focus is often on the total dose, which encompasses the cumulative effect of individually administered and experimentally measured doses, whether *in vitro* or *in vivo* [8].

Different types of doses exist, categorized based on **exposure, absorption, or administration.**

Table 1: Description of the principle dose types.

Dose Type	Description
Absorbed	Represents the quantity of a substance that entered the body through various routes (skin, eyes, lungs, or digestive tract) and was taken up by organs or specific tissues. It is also referred to as internal dose.
Administered	Pertains to the quantity of a substance administered, typically orally or through injection. It is essential to note that an administered dose taken orally may not necessarily be fully absorbed.
Total	The cumulative sum of all individual doses, providing a comprehensive view of the overall exposure.

Despite all these types, there is a base mathematical equation for the dose [2.1]. The equation encapsulates the case of multiple doses and one-time doses (single acute dose), a limiting case where the dose rate equals the dose. In the equation, N represents the total number of doses or time points, Dose Rates variable represents the rate at which the substance or agent is administered or absorbed at each time point, and n is the index variable, indicating each dose or time point [8].

$$\text{Dose} = \left(\sum_{n=1}^N \text{Dose Rates} \right) \quad (2.1)$$

The same authors refer to the fact that to adequately characterize the dose, consideration must be given to additional parameters, such as the frequency of administration and the duration of exposure. This broader perspective is crucial for a comprehensive understanding of the toxicological implications, as will be seen further.

2.1.3 Exposure conditions

According to [8], it is recognized that in the absence of exposure to hazardous compounds, toxicity does not occur. Thus, exposure conditions (or external factors) profoundly influence the interaction between a substance and living organisms, thereby shaping the actual risk and impact [10]. These factors include routes of exposure, the frequency and duration of the exposure, the number of substances involved in the exposure, and the environmental circumstances.

First, the **routes of exposure** delineate the pathways through which a substance infiltrates an organism, including ingestion, inhalation, dermal contact, or other means. Different routes can lead to varying degrees of exposure and potential toxicity.

Another crucial factor is the **frequency and duration of exposure**, which outlines how often and for how long individuals or populations are exposed to a substance. Exposure time may be continuous,

intermittent, or random, while exposure itself can be acute, short-term, sub-chronic, or chronic. The effects of continuous or frequent exposure may differ from those of intermittent or short-term exposure.

Additionally, the concept of **mixed exposures** recognizes that organisms are frequently exposed to multiple substances simultaneously, leading to potential interactions and synergistic or antagonistic effects. The presence of mixed exposures may modify the overall toxicity of individual substances.

Finally, consideration of **environmental circumstances** is essential, acknowledging the specific conditions and contexts in which exposure transpires, such as the presence of other chemicals, temperature, humidity, and ecological factors. These environmental circumstances play a pivotal role in modulating the actual impact of intrinsic toxicity in real-world settings [10].

2.1.4 Host dose-response

Xenobiotics

Xenobiotics are chemicals that do not naturally occur in the environment. They are either synthetically produced or found in increased concentrations due to anthropological activities. In contrast, endobiotics are naturally produced within organisms, but can become xenobiotics when present in excessive concentrations induced by humans [11].

These compounds serve various purposes in domestic, agricultural, and industrial settings and are classified based on their nature, uses, physical state, or physiological effects. They are subdivided into 17 groups: pesticides, pharmaceutical compounds, personal care products, illicit drugs, industrial products, and nuclear waste, as shown in Table 2.

Table 2: Classification of xenobiotics (some examples, modified version [11]).

Xenobiotics					
Industrial products					
Pharmaceutical	Illicit drugs	Personal care products	Pesticides	Nuclear waste	Others
• Essential medicines; • Antibiotics; • Psychiatric drugs.	• Hallucinogenics; • Stimulators; • Inhibitors.	• Fire retardants; • Fragrances; • Antiseptics; • Plasticizers.	• Insecticides; • Bactericides; • Fungicides; • Herbicides.	• Radioactive isotopes; • Radioactive contaminants.	• (In)organic waste; • Heavy metals; • Electronic waste.

Xenobiotics, as non-nutrient chemicals, are foreign to species during metabolism [3]. They present challenges in dose studies, as prolonged exposure, even to low concentrations, may lead to toxic, mutagenic, or teratogenic effects (birth defects). The accumulation of xenobiotics in living organisms is

attributed to the specific properties of these substances. Their small size, high water solubility, volatility, and polarity present challenges in terms of quantification, identification, and removal. This characteristic physicochemical structure contributes to their significant ability to accumulate within biological systems [11].

As these foreign substances undergo metabolic processes, they can yield metabolites categorized into detoxified (less toxic) or bioactivated (more toxic) forms. Interestingly, while the original xenobiotic might not directly induce harmful effects, its metabolites can harbor carcinogenic potential— a phenomenon termed bioactivation. Understanding these metabolic pathways emphasizes the significance of comprehending dose-response relationships, as varying doses and exposures to xenobiotics prompt diverse responses in organisms—there are toxic and non-toxic doses for any substance.

Dose-response Relationship

The dose-response relationship correlates exposure characteristics with effects on body functions or health [1]. It reflects the causality principle, where observed effects result from administered xenobiotics. This relationship is based on data from *in vivo* or *in vitro* experimental testing [12].

Key concepts include **inter-individual variability**, which highlights differences in individual responses to substances, emphasizing the need for standard references in evaluations. Another concept is **proportional biological response**, indicating that the intensity of a biological response correlates directly with the substance's concentration in the body fluids [12].

Additionally, the **threshold effect** refers to the minimum dose at which an effect begins to be observed, marking the point where detoxification or repair capabilities are exceeded. As the dose increases, it eventually reaches a lethal threshold [1].

From a practical perspective, there are two types of dose-response relationships: (1) the individual or graded, and (2) the quantal or all-or-none. Both of the approaches are described further.

(1) The individual or graded dose-response relationships

Individual dose–response relationships characterize the response of an individual organism to varying doses of a chemical in a continuous-scale. Typically, this response is described as a continuous measured effect over a range of doses [13]. This type of relationship verifies the proportional biological response concept, as the relationship is characterized by a dose-related increase in the severity of the response. Understanding the response involves not just assessing the individual nature but also comprehending the nature of the substance itself. Substances exhibit multiple mechanisms of toxicity, each with its own distinct “dose-response” relationship, leading to diverse adverse effects and specific endpoints [13].

The behaviour of this relationship can be portrayed with a graph on a logarithmic scale, where the data ends up fitting a straight line, forming a s-shaped curve [Figure 1], exhibiting nearly linear behavior within a mid-concentration range, ultimately becoming asymptotic towards both zero and maximum response levels.

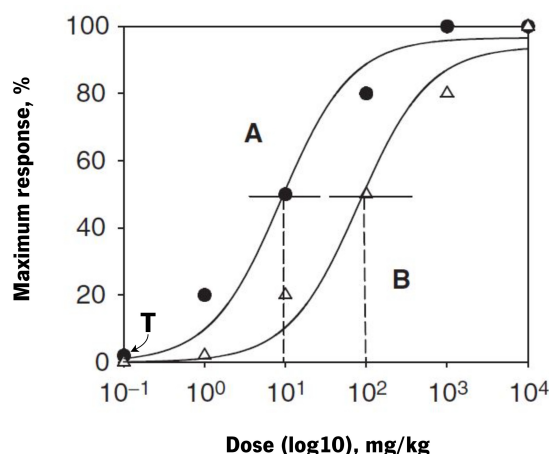


Figure 1: Individual dose-response curves of two chemicals (A and B). A - curve of chemical A. B - curve of chemical B. T - threshold dose. (adapted from [1]).

(2) The quantal or all-or-none dose-response relationships

In studies of groups of test organisms, as the dose is increased, there is not a specific dose at which all the organisms in the group will suddenly develop the same response, especially because of the inter-individual variability concept. Instead, there will be a range of doses over which the organisms will respond in the same way to the test substance. In this context, the quantal relationship arises, filling the need to specify this group behavior and relating the dose with the toxic effects observed in a population of individuals [1].

This dose-response relationship is generally represented against the log of the dose (exactly as the previous one), but has the particularity of being able to be represented in two different ways, depending on the specific need: (a) Normal Frequency Distribution (Gaussian) or (b) Quantal Dose-Response, as shown in figure 2.

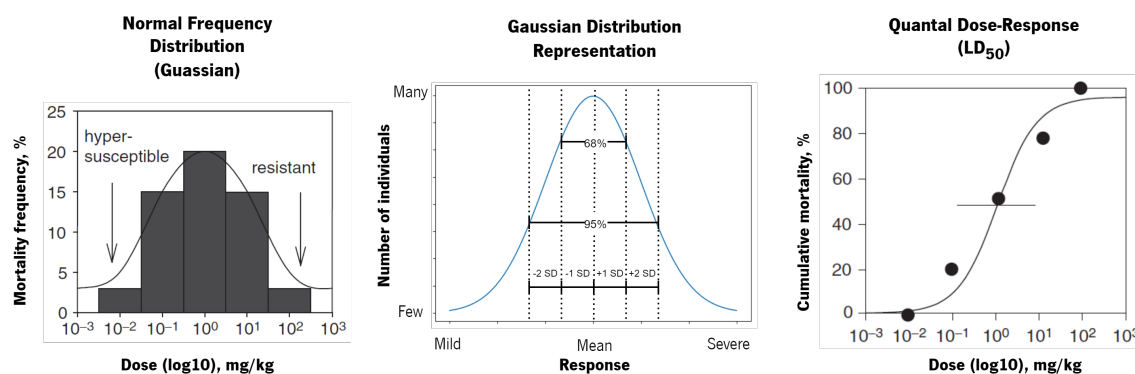


Figure 2: Comparison of shapes of the dose-response curves between Normal Frequency Distribution and Quantal Dose-Response. (adapted from [1]).

(a) Normal Frequency Distribution (Gaussian)

This population-level evaluation acknowledges that there is a range of doses over which the organisms respond in a similar manner, rather than exhibiting a continuum of response levels. This perspective allows for a broader understanding of how populations react to specific substances and forms the basis for evaluating the safety of these substances at larger scales, overcoming the concept of inter-individual variability. As visible in the first plot of figure 2, this distribution presents a frequency response plot with the percentage of responding individuals in each group as a function of dose.

(b) Quantal Dose-Response

Despite being more used for lethal data, the cumulative-response curve also offers a method to depict nonlethal frequency-response data, plotting the logarithm of the dose against the percentage of individuals exhibiting at least a specified response. Key metrics derived from this curve include the “no observed adverse effect level” (NOAEL), denoting the highest dose where no discernible toxicity manifests, and the “lowest observed adverse effect level” (LOAEL), representing the lowest dose instigating toxicity. Additionally, the TD_{50} signifies the “dose statistically associated with toxicity” in 50% of the test organisms. In instances where the outcome of interest is lethality, LD_{50} stands as the pertinent notation for the “lethal dose” for 50% of the organisms. Finally, the “effective dose” (ED) is commonly used in toxicology to denote the dose or concentration of a substance that produces a specific effect or response in a certain percentage of a population or test subjects. Generally, the 50% response level is used (ED_{50}), although any response level, such as ED_{01} , ED_{10} , or ED_{30} , could also be chosen [1, 13].

In the third plot of figure 2, the vertical line indicates the dose estimated to cause the death of 50% of the population. This type of representation is commonly applied to individual dose-response relationships

(as presented previously in figure 1), allowing to understand in detail how individual variations contribute to the cumulative response scenario.

Potency & Hormesis

The concept of potency, although not yet formalized, has already appeared in section 2.1.4. It refers to the amount or concentration of a substance required to produce a specific effect, often referring to the strength or efficacy of a drug or toxicant. A more potent substance requires a lower dose or concentration to produce a given effect compared to a less potent substance [13].

According to [14], hormesis is a phenomenon where exposure to low doses of a substance or stressor elicits a beneficial or stimulatory effect, whereas higher doses produce adverse or toxic effects. In hormesis, the dose-response curve typically displays a U-shaped or J-shaped curve, where low doses may result in a positive response (enhanced growth, improved health), moderate doses may have a neutral effect, and high doses result in toxicity or harm [15].

2.2 Toxicity testing

2.2.1 In vivo testing

In vivo testing involves the use of animal models (rodents and non-rodents) to assess acute or sub-chronic toxicity. Acute toxicity testing typically determines the median lethal dose (LD₅₀), while sub-chronic testing involves multiple dosage administrations over specific intervals, which can extend up to 1 or 2 years for certain types of toxicity, such as carcinogenicity.

The evaluation in *in vivo* testing involves various endpoints. It includes assessing effects on food intake (where the chemical action differs based on oral, dermal, inhaled, or injected administration), monitoring changes in body weight, analyzing blood chemistry (providing insights into the chemical's toxicokinetics by understanding processes like absorption, distribution, metabolism, and excretion), and employing *postmortem* tissues for histopathological analysis.

2.2.2 In vitro testing

In vitro testing involves experiments conducted outside a living organism, typically in a controlled laboratory setting, using cell cultures or isolated tissues, as opposed to *in vivo* testing with animal models. This method is favored for its lower cost, shorter duration, and versatility in evaluating a wide range of chemicals.

These tests facilitate controlled examinations of chemical interactions with biological systems at cellular or molecular levels, assessing endpoints such as cytotoxicity, genotoxicity, and affected molecular pathways [16].

In vitro models range from simple setups with individual enzymes to complex systems using immortalized or primary cells. For example, simple tests like the P450-CYP assay evaluate cytochrome enzyme interactions, while more complex models, such as the Ames test, assess carcinogenic potential [16].

An initiative like Tox21, led by the National Institutes of Health (NIH), utilizes quantitative high-throughput screening (qHTS) to evaluate nearly 10,000 chemicals and drugs, identifying those that alter biological pathways, potentially leading to toxicity [17].

Overall, *in vitro* testing provides crucial insights into chemical effects, aiding toxicity assessment and understanding the mechanisms and risks associated with chemical exposures.

2.3 Toxic endpoints

Exposure to chemicals can trigger chronic or acute effects, adversely affecting organs and disrupting normal physiological functions [13]. It is crucial to ascertain a chemical's potential for harmful effects, including understanding its mode of action. Assessing chemical toxicity typically involves testing through *in vivo* or *in vitro* models.

The sensitivity of organs to chemical exposure varies based on their unique structures and functions. Consequently, different organs exhibit differing levels of susceptibility to chemical impacts. Toxic endpoints, specifically associated with environmental chemicals, denote distinct adverse effects caused by specific toxicants.

These endpoints serve as indicators, pinpointing the precise adverse effects triggered by particular toxicants. By delineating these endpoints, relevant insight into the risks posed by environmental chemicals is obtained, enabling targeted strategies for assessment and mitigation.

2.3.1 Quantifying toxic endpoints

Specific types of adverse effects of a specific compound can be classified either qualitatively or quantitatively. Qualitative classification focuses on describing the toxic effects of a substance in general terms, while quantitative classification is concerned with the measurement and quantification of toxic effects in relation to the dose.

The **qualitative endpoints** include binary classification (toxic or non-toxic) and ordinal classification

(low, moderate, or high toxicity).

On the other hand, **quantitative endpoints** are those related to a specific concentration from which damage occurs. This quantitative classification is centered on the precise measurement and quantification of toxic effects concerning the dosage of the substance. It furnishes numerical data that elucidates the correlation between the substance's dosage and the corresponding biological response. Exemplary measures within this category have already appeared in quantal dose-response relationships (e.g., LD₅₀, ED₅₀, etc.) [6].

2.3.2 Describing major organ-specific toxicities

Carcinogenesis

Carcinogenesis is the transformation of normal cells into neoplastic cells, leading to tumor formation [16]. As noted in Xenobiotics (subsection 2.1.4), carcinogens can interact with DNA, RNA, and proteins, causing alterations in cellular functions, such as proliferation. They are classified as genotoxic or nongenotoxic based on their mechanisms. Genotoxic carcinogens directly damage genetic material, are typically mutagenic, and are easier to assess, following stages of initiation, promotion, and progression [18, 16]. In contrast, nongenotoxic carcinogens induce carcinogenesis through non-genetic means, complicating assessment due to variations in dose and species-specific responses, where a carcinogen may affect one species but not another [16, 19].

Tests for genotoxicity in *in vivo* models include point mutations and chromosome damage analysis, with many known carcinogens identified by these methods, as reported by the National Toxicology Program (NTP).

Hepatotoxicity

Hepatotoxicity results from chemicals that adversely affect the liver, a key target for environmental toxins due to its structural role in detoxifying blood. This process includes the removal of xenobiotics through enterohepatic circulation, where bile acids are reabsorbed into the bloodstream and transported back to the liver for potential detoxification. Some xenobiotics may undergo cycles of absorption and metabolism, while others are directly excreted [20]. Liver toxicity manifests as lipid accumulation (steatosis), disrupting normal lipid uptake, and can lead to cholestasis, jaundice, fibrosis, or even tissue necrosis [16].

Assessing hepatotoxicity involves blood tests that measure proteins released upon liver damage, providing insights into liver function and bile flow. Genotoxicity studies focus on gene expression patterns

linked to hepatotoxicity in isolated human hepatocytes, revealing mechanisms of injury and early adverse effects on the liver [20].

Renal Toxicity

Renal toxicity arises from xenobiotic exposure that damages the kidneys, particularly vulnerable due to their structure and functions [16]. The nephron concentrates toxicants during urine formation, leading to their accumulation. While filtering blood, the kidney reabsorbs essential elements, leaving toxic components concentrated in the residual filtrate.

This toxicity manifests as kidney impairment reflected as altered urine composition and elevated levels of urea nitrogen, serum creatinine, and biomarkers like Kidney Injury Molecule-1 (KIM-1), indicating glomerular damage [21]. Advanced toxicity may also obstruct urine outflow. Gene expression profiling aims to identify biomarkers for kidney toxicity, with non-invasive methods developed to detect early damage, including KIM-1 and Neutrophil Gelatinase-Associated Lipocalin (NGAL) [16].

Neurotoxicity

Neurotoxicity encompasses acute and chronic effects caused by neurotoxicants in the environment, reflecting an agent's ability to impair the structural or functional integrity of the nervous system. The vulnerability of the nervous system to these agents leads to structural effects, including demyelination, axonopathy, and neuronopathy [22, 16].

Demyelination disrupts nerve signal transmission by damaging the myelin sheath, which protects nerve fibers. Various chemicals can specifically affect different nerves; for instance, some may harm the optic nerve while others impact cranial nerves. Long-term exposure to neurotoxic substances, such as cyanide, carbon monoxide, or lead, can damage the myelin-producing cells (Schwann cells and oligodendrocytes).

Axonopathy leads to the breakdown of nerve fiber parts, particularly at the ends, resulting in dying back neuropathy. This condition can arise when toxic substances disrupt the energy supply needed by nerves or target specific nerve structures [16].

Neuronopathy involves damage to nerve cells, particularly in the central nervous system. Certain pesticides can interfere with the function of acetylcholine, a key neurotransmitter, thereby altering signal transmission between neurons [23].

Reproductive Toxicity

Reproductive toxicity occurs when a chemical disrupts normal reproductive tissue function or fertility. The reproductive system, governed by the Hypothalamic-Pituitary-Gonadal (HPG) axis and hormone release, can be adversely affected by chemicals, impacting sperm and egg development as well as fertilization. Testicular and ovarian tissues are particularly sensitive during development, leading to changes that affect postnatal development, behavior, and reproductive success when toxic exposure occurs [16].

Assessing this toxicity often involves *in vivo* studies, directly examining sperm or egg development, mating and fertilization outcomes, normal tissue development, embryo growth, and birth success. *In vitro* assays assess if hormone-responsive tissues are targeted by these toxicants, while *ex vivo* rodent tissue studies determine if a chemical inhibits hormone synthesis [13, 16].

Mutagenic Toxicity

Mutagenic toxicity refers specifically to the potential of certain substances to cause mutations in the DNA of living organisms, leading to cancer and other genetic disorders. Mutagens can be classified as chemical agents, physical agents (like radiation), and biological factors (such as certain viruses).

The mechanisms by which mutagens exert their effects often involve direct interaction with DNA, resulting in alterations to its structure or sequence.

The assessment of mutagenic toxicity is typically conducted through a combination of *in vitro* and *in vivo* tests (e.g. Ames test and mammalian cell assays that observe chromosomal alterations). These methods help identify whether a substance can induce genetic mutations, providing critical information for regulatory evaluation and safety standards [24].

2.4 Databases

Toxicity databases (Table 3) are crucial tools in toxicological research, consolidating and organizing data on chemical compounds and their adverse effects, which provides detailed information on many toxicity endpoints. Integrating data from various sources, including *in vitro* and *in vivo* assays and computational toxicity predictions, these resources offer a comprehensive view of chemical risk profiles.

Table 3: Information about main toxic databases/datasets. N/A means "not applicable". * means "not applicable". * means information is incomplete due to request dependency. DD marks fields that vary depending on the database/dataset.

Source Name	Brief Description	Endpoints	Features	Shape	License
Tox21[17]	Dataset used for the Tox21 Data Challenge 2014. Contains results from <i>in vitro</i> assays for assessing the biological activity of more than 10k compounds across different toxicity endpoints.	Nuclear Receptor Signaling Panel and Stress Response Panel (12 binary endpoints)	The complete .sdf file ("Tox21 10K") has many features for each molecule (e.g. Molecule Identifiers, Coordinate and Bond Sections, Chemical Formula, Formula Weight)	10,000 compounds and 12 endpoints.	Public
ToxCast[25]	Another initiative by the U.S. Environmental Protection Agency (EPA) that aims to predict the toxicity of chemicals using high-throughput screening assays. It provides data on the biological activity across hundreds of <i>in vitro</i> assays. ToxCast is aggregated from more than 20 assay sources, including Tox21 program. The dataset can be easily seen using the dashboard of EPA.	1485 assays were done.	Some features are structural diagram of the chemical, CAS information, PubChem ID and Chemical Properties.	1,800 compounds and 700 endpoints.	Public.
CompTox[26]	The CompTox Dashboard integrates data from ToxCast and Tox21 programs.	1485 assays (endpoints)	Information of the chemical structures, properties and toxicity data	1485 assays (endpoints) and 498 chemicals	Public.
SIDER[27]	1,427 marketed drugs and adverse drug reactions (ADR), grouped into 27 system organ classes.	27 main ADR (e.g. Hepatobiliary disorders, Metabolism and nutrition disorders, Product issues, Eye disorders, etc.)	Only SMILES strings	1,427 compounds and 27 ADR	Contact for commercial use.
ToxBank[28]	A dedicated web-based warehouse for toxicity data management and modelling, a "gold standards" compound database and repository of selected test compounds, and a reference resource for cells, cell lines and tissues of relevance for <i>in vitro</i> systemic toxicity research carried out across the Alternative Testing Strategies SEURAT program.	<i>In vitro</i> systemic toxicity endpoints*	Dose-response and omics results	N/A	Request needed.
ToxNet[29]	Toxicological information services provided by the National Library of Medicine (NLM). A collection of databases that provided a wide range of toxicological information. As part of a broader NLM reorganization, ToxNet was retired in December 2019, and most of its content have been integrated into other NLM products and services.	DD	DD	DD	Public
VEGA-AHUB[30]	A repository of freely available, downloadable tools based on computational toxicology methodologies. The main software tool available in VEGA-AHUB is VEGA QSAR software encoding more than 90 QSAR models for tens of endpoints for human toxicology, ecotoxicology, environmental, physicochemical and toxicokinetic properties.	tens of endpoints for human toxicology, ecotoxicology, environmental, physicochemical and toxicokinetic properties	SMILES strings, ID numbers and other chemical properties.	DD	Mentioning of VEGA-AHUB platform and relative reference articles needed.
ClinTox	Qualitative data of drugs approved by the FDA and those that have failed clinical trials for toxicity reasons.	Not specified. Seems to be a mixing of several endpoints that resulted in final classification.	SMILES strings only.	[1485,3]	Public
QSAR oral toxicity	Data set containing values for 1024 binary attributes (molecular fingerprints) used to classify 8992 chemicals into 2 classes (very toxic/positive, not very toxic/negative).	Oral toxicity, not properly specified.	Molecular fingerprints.	[8992,1024]	(CC BY 4.0)
DILIrank Dataset[31]	Drug Induced Liver Injury Rank dataset includes 1,036 FDA-approved drugs divided into four classes according to their potential for causing drug-induced liver injury.	Occurrence or development of drug-induced liver injury (DILI).	Compound information (e.g. name, severity class, label section, concern, etc.)	[1036,6].	Public, suggests contacting in case of download
OpenFood-Tox [32]	The European Food Safety Authority (EFSA) chemical hazards structured database	Quantitative endpoints (e.g. NOAEL, NOEL, etc.) for variety of toxicity types (e.g. systemic, pulmonary, etc.) and effect observed. Contains toxic tests detailed information. Chemicals share only 3 generic binary endpoints (carcinogenic, mutagenic and genotoxic).	DD, but include physico-chemical properties and pharmacokinetic/toxicokinetic data.	DD	Creative Commons Attribution 4.0 International

Chapter 3

Chemoinformatics

The definitive inception of chemoinformatics lacks a specific founding moment due to its gradual evolution driven by the intricacies of computational methods. However, in the 1960s, the abundance of accumulated chemical data necessitated effective storage and management, compelling a shift towards electronic formats. This transformation facilitated enhanced exploration, characterization of chemical properties, and rigorous analysis of this data, marking the nascent stage of the term [33].

Initially seen as a fresh field primarily focused on handling “chemical information”, chemoinformatics also arose as a response to the realm of computational chemistry. However, these domains diverge significantly. Computational chemistry, primarily dedicated to theoretical quantum mechanical calculations, does not align with the central objective of processing chemical information, which is the essence of chemoinformatics [33].

The term “chemoinformatics” gradually crystallized over time, leading to the rapid emergence of various definitions, being possible to construct a global definition for the term that includes all the nuances. Chemoinformatics is an interdisciplinary field at the intersection of chemistry, computer science, and information technology [34, 35, 36, 37], which encompasses the systematic organization, management, and analysis of chemical data through diverse computational techniques. This process involves the creation, curation, and exploration of extensive chemical databases, facilitating efficient searching and retrieval of molecular information [36]. Additionally, chemoinformatics utilizes computational models, molecular descriptors, and representations to predict and analyze diverse properties and behaviors of chemical compounds based on their molecular structures [36].

Furthermore, it also involves encoding molecules into suitable computational formats, employing molecular descriptors and representations, to enable comprehensive computational analysis and manipulation of chemical entities. In essence, chemoinformatics serves as a pivotal platform for handling, interpreting, and extracting valuable insights from chemical data [34], crucial for drug discovery, material science, and various other chemical domains [38].

3.1 Molecular Descriptors

Molecular descriptors are computational depictions of a molecule obtained through the application of a well-defined algorithm to a known molecular portrayal (unequivocal molecular representation) or a well-defined experimental procedure, which includes diversity analysis, library design, environmental/toxicological hazard estimation and virtual screening [37, 39]. This chemical representation, commonly named “theoretical molecular descriptor”, is a numerical value that captures the properties of a molecule, reflecting physicochemical, structural, electronic, topological, biological, quantum mechanical or even substructure patterns properties.

Thousands of molecular descriptors have been proposed over the years, ranging from simple bulk properties to complex 3D interpretations and molecular fingerprints of thousands of bits [37, 39]. Given the multitude of available molecular descriptors, it is crucial to establish a framework outlining essential requirements that these descriptors must fulfill. Additionally, it is desirable for them to exhibit certain properties, though these are not mandatory prerequisites (Table 4).

Table 4: Requirements and properties of molecular descriptors [39]).

Requirement (must be/have)	Criteria purpose
1. Be invariant to atom labeling and numbering	• Well-Defined Nature • Invariance to Molecule Manipulation • Applicability Domain
2. Be invariant to the molecule roto-translation	
3. Be defined by an unambiguous algorithm	
4. Have a well-defined applicability on molecular structures	
Preferred Attributes (should be/have)	Criteria purpose
1. Should have structural interpretation	• Interpretability and Relationship to Experimental Properties • Limited Association with Existing Descriptors • Continuity and Low Degeneracy • Dependency on Molecular Representation
2. Should have a good correlation with at least one experimental property	
3. Should not have trivial relation with other molecular descriptors	
4. Should not be based on experimental properties	
5. Should preferably be continuous	
6. Should preferably show minimal degeneracy	
7. Should preferably be simple	
8. Should preferably be applicable to a broad class of molecules	
9. Should preferably be able to discriminate among isomers	
10. Should preferably have calculated values in a suitable numerical range for the set of molecules where it is applicable to	

Basically, **requirements** define if a molecular descriptor is well defined: they should be invariant to all the operations related to molecule manipulation, which do not affect the molecular structure. They should have an applicability domain, being clear to which class of molecules they can be applied (salts, ionic liquids, proteins, polymers, etc.).

Additionally, **preferred attributes** delineate the utility of a descriptor. They necessitate interpretability, suggesting a connection to experimental properties without excessive reliance on existing descriptors. The descriptor should ideally be a continuous variable, demonstrating low degeneracy, signifying its capability to capture variations in molecular structure, even when they are minimal.

This representation allows to encode a wide range of molecular information, which has a direct impact on the outcome, performance, and applicability of computational models, highlighting the importance of this study [39].

Despite the myriad of molecular descriptors presented in the literature, the emphasis will be on physicochemical and structural properties.

3.1.1 Physicochemical descriptors

Physicochemical descriptors encompass a broad spectrum of characteristics pertaining to both the physical and chemical attributes of molecules. These descriptors typically encompass properties that can be determined experimentally, while also being computationally derived.

There are three main types of physicochemical properties [40]:

- **Physical properties:**

- **Freezing, Boiling, and Melting Points:** specific temperatures at which a substance transitions between different states of matter (solid, liquid, gas);
- **Flammability:** capacity to ignite and burn when exposed to a heat source;
- **Corrosivity:** ability to gradually degrade or destruct materials;
- **Oxidizing ability:** ability to give off oxidizing substances or oxidize combustible materials, increasing fire or explosion hazards;
- **Vapor pressure:** measure of the tendency of a substance to evaporate or transition from its liquid or solid state to the gaseous phase.

- **Solvation properties:** describes the behavior of a chemical, concerning its interactions with different phases and partitioning between them;

- **Phase partitioning:** explores the concept of partition or distribution coefficients, specifically examining measurements across immiscible phases, notably between water and a hydrophobic solvent like 1-octanol (represented by the octanol/water partition coefficient ($\log P$, measured using $[K_{ow}]$);
- **Solubility:** focuses on the ability of a chemical to dissolve in a solvent, particularly emphasizing measurements in water, crucial in chemical alternatives assessment;
- **Molecular Attributes:** used to describe properties related to molecular shape and size (can be related to bioavailability and reactivity). Some examples are:
 - **Frontier orbital energies:** refer to the highest occupied molecular orbital (HOMO) and lowest unoccupied molecular orbital (LUMO) energies, influencing a molecule's reactivity;
 - **Polarizability:** ability to form instantaneous dipoles when interacting with an electric field, influencing its response to external forces.

3.1.2 Structural descriptors

Structural descriptors describe topological features readily determinable from the structure of the molecule alone. They are usually divided by their respective dimensionality, and include [41, 42]:

- **0D: Constitutional** - Derived from chemical formulas, capturing atom types and occurrences without information about atom connections (e.g. number of atoms, number of bonds, molecular weight);
- **1D: Structural keys** - Obtained from substructure lists, representing partial structural fragments, not the entire topology, useful for similarity/diversity analysis (e.g. fragment counts, counts of atom types);
- **2D: Topological** - Derived from 2D representations including atomic composition and connectivity in a molecular graph (number of molecular walks, connectivity indices χ , kappa shape indices);
- **3D: Geometrical** - Incorporate spatial information with three-dimensional coordinates of atom positions, considering the molecule as a rigid geometrical structure (polar surface area, radial distribution function, molecular eccentricity);

Regardless of the molecular descriptor definition, the information gatherable by a molecular structure depends on the selected molecular representation.

3.2 Molecular Representations

Machine-readable chemical representations are fundamental for designing comprehensive chemical databases used in the analysis and manipulation of chemical information. These representations encompass various formats, including linear, 2D, and 3D structures [36].

Chemical structure information finds storage in diverse chemical file formats. Despite the abundance of formats available [39], each focuses on specific atomic or molecular details, incapable of encapsulating all potential chemical data.

Within this context, two primary types of chemical files emerge: those dedicated to topological representations (2D) and others to geometrical representations (3D) of molecules. Topological representations encompass diverse linear notations, prominently the widely utilized SMILES file format. Geometrical representations, on the other hand, store 3D atomic coordinates and are embodied in formats like MDL and SYBYL. Additionally, efforts have been made to establish a standardized format for chemical data storage based on an XML dialect, known as the Chemical Markup Language (CML) [39].

The representation of molecules can take various forms, each capturing distinct information about the molecule. [43] emphasizes specific guidelines for drawing organic molecules, including the importance of realism, economy, and clarity in their structural depiction. This guideline comprises three key suggestions: firstly, drawing chains of atoms as zig-zags; secondly, omitting the Hs and C–H bonds attached to carbon atoms; and thirdly, excluding capital Cs to represent carbon atoms.

Subsequently, three primary methods exist to communicate the **chemical structure** of organic molecules: Lewis, Bond-line, or Condensed structures (or Line Notations), with the latter being elaborated upon in the following section.

3.2.1 Line Notations

The structure (Lewis structural formula) is communicated without explicitly drawing the chemical bonds. Instead, all atoms are listed in a linear sequence based on its elemental symbols. The connections between atoms are implied based on the arrangement in the sequence (the order of the atoms suggests the connectivity). This representation is very compact and does not depict any bonds explicitly, relying on the reader's understanding of how atoms are bonded in a typical organic molecule (condensed/line formulas can be read from either direction).

Using the following example of propane (Figure 3), it is possible to observe that it can be depicted in various structural forms, illustrating its versatility and significance in organic chemistry, and allowing

to compare how each representation highlights different aspects of its molecular structure, such as bond connectivity, spatial arrangement, and the relative positioning of atoms within the molecule.

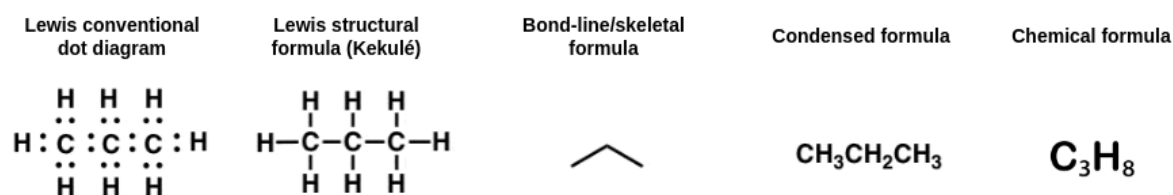


Figure 3: Four ways of depicting propane.

SMILES

The evolution of chemical formalization, specifically in the realm of nomenclature, is crucial for advancing the comprehension and communication of chemical compounds. The historical trajectory underscores the transition from rudimentary chemistry to the computer era, where computational capabilities facilitated the processing and storage of chemical information. [44] introduces the methodology and encoding rules integral to the SMILES system, marking a significant advancement in the pursuit of efficient chemical information retrieval systems.

SMILES is a chemical notation that encodes molecular structure as a single line of text (formed by a set of characters, ASCII strings), allowing users to represent chemical structures in computer language. This notation identifies the nodes and edges of a molecular graph. Created in 1986 at the US EPA (US Environmental Research Laboratory) by [44] and further developed by Daylight Chemical Information Systems [45], this notation [44, 37]:

- Captures connectivity (chirality, ions, shapes) and stereochemical configurations;
- Is easily learned and the most popular flexible line notation used in the past three decades;
- Is based on principles of molecular graph theory, which allowed its development;
- Uses very small and natural grammar;
- Is well suited for high-speed machine processing;
- And, its machine compatibility allowed many highly efficient chemical computer applications to be designed including generation of a unique notation, constant-speed database retrieval, flexible substructure searching, and property prediction models.

Table 5: SMILES Specification Rules.

Rule	Example	Explanation
Atoms	Carbone (C): C Brome (Br): Br	Atoms are encoded as their atomic symbol.
Branches	Methane with a methyl group attached to the first carbon: CC(C)C, where (C) represents a branch specifying a methyl group. Benzene with alternating single and double bonds: c1ccccc1, where the absence of spaces signifies the sequential representation of alternating single and double bonds in the ring structure.	Branches are specified by enclosures in parentheses and can be nested or stacked. Nested are bonds within brackets that signify complex bond types or relationships. Stacked bonds can be represented sequentially without spaces to indicate different bond types or connectivity.
Charges	Positive charge: (C+) Negative Charge: (C-)	A formal charge is represented by one of the symbols + or -.
Closed Brackets	Gold: (Au) Chlorine: Cl Water (H2O): (OH2)	Elements are always in closed brackets unless they are elements of the “organic subset” (B, C, N, O, P, S, F, Cl, Br, and I), which do not require brackets unless they are charged. If hydrogens are explicitly implied brackets are used.
Hydrogen Suppression	Propane: (CH3)(CH2)(CH3)	Hydrogens are generally suppressed except when connected to other hydrogens, zero or more than one atom, or specified isotopically.
Bonds	Methane: CC Ethylene: C=C Acetylene: C#C Aromatic Bond: Benzene - c1ccccc1	SMILES defines single, double, triple, and aromatic bonds. It prefers covalent bond representations for delocalized bonds to maintain topological symmetry. Single, double, triple, and aromatic bonds are represented by the symbols, —, =, #, and :, respectively. Single and aromatic bonds may be, and usually are, omitted.
Rings	Benzene: c1ccccc1	Represented by breaking one single or aromatic bond in each ring, and designating this ring-closure point with a digit immediately following the atoms connected through the broken bond. Atoms in aromatic rings are specified using lowercase letters.
Tautomers	2-Pyridinol: Oc1ccccc1 or O1[nH]cccc1	Tautomeric structures are explicitly specified, but the choice between tautomers depends on convention and stability.
Aromaticity Detection	Benzene: c1ccccc1 (aromatic evidence, lowercase) or C1=CC=CC=C1 (Kekulé form, default)	Aromaticity in SMILES is identified based on sp ² -hybridized atoms in a ring following Huckel’s rule ($4N + 2\pi$ electrons). The notation can explicitly indicate the aromatic bond (lowercase) or follow the default aromatic assumptions, using Kekulé form.

SMILES specification rules [44, 46] are defined in detail in table 5.

The next figure of the caffeine molecule (figure 4) helps to understand deeply this notation.

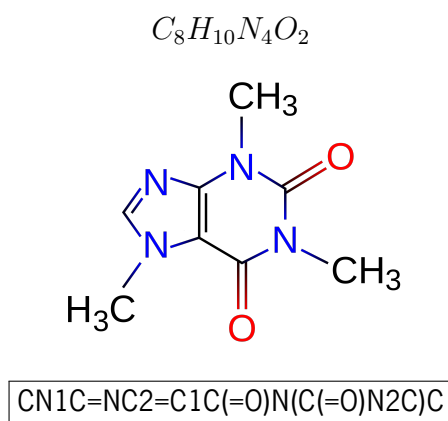


Figure 4: Caffeine molecule and its various representations. On top, we see the molecular formula, and in the middle, the molecular 2D structure. Below, the canonical SMILES notation. Some noteworthy features are: **(1) atoms:** it includes Carbon (C), Nitrogen (N), and Hydrogen (H) atoms; **(2) connectivity:** the connections between atoms are represented by the arrangement of characters and symbols. For instance, CN implies a Carbon atom connected to a Nitrogen atom, and C=O represents a double bond between a Carbon and an Oxygen atom; **(3) ring structure:** the cyclic structure of caffeine is demonstrated through the ring notation. The numbers 1 and 2 indicate the connectivity and ring closure, suggesting the formation of cyclic structures; **(4) functional groups:** functional groups like =O (carbonyl) and =NC (amidogen) signify specific arrangements of atoms within the molecule, highlighting characteristic chemical groups.

InChI

InChI aims to provide a machine-readable, unique depiction of molecular structures by organizing information into layers with a specific order [37]. InChI provides advantages in canonical representation and interoperability, but faces limitations in human readability and certain chemical representations compared to SMILES [47, 48]. It begins with a core parent structure to which further information may be added, such as layers that encode other details, including the empirical formula, skeletal connections, and hydrogen atoms. Additional layers include stereochemistry, isotopic information, and charges. The separator between layers is the forward slash (/) delimiter, and all layers start by a lowercase letter identifying the layer (except for the first layer).

The empirical formula layer shows chemical composition, while the skeletal layer (/c) details atom connections. The hydrogens (/h), charge (/q), and protonation (/p) layers specify bonds, net charge, and proton changes. Stereochemistry encodes spatial arrangement via double bond (/b), tetrahedral (/t), and isotopic (/i) sub-layers [49].

InChIKey, a hashed counterpart to InChI, provides unique identification and hierarchical, layered rep-

representations for chemical structures, facilitating canonical representations and seamless linkage within databases (interoperability) [48].

The Figure 5 visually depicts the respective layers within the standard InChI notation for [(R)-carboxy(chloro)methyl]azanium, showcasing their arrangement and hierarchical organization as prescribed by the InChI system.

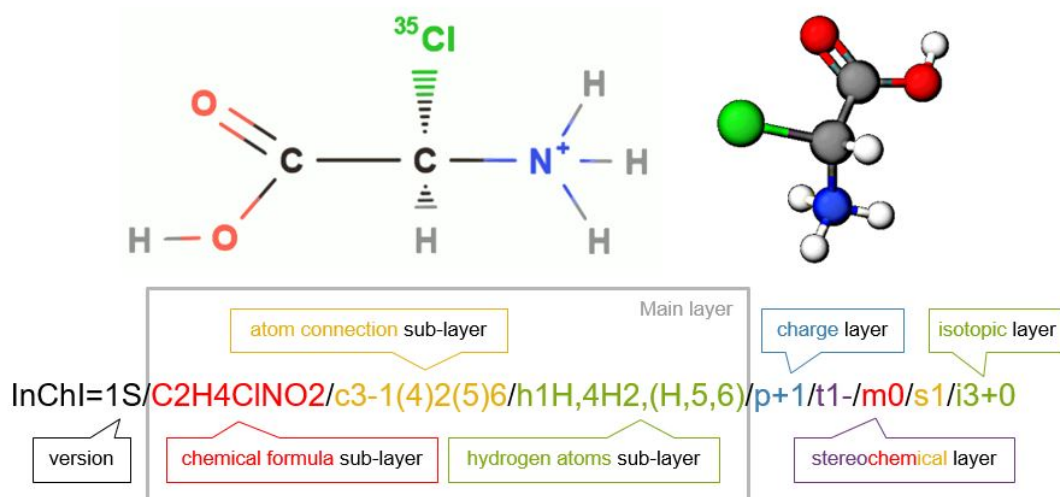


Figure 5: The primary layers within the standard InChI notation for [(R)-carboxy(chloro)methyl]azanium, representing the protonated form of 2-(³⁵Cl)chloro-R-glycine [50].

Molecular Fingerprints

Molecular fingerprints offer a condensed representation of molecular structures. This depiction is essential for diverse applications in drug discovery, similarity analysis, and virtual screening, has seen in section 3.1. These fingerprints exhibit diverse types and complexities [51].

They are numerical representations (a bit string) derived from molecular structures. These fingerprints encode structural features, such as the presence or absence of particular substructures, functional groups, or topological properties. Each unique structural aspect is mapped to a specific bit or value (0 or 1), creating a pattern or fingerprint that characterizes the molecule (fingerprint encoding) [52, 51].

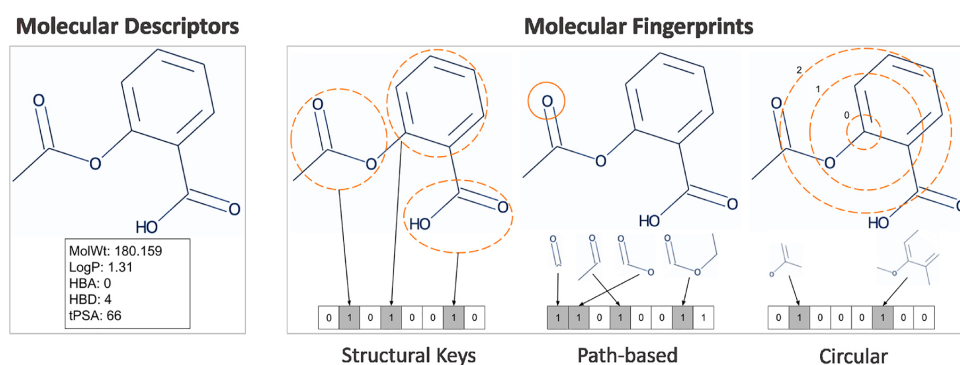


Figure 6: Several representations of Salicylic acid ($C_6H_4(OH)COOH$), using molecular fingerprints. A common depiction used in ML tasks. In the structure, it is visible the benzene ring with two other groups attached. The ring and the groups are encoded as bits, with a value of 1, in the fingerprint. Bits left turned off, with a value of 0, indicate the absence of the features. [53].

As shown in Figure 6, several types of fingerprints exist, including structural keys, hashed fingerprints, circular fingerprints, and pharmacophore fingerprints. These vary in their generation methods, capturing different aspects of molecular structure and properties. They can be computed based on molecular graphs, connectivity, atomic properties, or physicochemical attributes [39]

These fingerprints are often classified into distinct categories, including [51, 39]:

- **Topological fingerprints** map molecular feature paths within a set bond limit, occasionally leading to bit collisions, but ensuring identical substructure representation in different molecules. They are adept at clustering structurally akin compounds;
- **Structural key fingerprints** encode specific substructures or fragments within a molecule: they associate bit presence with predefined functional groups or substructure motifs, devoid of specific bit meaning. (e.g. Molecular Access System (MACCS) keys [54] or PubChem fingerprints [55]);
- **Circular fingerprints** encode molecular substructures based on atom neighborhoods within a specific radius. They can record atom connections within radial environments, expanding with increased atom shells (e.g. Extended Connectivity Fingerprints (ECFP) or Functional Connectivity Fingerprints (FCFP) [56]);
- **Pharmacophore fingerprints** represent molecular features essential for biological activity. They focus on triplet or quadruplet features, encoding feature combinations and distances in 2D or 3D patterns, potentially generating lengthy bit strings capturing numerous properties;
- **Hybrid fingerprints** combine different fingerprinting approaches;

- **Miscellaneous fingerprints** concentrate on protein-ligand interactions rather than conforming to conventional fingerprint categorizations.

Ranging from basic representations of topological distances or functional group occurrences to intricate 3D pharmacophore arrangements, their effectiveness is not always contingent on complexity. Despite the potential of 3D fingerprints to encapsulate more property-relevant data, studies indicate that their performance in applications like virtual screening does not consistently outstrip 2D counterparts. Surprisingly, 2D atom pair descriptors have demonstrated remarkable performance in perceiving 3D molecular shapes. Extensions of these 2D descriptors, like 2D pharmacophoric fingerprints (Xfp), have shown improved recognition of biologically active compounds and the ability to discern stereoisomers [51]

Furthermore, the choice of similarity measures, usually employing Tanimoto coefficients is not absolute. Alternatives such as Dice or Tversky similarity measures have proven effective in specific contexts, showcasing the diversity of evaluation approaches in molecular fingerprint analysis [51].

$$\text{Tanimoto} = \frac{N_{ab}}{N_a + N_b - N_{ab}} \quad \text{Dice} = \frac{2N_{ab}}{N_a + N_b + 2N_{ab}} \quad \text{Tversky} = \frac{N_{ab}}{\alpha N_a + \beta N_b + N_{ab}}$$

where:

- N_a and N_b represent the number of bits set to one in the fingerprints representing compound a and compound b, respectively;
- N_{ab} represents the number of bits set to one in both compound a and compound b;
- α and β are parameters in the Tversky similarity measure, allowing asymmetry in the comparison between compounds.

3.2.2 Molecular Graphs

Molecular graphs serve as a graphical representation encapsulating the intricate structural arrangement of atoms within a molecule. Within this representation, atoms take the form of nodes, while chemical bonds are depicted as edges connecting these nodes. The degree of a node within a molecular graph serves as a key indicator, corresponding to the number of bonds an atom forms. In graph theory, this representation provides valuable insights without exhaustive exploration of intricate details [57].

Matrix notations or linear codes are the methods developed for mapping molecular graphs into the computer-friendly data. A variety of matrix systems (2D descriptors) were designed, where the numeric representation depends upon the scheme used for coding atoms and bonding.

Popular notations include the following matrices: adjacency, atom connectivity, distance (topological or geometrical), bond, bond electrons, incidence [39].

Despite this variety, molecular graphs are commonly implemented using adjacency matrices (visible in Figure 7) to store detailed topological data. Nodes and edges can carry properties, such as spatial location and bond order/type, allowing for the encoding of intricate structural information in a readily processable form [57].

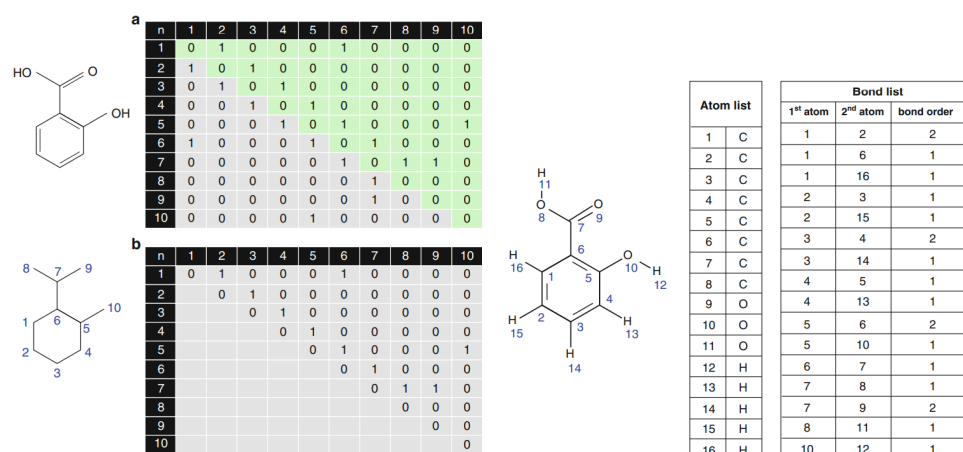


Figure 7: On the left, two possible forms of an adjacency matrix of salicylic acid: $a_{ij} = 1$, if i and j are bonded or 0 if not bonded [39]. On the right, the connection table of salicylic acid [39].

Although a matrix system performs well, the number of entries needed to describe a molecule squares with the increase in the number of atoms in the molecule, making it impractical for storing large molecules or large datasets. Therefore, a CT (visible in Figure 7) is an alternative method that allows to present molecular graphs in a form of lists of atoms and bonds in a molecule, which increase linearly with the number of atoms. The CT notation has now become the predominant form of representing molecular structures [39].

3.2.3 Three-Dimensional Representations

Representing a molecule solely through a CT may overlook its 3D conformation, limiting the understanding of its spatial arrangement. To overcome this, various methods have been devised to provide a more comprehensive view of molecular geometry.

One widely adopted approach involves linking a coordinate system to the CT, effectively placing the molecule within a Cartesian space. This method assigns each atom in the CT three spatial coordinates (x, y, z), enabling a detailed 3D representation [38].

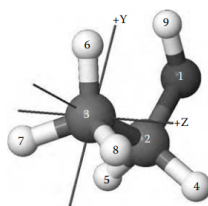


Figure 8: 3D Cartesian model of ethanol [38].

An alternative strategy utilizes internal coordinates, such as bond lengths, bond angles, and torsion angles, to describe each atom's position relative to its neighbors, offering flexibility in representing molecular structures [38].

While these approaches yield insightful 3D graphs, they may fall short of capturing the full volume occupied by a molecule. In reality, a molecule is enveloped by an electron cloud, a crucial determinant of its properties. Molecular surfaces address this limitation by representing a molecule as a closed surface that delimits its volume, intricately linked to electron density thresholds within the surrounding electron cloud.

The utilization of molecular surfaces introduces a visual delineation of the electron influence of the cloud, characterizing not only the spatial extent of the molecule, but also allowing for the association of properties, such as electrostatic or hydrophobicity potential at specific locations [38]. This method adds another layer of sophistication to the understanding of molecular behavior, enabling a more nuanced exploration of chemical properties and interactions.

3.2.4 Embeddings

Embeddings offer a powerful tool for representing complex molecular structures in a manner that is amenable to ML algorithms. By transforming molecular information into fixed-size, continuous vector representations, embeddings enable models to learn and recognize patterns that would be otherwise difficult to discern.

One of the most common approaches to generating molecular embeddings is through molecular fingerprints. A prominent example is the Extended-Connectivity Fingerprint (ECFP), which generates unique vectors based on the molecular graph's topology. In this context, graph-based embeddings have emerged as a promising alternative. By representing molecules as graphs, where atoms are nodes and bonds are edges, it is possible to capture both local and global structural properties.

Recent advancements have introduced several DL-based methods for creating molecular embeddings. Techniques such as SMILESVec and Mol2Vec utilize SMILES representations to learn embeddings through

Unsupervised Learning (UL) approaches, effectively capturing chemical intuition from the molecular structure. Additionally, Seq2Seq Fingerprint employs Recurrent Neural Networks (RNNs) to directly learn from SMILES strings, while PaccMann and SA-BiLSTM leverage supervised learning to incorporate biological target information, enhancing predictive performance.

These embeddings provide a rich representation of molecular features. However, empirical comparisons have shown that while these embeddings can be competitive, traditional representations such as molecular descriptors and fingerprints often yield comparable or superior performance [58].

3.3 Databases

In literature, there are several database taxonomies that can be found, each one characterized by the format model they follow, their content, and the data type of the information they store, as shown in Table 6.

Table 6: Database taxonomies found in literature [39].

Formal model	Data type	Content
Network	Biological activity	Literature
Relational	Spectroscopy	Factual
Object-based	CrystallographyGenetic	Numeric
Hierarchical	Proteomics	Meta-databases (entries for other databases)
		Catalogs of chemical compounds
		Research projects databases
		Structure databases where 3D molecular structures are entries
		Small molecules
		Bioinformatics data, i.e., proteins and genetic (DNA and RNA) data
		Chemical reaction databases
		Internet, e.g. web search engine Google

Chemoinformatics relies on many primary types of databases, each serving distinct purposes [33]: literature, factual, and structural.

Within the realm of **literature databases**, bibliographic sources, such as Chemical Abstracts Service (CAS) [59] index and retrieve publications, while full-text databases like ACS Journals [60] provide access to complete articles.

Turning to **factual databases**, these platforms offer immediate textual or alphanumeric informa-

tion: numerical databases, exemplified by Beilstein [61], focus on physicochemical values, while meta-databases like Gale Directory [62, 63] offer insights into content, and research project databases such as Environment Research in Progress (UFORDAT) [64, 65] categorize information based on ongoing projects.

Additionally, **structure and reaction databases** like SPRESI [66] play a central role, storing crucial data on chemical structures, reactions, and their outcomes. Lastly, Molecular Biology Databases such as GenBank [67] and European Molecular Biology Laboratory (EMBL) [68] contribute to the interdisciplinary nature of chemoinformatics by housing genetic information.

Presently, online access to databases via the Internet is a standard practice, encompassing both charged licensed and freely available systems across diverse organizational frameworks [39].

The overarching objective of these databases is to efficiently manage and retrieve chemical and biological information, thereby supporting a wide array of research endeavors across disciplines. Some other relevant databases to this specific project are listed below, with their detailed description and type, according to the various taxonomies found in literature.

Table 7: Chemoinformatics Databases with Types.

Database Name	Description	Type
PubChem [69]	A database of chemical molecules maintained by the National Center for Biotechnology Information (NCBI).	Catalogs of chemical compounds
ChEMBL [70, 71]	A large-scale bioactivity database for drugs and compounds.	Bioinformatics data, i.e., proteins and genetic (DNA and RNA) data
ChemSpider [72]	A chemical structure database with information on various compounds.	Structure databases where 3D molecular structures are entries
ZINC [73]	A database of commercially available compounds for virtual screening.	Catalogs of chemical compounds
DrugBank [74]	A comprehensive database combining chemical and pharmacological data.	Catalogs of chemical compounds
PubMed [75]	A vast collection of bibliographic information and abstracts from a wide range of biomedical literature.	Literature

As shown in Table 7, PubChem, ChEMBL, and ZINC are categorized under “Catalogs of chemical compounds”. These databases primarily serve as repositories for chemical molecules, bioactivity data for drugs and compounds, and commercially available compounds for virtual screening, respectively.

ChemSpider is classified as “Structure databases where 3D molecular structures are entries”. It is known for its extensive collection of chemical structures and related information.

DrugBank, like PubChem, ChEMBL, and ZINC, falls under the category of “Catalogs of chemical compounds”, standing out as a comprehensive database combining chemical and pharmacological data, offering a wealth of information on drugs and related compounds.

Lastly, PubMed is classified as “Literature” database, serving as a vast collection of bibliographic information and abstracts covering a wide spectrum of biomedical literature.

3.4 Software, packages and tools

Cheminformatics employs computational methods and software tools for extracting insights from chemical data. These tools fulfill various purposes, such as molecular structure handling, data analysis and workflows, and molecular docking. Python, recognized for its versatility, is widely used in cheminformatics with libraries like RDKit and ChemPy. Facilitating the reproducibility of published work and simplifying the development of new approaches, these libraries include the following examples:

- **RDKit [76]:** An open-source toolkit for cheminformatics and molecular informatics (available in C++ and Python). It provides a wide range of functions for handling and analyzing chemical structures, including molecular similarity, substructure searching, and descriptor calculations (capable of converting between molecular representations, fingerprint calculation, generating various molecular descriptors, etc.);
- **Open Babel [77]:** An open-source chemical toolbox designed to speak the many languages of chemical data, allowing conversion between different chemical file formats;
- **KNIME Analytics [78, 79]:** A data analytics and integration platform that allows users to design and execute data workflows, making it suitable for cheminformatics data analysis and visualization;
- **AutoDock Vina [80]:** A molecular docking program used for predicting the binding modes of small molecules to macromolecular targets. It is widely used in virtual screening and drug discovery applications in cheminformatics;
- **CDK [81, 82]:** Library in Java and R, capable of utilizing a variety of molecular representations, 3D geometry generation, substructure search, fingerprint calculation, and many other functionalities.

Chapter 4

Machine Learning

Artificial Intelligence (AI) is a broad field of computer science that focuses on creating systems and machines capable of performing tasks that would typically require human intelligence. Thus, AI is the capacity of such systems to simulate human cognitive functions, including learning, reasoning, advanced problem-solving, perception, language understanding, and decision-making. This capacity is based on analytical models that generate predictions, rules, answers, or related outcomes, through a variety of approaches and techniques, ranging from rule-based systems to complex ML algorithms [83]. Conceptually, AI includes ML, which itself has DL as one recent sub-field, as shown in Figure 9.

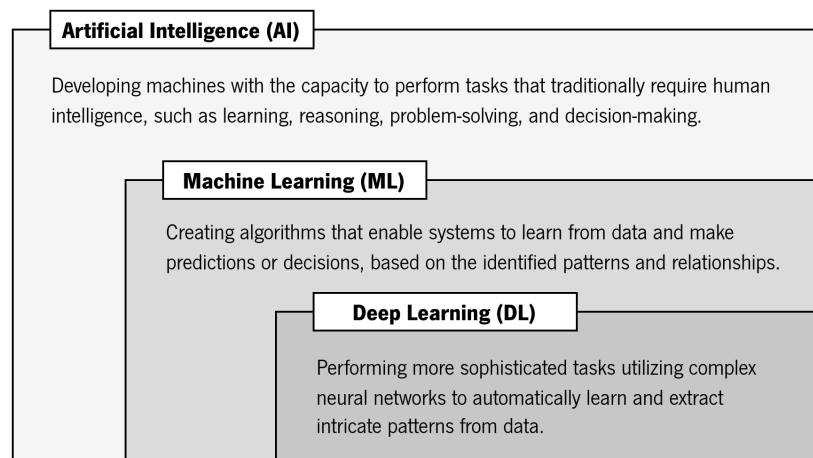


Figure 9: Diagram of ML traditional classes and their goals.

ML focuses on the development of algorithms and statistical models that enable computers to perform tasks without explicit programming. These algorithms include regression models, instance-based algorithms, decision trees, Bayesian methods, and ANN. The fundamental idea behind is to empower machines to learn from examples and observations (data), automatically identify patterns and meaningful relationships, and make decisions or predictions based on that knowledge [84].

As highlighted by [85] and [86], the practicality of new programming frameworks, increased data

availability, and widespread access to robust computing power have fueled the surge in the adoption of ML. This progress was also marked by the evolution of ANN into increasingly deep architectures—a breakthrough known as DL.

DL is also a subfield of ML that focuses on the development and application of ANNs with multiple layers, also known as Deep Neural Network (DNN)s. The primary goal of DL is to enable machines to learn and represent complex patterns and hierarchical structures within data [85], leading to more sophisticated and accurate decision-making capabilities (progressively extract higher-level features from raw data).

4.1 Types of Machine Learning

4.1.1 Supervised Learning

Supervised Learning (SL) is a paradigm where learning is carried out through the provision of the desired outcomes. This mechanism needs supervision with an appropriate training model [87]. The respective algorithm learns from labeled training data, which includes both input features and the corresponding output labels. In the training process, the algorithm aims to learn a mapping function from the input to the output, enabling it to make predictions or decisions when new input data is presented.

SL can be further categorized into Regression, where the algorithm predicts continuous values (quantitative labels), and Classification, where the algorithm predicts categorical classes (qualitative labels). As an illustration, in the prediction of the number of active users subscribed to a platform in a specific time range, the predictive model would rely on the input characteristics, such as the number of sold products and user reviews, to forecast the engagement of each user in the platform.

4.1.2 Unsupervised Learning

UL is a paradigm where learning takes place without the provision of the desired outcomes. This process involves the discovery of patterns/regularities in the input data and the user does not have to supervise the model [87]. The training of the model use input data without labeled responses. In UL, the algorithm explores the structure of the data to extract meaningful insights, such as identifying patterns, clusters, or relationships within the data. It is usually employed as an exploratory tool, including common tasks as clustering, where similar data points are grouped together, and dimensionality reduction, where the goal is to reduce the number of input variables while retaining important information [84]. As an illustration, consider clustering similar customer segments, based on their purchasing behavior without predefined

categories.

4.1.3 Reinforcement Learning

In Reinforcement Learning (RL), an agent learns to make sequential decisions by interacting with an environment. The agent follows a dynamic approach and receives feedback in the form of rewards or penalties based on its actions, where the goal is to learn a policy that maximizes the cumulative reward over time [87]. Some essential components of RL are the “state” (current situation) and the “optimal solution” (best solution), achieved when the system receives the maximum positive feedback [87]. RL is well-suited for problems involving decision-making and control, and it has been successfully applied in domains such as game playing, robotics, and autonomous systems [88].

4.1.4 Generative Learning

Generative Learning is a recent subfield of ML, that involves systems capable of generating new content, data, or outputs, often mimicking or creating novel instances based on patterns learned from existing examples, usually dealing with unlabeled data. The goal is to model the probability distribution of the input data and use it to generate new samples that are similar to the original data [86]. This approach encompasses various techniques, such as generative models and neural networks, enabling machines to produce original and realistic content, such as images, text, or other forms of data.

4.2 Basic principles

4.2.1 Concepts and Terms

Features refer to the input variables or attributes that are used to make predictions in a ML model. These can include numerical values, categorical variables, or any other relevant information. **Labels**, on the other hand, represent the output or the prediction that the model aims to make based on the input features. In a SL scenario, the model is trained on a dataset comprising both features and corresponding labels.

The **dataset** is typically partitioned into three distinct subsets: the training data, the validation data, and the test data.

The training data, commonly referred to as the **training set** serves as the foundation for training the model, enabling it to discern intricate patterns and relationships between features and labels. Through

this process, the model iteratively adjusts its parameters to maximize performance.

Conversely, the validation data, often denoted as the **validation set**, is used to fine-tune hyperparameters and to assess the performance of the model throughout the training phase. **Hyperparameters**, distinct from the learned parameters, dictate the learning process of the model and include variables such as the learning rate or the number of hidden layers within a neural network. By subjecting the model to the test set and evaluating its performance, one can iteratively refine hyperparameters to enhance the generalization capacity of the model without succumbing to overfitting, whereby the model excessively tailors itself to the intricacies of the training data.

Subsequently, the test data, commonly termed the **test set**, assumes the critical role of evaluating the proficiency of the model on novel, unobserved data instances, thereby offering valuable insights into its generalization prowess. This set is instrumental in appraising the ultimate performance of the model, post-hyperparameter tuning and exhaustive training on the complete training set.

Bias refers to the error introduced by approximating a real-world problem, which may be complex, by a simplified model. High bias can lead to underfitting, where the model fails to capture the underlying patterns in the data. **Variance**, on the other hand, is the model's sensitivity to fluctuations in the training data. High variance can result in overfitting, where the model performs well on the training data but fails to generalize to new data. The **Capacity** refers to the ability of the model to fit a wide range of functions, which can be controlled by adjusting hyper-parameters.

In **singletask training**, the model is optimized for a single task, such as classification, regression, or segmentation, with the goal of maximizing performance on that specific task. This approach is commonly used when there is a well-defined application and a dataset focused on a single objective. On the other hand, **multitask training** aims to train a single model to perform multiple tasks at the same time. This can be advantageous in scenarios where tasks are related, allowing the model to share knowledge between them, leading to greater efficiency and, in some cases, better overall performance.

Proper evaluation of ML models is crucial for assessing their performance, capacity, and robustness. One of the methods commonly used is CV that is followed by a systematic parameter tuning procedure, culminating in the selection of the top final models.

CV is a widely used resampling method to estimate the performance of a model while mitigating the risk of overfitting. In k -fold CV, the dataset is split into k equally sized folds. For each fold, the model is trained on $k - 1$ folds and validated on the remaining fold. This process is repeated k times, ensuring that each fold is used for validation once. The final performance metric is obtained by averaging the results from all k iterations, which provides a robust estimate of model performance on unseen data. This

technique is beneficial when the available dataset is relatively small, as it maximizes the use of all available data points, allowing for a more reliable evaluation by reducing variance in the performance estimates. The selection of the parameter k (commonly 5 or 10) depends on the trade-off between computational cost and performance estimation accuracy.

Parameter tuning, also known as hyperparameter optimization, is the process of selecting the best hyperparameters for a model to optimize its performance. Unlike model parameters, which are learned during training, hyperparameters are set prior to the training process and can significantly influence the model's behavior. In this context, the Tree-structured Parzen Estimator (TPE) is used for hyperparameter tuning. TPE is a Bayesian optimization method that models the objective function to efficiently explore the hyperparameter space. Unlike grid search and random search, which evaluate all or randomly selected combinations of hyperparameters, TPE uses a probabilistic model to focus on regions of the hyperparameter space where good results are more likely. It divides the hyperparameter space into two models: one for hyperparameter configurations associated with high performance (good configurations) and another for low performance (bad configurations). TPE then uses these models to evaluate the most promising configurations in each iteration. TPE is particularly effective for tuning models with complex or high-dimensional hyperparameter spaces, offering a balance between exploration and exploitation. It is computationally efficient and often yields better results than traditional search methods, especially when limited computational resources are available.

After tuning the hyperparameters using the TPE algorithm and evaluating models with CV, the final model is selected based on its performance on the validation set. The best-performing hyperparameters identified through TPE are used to retrain the model on the full training dataset. The final evaluation on the test set ensures that the selected model is the most capable of generalize (performs well on unseen data).

4.2.2 Performance Measure

The performance measure varies depending on the ML task at hand.

As previously delineated, the dataset undergoes division into training, validation and test sets. Subsequently, the training data facilitates model training, which is then employed to predict all instances within the test data. These predictions are then compared against the corresponding ground-truth values of the test set. Therefore, evaluation metrics essentially serve as a mean to quantify the quality of a model by comparing its predictions with actual values.

Evaluation metrics play a pivotal role in assessing and quantifying the performance of a given model,

bestowing upon it reliability and robustness in generating results. The evaluation process serves the primary purpose of discarding methods that exhibit poor performance while further refining those that show promise.

To assess the generalization capability of models comprehensively, it is imperative to employ not only practical and effective estimation methods but also performance measures capable of quantifying such generalization ability. Diverse performance measures cater to the varying demands of tasks, yielding distinct evaluation outcomes. Essentially, the quality of a model is a relative concept contingent upon the algorithm, data, and task requirements [89].

Confusion Matrix

The confusion matrix, also known as the contingency matrix, is a two-dimensional table consisting of actual values and predicted values. In binary classification, data instances are typically predicted to be either positive or negative. A positive label signifies the presence of something, while a negative label indicates the absence of the same. Each predictive label can then be categorized into four possible designations [90]:

- **True Positive (TP)**: When a positive instance is correctly predicted as positive;
- **True Negative (TN)**: When a negative instance is correctly predicted as negative;
- **False Positive (FP)**: When a negative instance is incorrectly predicted as positive;
- **False Negative (FN)**: When a positive instance is incorrectly predicted as negative;

Table 8: Confusion/Contingency Matrix.

		Predicted	
Actual		Positive	Negative
	Positive	True Positives (TP)	False Negatives (FN)
	Negative	False Positives (FP)	True Negatives (TN)

A confusion matrix, typically presented as shown in table Table 8, is then a 2×2 matrix that contains the counts of TP, TN, FP, and FN observations, utilized to compute several other metrics for evaluation of binary classifiers. The most commonly used are shown in table 9.

Common Classification metrics

Table 9: Common classification metrics.

Metric	Description	Formula
Accuracy	Proportion of correctly classified instances in the set of all the instances.	$\text{Acc} = \frac{TP + TN}{TP + TN + FP + FN} \in [0, 1] \quad (4.1)$
Precision	Proportion of the instances classified as positive that are truly positive.	$\text{Prec} = \frac{TP}{TP + FP} \in [0, 1] \quad (4.2)$
Specificity, True Negative Rate (TNR)	Proportion of negative samples that were correctly classified (the truly negative instances).	$\text{Spec} = \frac{TN}{TN + FP} \in [0, 1] \quad (4.3)$
Recall, Sensitivity, True Positive Rate (TPR)	Proportion of actual positive cases that were correctly identified (the truly positive instances)	$\text{Rec} = \frac{TP}{TP + FN} \in [0, 1] \quad (4.4)$
F1-Score	Harmonic mean of precision and recall.	$F1 = \frac{2 \times \text{Prec} \times \text{Rec}}{\text{Prec} + \text{Rec}} \in [0, 1] \quad (4.5)$
Cohen's Kappa Coefficient	Level of agreement between two raters or judges when they categorize or classify items into N mutually exclusive categories (agreement between the predicted and the real classes). $\text{Pr}(a)$ represents the proportion of observed agreement between the two raters. $\text{Pr}(e)$ represents the agreement that would be expected to occur by chance alone.	$\kappa = \frac{\text{Pr}(a) - \text{Pr}(e)}{1 - \text{Pr}(e)}, \in] - \infty, 1] \quad (4.6)$ $\text{Pr}(e) = \frac{(TP + FN)(TP + FP) + (TN + FP)(TN + FN)}{(TP + TN + FP + FN)^2}$
Balanced Accuracy	Average number of TP (Recall) and TN (Specificity). Usually the go-to metric in instances where there is an imbalance in classes.	$\text{Bal-acc} = \frac{\text{Recall} + \text{Specificity}}{2}, \in [0, 1] \quad (4.7)$
Receiver Operating Characteristic (ROC) Curves	Graphical representation with diagnostic capacity of a binary classifier system. The curve analysis is done using its graphical representation, however, the area under the curve (ROC AUC) can also be calculated, ranging between 0 and 1. Where 1 represents a perfect classifier, 0.5 shows that the classifier is approximately random and a ROC area of 0 indicates that the classifier is always wrong. The ROC curve can be used to evaluate the performance of a classifier for different classification thresholds. It is constructed by measuring the FPR and the TPR for each classification threshold possible.	$\text{AUC-ROC} = \int_0^1 \text{TPR} * \text{FPR} d\text{FPR} \quad (4.8)$

4.3 Traditional Machine Learning

4.3.1 Supervised Learning

Support Vector Machine

SVM, a method within SL, primarily serves for classification tasks but also extends to regression problems. Its foundation rests upon the notions of margin and support vectors (figure Figure 10). Fundamentally, SVM endeavors to construct a hyperplane in the sample space, discerning it as the optimal separator for samples belonging to different classes. While multiple hyperplanes may meet the criteria, typically, the one selected exhibits the highest generalization capacity and yields the most robust classification outcomes, as expounded by sources such as [89] and [91].

Support vectors denote the instances proximate to the margin, with the margin representing the total distance between two support vectors of distinct classes and the separating hyperplane. Consequently, SVM addresses the task of maximizing this margin, ensuring effective class separation. SVM's underpinnings lie in statistical learning theory, also recognized as VC (Vapnik and Chervonenkis) theory.

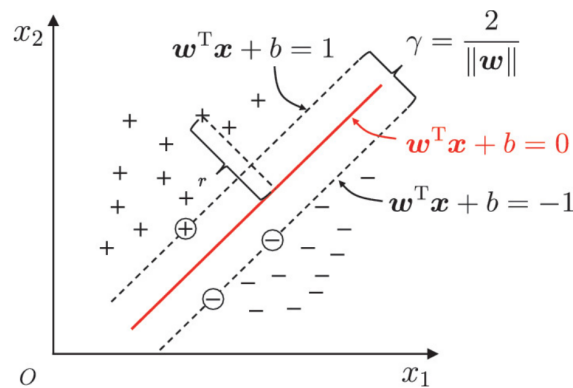


Figure 10: Main concepts of SVM: support vectors and margin [89].

$$\text{Minimize: } \frac{\|w\|^2}{2}$$

$$\text{Subject to: } y_i(w^T x_i + b) \geq 1, \quad \text{for all } (x_i, y_i) \in D$$

In certain cases, transforming samples from the original feature space to a higher-dimensional one can make them linearly separable. This technique is known as the “kernel trick”, and there are a number of kernels available, a popular one is the radial basis function kernel, or gaussian kernel [86]

Decision Trees and Random Forests

Decision trees are fundamental SL models that recursively partition data through a series of rules, aiming to maximize information gain [92]. They follow the “divide-and-conquer” strategy, that is, separating the instances under certain conditions to later classify them. As shown in Figure 11, the successive divisions carried out create a representative structure of the tree model, hence the name decision trees [93]. They offer interpretability and mimic human decision-making processes, making them popular in various fields [86, 94]. However, they face challenges such as overfitting and limited predictive power. Ensemble methods like Random Forests address these limitations by combining multiple decision trees [89, 94].

Decision trees are constructed using a top-down greedy approach, starting with a root node and recursively partitioning data based on feature splits [86, 94]. The choice of split is based on maximizing information gain, measured through metrics like Gini index for classification and Mean Squared Error (MSE) for regression [92, 94]. The depth of the tree affects the bias-variance trade-off, with deeper trees reducing bias but increasing variance [86].

Random Forests combine multiple decision trees to improve predictive performance and robustness [89, 94]. Each tree is trained on a bootstrapped sample of the data, and during construction, only a random subset of features is considered for each split, ensuring diversity among trees. The final prediction is determined by aggregating the outputs of individual trees, either through majority voting for classification or averaging for regression. Random Forests mitigate overfitting and are robust to noise, although the number of trees in the ensemble affects computation time [92, 94].

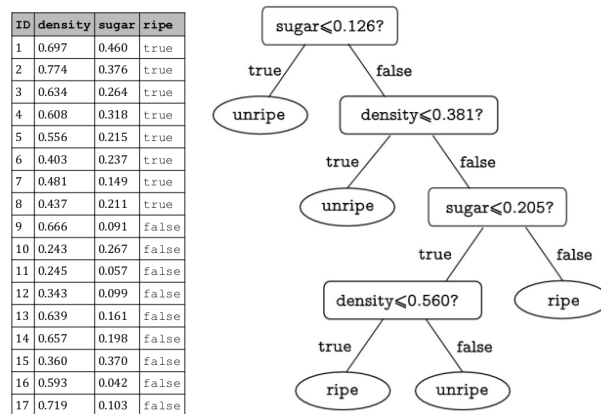


Figure 11: On the left side, an example dataset to predict the target `ripe`. On the right side, the result decision tree [89].

Bayesian Classifiers

Bayesian classifiers are a family of probabilistic classifiers based on Bayes' Theorem, which describes the probability of an event, based on prior knowledge of conditions that might be related to the event. To each new example is assigned the probabilities of belonging to the several classes, and the class with the highest probability is the one chosen. These classifiers are particularly useful in scenarios where the goal is to predict the category or class of a given data point [95, 89].

The **Naive Bayes classifier** is a specific type of Bayesian classifier that assumes independence between features. This simplification allows the classifier to be highly efficient and effective in many practical applications. The decision rule for the Naive Bayes classifier is to choose the class with the highest posterior probability [89].

There are several variants of the Naive Bayes classifier, each tailored to different types of data. The **Gaussian Naive Bayes** assumes that the continuous features follow a normal (Gaussian) distribution. This variant is suitable for data with continuous features that can be modeled using a Gaussian distribution. It calculates the probability of a given class based on the probability density function of each feature, assuming independence between features.

Artificial Neural Networks

Research on neural networks started quite a long ago with the first definition from [96]: Artificial Neural Networks (ANNs) are massively parallel interconnected networks of simple (usually adaptive) elements and their hierarchical organizations which are intended to interact with the objects of the real world in the same way as biological nervous systems do. The basic element (unit) of a neural network is the neuron, commonly referred to as "perceptron" [86]. In biological neural networks, neurons communicate by sending neurotransmitters to adjacent neurons when they become "excited." This process alters the electric potentials of the interconnected neurons. When the electric potential surpasses a certain threshold (i.e., bias), the neuron is activated (i.e. "excited") and subsequently transmits neurotransmitters to other neurons (i.e. the "transfer function") [89].

In 1943, [97] abstracted this biological process into a simplified model known as the McCulloch–Pitts model (M-P neuron model), which remains in use today. According to this model, as depicted in Figure 12, each neuron receives input signals from multiple neurons through weighted connections. The weighted sum of these incoming signals is then compared against a predetermined threshold, and the resulting output signal is generated by the activation function. Activation functions are essential in neural networks as they enhance the capacity of the network to learn and generalize from data [98]. First, they introduce

non-linearity, enabling the network to model complex relationships and create intricate decision boundaries. Second, they act as thresholding mechanisms to control neuron activation, normalize outputs for interpretability, and affect gradient-based optimization, crucial for efficient training.

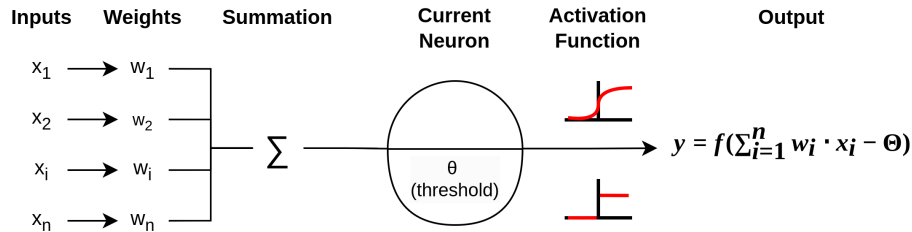


Figure 12: The M-P neuron model. Single neuron performing a weighted sum of the inputs and applying a non-linear activation function (sigmoid and step functions) to the result.

ANNs comprise then interconnected neurons arranged in multiple layers, facilitating the transformation of input data into meaningful output. Typically, they consist of an input layer, one or more hidden layers, and an output layer (Figure 13). The Multi-Layer Perceptron (MLP) architecture, widely employed in ANNs [92], leverages the stacking of consecutive non-linear activation functions to enable the model to learn non-linear transformations of its inputs [98, 86]. This arrangement allows ANNs to approximate complex non-linear functions, expanding their applicability in various domains.

The architecture of an ANN crucially determines its performance and capabilities. The number of neurons, their arrangement into layers, and the connections between them collectively define the architecture of the network. The depth, referring to the number of layers, and the width, indicating the number of neurons per layer, are primary considerations in their designing. These parameters influence the network's capacity to capture and model intricate relationships within data [89].

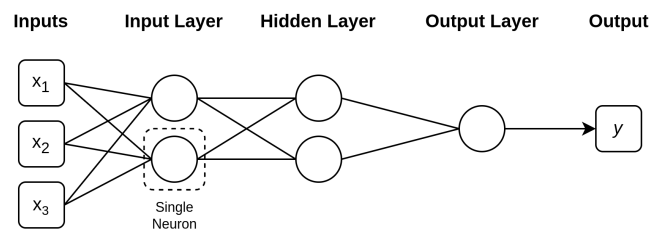


Figure 13: Small ANN with five neurons organized into three layers. Namely a input layer with two units, a hidden layer with two units, and a output layer with a single unit.

The **learning process** of neural networks involves adjusting connection weights and thresholds

based on training data. Neural networks learn from experience, refining their internal parameters to minimize prediction errors and improve performance. [89]. The key learning steps are [92, 86, 98, 99]:

1. **Initialization:** Initially, the parameters of the neural network, including the weights of the connections between neurons and any biases, are initialized. Commonly techniques are Glorot, He and random initializations;
2. **Forward Propagation:** Input data is fed into the neural network, and computations are performed layer by layer to generate predictions or outputs. Each neuron computes a weighted sum of its inputs and applies an activation function to produce an output that is passed to the next layer;
3. **Loss Calculation:** After the forward pass, the output of the network is compared to the actual or target output using a predefined loss function. This loss function quantifies the error between the predicted and actual values.
4. **Backpropagation:** Involves calculating the gradients of the loss function with respect to the network parameters (weights and biases) using the chain rule of calculus. These gradients indicate how the loss would change if each parameter were adjusted slightly;
5. **Parameter Update:** With the gradients computed during backpropagation, the parameters of the neural network, such as weights and biases, are updated iteratively to minimize the loss function. This step involves adjusting the parameters by small amounts in the direction that reduces the loss, as determined by the optimization algorithm, usually the gradient descent;
6. **Iterative Optimization:** Steps 2-5 are repeated iteratively for a specified number of epochs or until a convergence criterion is met. During each iteration, the network learns from the training data by adjusting its parameters to minimize the loss function.

4.3.2 Unsupervised Learning

Principal Component Analysis

Principal Component Analysis (PCA) is a linear dimensionality reduction technique that transforms the data into a smaller set of representative variables (orthogonal/principal components), which collectively explain most of the variability in the original set [100]. PCA is particularly useful for reducing the complexity of high-dimensional datasets while preserving as much variance as possible, making it easier to visualize and analyze large datasets.

T-Distributed Stochastic Neighbor Embedding

t-SNE is a non-linear dimensionality reduction method often used for visualizing high-dimensional data in two or three dimensions. Unlike PCA, which focuses on maximizing variance, t-SNE emphasizes preserving the local structure of the data, making it highly effective for visualizing clusters and patterns in complex datasets. In the context of t-SNE (t-distributed Stochastic Neighbor Embedding), two crucial parameters and concepts that shape the behavior of the algorithm are perplexity and divergence.

Perplexity is a hyperparameter that controls the balance between local and global aspects of the data when constructing the probability distribution. It can be seen as a smooth measure of the effective number of neighbors each point considers. Mathematically, perplexity is related to the variance of the Gaussian distributions used in high-dimensional space, and it is calculated as: $Perp(P_i) = 2^{H(P_i)}$, where $H(P_i)$ is the Shannon entropy of the probability distribution over neighbours for a data point i . Low perplexity values (e.g., 5-30) focus on the preservation of local structures in the data. This means that points in close proximity in the original high-dimensional space are more likely to remain close in the lower-dimensional embedding. High perplexity values allow t-SNE to take into account more distant points in the dataset, focusing more on the global structure and relationships between distant clusters.

Kullback-Leibler Divergence (KL Divergence) is the cost function that t-SNE minimizes to create the final low-dimensional embedding, measuring how different two probability distributions are from one another. In t-SNE, a lower and higher distributions are compared, the lower is constructed based on the distances between points in the lower-dimensional embedding, while, in the higher one, points are neighbours in the high-dimensional space. KL Divergence, quantifies how much the two distributions (P and Q) deviate from one another, being expressed as $KL(P \parallel Q) = \sum_{x \in X} (P(x) \log(\frac{P(x)}{Q(x)}))$, where x is a point from the sample space X . Being so, t-SNE seeks to minimize the KL Divergence between these two distributions by iteratively adjusting the low-dimensional positions of the points. A high KL Divergence means that the low-dimensional embedding poorly represents the original high-dimensional distances, while a low KL Divergence implies that the embedding closely resembles the high-dimensional structure, meaning similar points in the original space remain close in the lower-dimensional embedding.

K-means clustering

K-means is a popular algorithm for partitioning data into distinct groups based on their similarities. It works by iteratively assigning data points to the nearest cluster centroid and updating the centroids until convergence. is widely used in exploratory data analysis, where the goal is to uncover inherent groupings within the data. The Within-Cluster Sum of Squares (WCSS) is an important concept in the context of

clustering algorithms, and evaluates how well data points fit within their assigned clusters, providing a way to quantify the compactness of clusters—the lower the WCSS, the more compact the clusters are, and thus the better the clustering. The mathematical formula is $WCSS = \sum_{i=1}^k \sum_{x \in C_i} \|x - \mu_i\|^2$, where k is the number of clusters, C_i is the set of points of cluster i , μ_i is the centroid of cluster i , and $\|x - \mu_i\|^2$ is the squared Euclidean distance between point x and centroid μ_i .

4.4 Deep Learning

DL is a particular case of ANNs where Deep Neural Networks (DNNs) are used. It is used for complex tasks such as object recognition, speech recognition, autonomous vehicles, and has proven successful in multiple fields such as computer vision, speech recognition and translation [101]. However, there are still challenges associated with these technologies, including data privacy, ethical considerations, and the need for transparency in the decision-making process, in order to meet the need for a more explainable AI.

4.4.1 Architectures

DNNs consists of several layers of nodes. Different architectures have been developed to solve problems depending on the specific domain or use-case. While CNN is used most of the time in computer vision and image recognition, RNN is commonly used in time series problems/forecasting. DNN, CNN, RNN, and Graph Neural Network (GNN) represent four common architectures that play a pivotal role in processing different types of data, such as structured grids (in images) or more complex structures like graphs [102].

Deep Neural Network

DNNs are characterized by the interconnection of every neuron in one layer to every neuron in the subsequent layer, and they consist of three main components: an input layer that receives the data, one or more hidden layers that process the data through activation functions (e.g., ReLU, sigmoid), and an output layer that generates predictions.

During the forward pass, the input data is propagated through the network. Each neuron's output is calculated as a weighted sum of inputs plus a bias, followed by an activation function to introduce non-linearity. The backpropagation algorithm trains the network by minimizing the error between predicted and actual outputs. This involves a forward pass to compute outputs and a backward pass to calculate gradients, updating weights using optimization techniques like stochastic gradient descent.

DNNs are capable of approximating any continuous function, making them versatile. However, they are prone to overfitting, especially with complex architectures or small datasets. Therefore, careful design and regularization are essential for effective training.

Convolutional Neural Network

Convolutional Neural Networks (CNNs) have emerged as a fundamental tool in computer vision, speech recognition and drug discovery tasks. Specifically tailored to handle datasets with inherent spatial relationships, such as images, CNNs excel in extracting intricate patterns and features from such data.

The architecture of a CNN is structured in a series of layers of nodes, each serving a specific purpose in the hierarchical feature learning process dictated by the task at hand. The foundation of CNNs lies in their inherent grid-like structure, which reflects the spatial arrangement of data and the local relationships between neighboring features. This structure can manifest in various dimensions: one-dimensional for time-series data, two-dimensional for images, three-dimensional for videos, or even in higher dimensions.

A crucial component of a CNN is the convolutional layer (a single layer within the network), a layer that applies local transformations across the entire input by utilizing a sliding window of weights known as a kernel or filter. These filters are typically much smaller than the input data and facilitate the computation of local weighted sums. Unlike traditional DNNs where each input feature has a unique weight, CNNs apply the same set of weights (filters) across the entire input, effectively reducing the total number of parameters that need to be learned. This approach enhances efficiency, particularly when applied to suitable datasets. When designing a convolutional layer, the key considerations revolve around determining the number of filters and their respective sizes, which significantly impact the performance of the network and its ability to extract meaningful features from the input data [86]. In the context of object recognition within images, the CNN takes a 2D $n \times n (\times \text{channel number})$ pixelated image as an input (Figure 14).

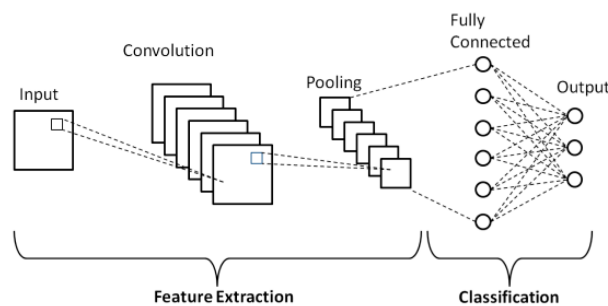


Figure 14: Schematic diagram of a basic CNN architecture [103].

The layers consists of groups of 2D neurons (the filters/kernels). The neurons in each feature ex-

traction layer are only connected to a fixed-sized and partially overlapping region of the input, known as the local receptive field. This reduces overfitting and training time. Neurons in a filter share weights and biases, speeding up learning and reducing memory requirements. Pooling layers are components commonly found in these networks that effectively downsample the feature maps generated by the previous convolutional layers. By reducing the spatial dimensions (width and height) of the input volume, operations such as pooling contribute to a degree of translation invariance by retaining essential features while discarding precise positional information. These layers, along with shared weights and local receptive fields, make the network less susceptible to shifts and distortions in images. Max, mean pooling, or local averaging filters achieve the sub-sampling. Finally, the final layers perform classifications, with neurons fully connected between layers. Deep CNNs consist of multiple weight-sharing convolution layers and sub-sampling layers, resulting in high-quality representations while maintaining locality and reducing parameters [102]. Activation functions are mathematical equations that determine the output of a neural network node based on its input. They introduce non-linearity, enabling the network to learn complex patterns. Common types include sigmoid, ReLU, and tanh, each with unique characteristics influencing learning speed and accuracy in various applications.

Recurrent Neural Network

RNNs are designed explicitly for sequential data structures such as time-series data, event sequences, and natural language. Their architecture offers internal feedback loops and therefore enables sequential pattern learning to model time dependencies by forming a memory. In RNNs, each input sequence is divided into time-steps, and a single layer of the network processes these inputs sequentially, one time-step at a time. A recurrent loop within the RNN carries the output from previous steps into the current step, enabling the network to incorporate information from past steps. This is achieved through a weighted sum of the features from each time-step and the output of the previous step. These weights are parameterized into input weights and recurrent weights, allowing the network to learn and adjust based on the input sequence. Additionally, RNN layers can consist of multiple units, each receiving input from the current step and outputs from all units in the previous step, forming dense recurrent connections between steps.

However, as the number of time-steps increases, RNNs encounter challenges such as the vanishing gradients during backpropagation, resulting in little or no influence of early memories. This phenomenon hampers the training process and makes learning long-term dependencies difficult. More sophisticated architectures, such as LSTM networks with advanced attention mechanisms, or specialized units like GRU attend to this problem [104], incorporating gates, which are learnable parameters controlling the flow of

information throughout the steps [84].

Long Short-Term Memory

LSTM is a specialized implementation of RNN. Unlike traditional feed-forward network architectures, LSTM addresses the issue of vanishing gradients in RNNs, allowing gradients to pass unaltered. This capability enables them to retain knowledge of earlier states, making it suitable for tasks that require memory or state awareness. Long Short-Term Memories (LSTMs) are blocks of memory cell states through which signals flow, regulated by input, forget, and output gates. These gates control what information is stored, read, and written in the cell, respectively. The LSTM network is composed of three main gates: the input gate, the forget gate, and the output gate. These gates are controlled by sigmoid and hyperbolic tangent functions and involve element-wise summations and multiplications. The input gate determines which new data is added to the cell state, the forget gate controls what information is retained in the cell state, and the output gate determines the output of the LSTM block [102].

The operations of the gates are learned and depend on the inputs, making the network adaptable to different tasks. At each time step, an LSTM receives three inputs: the features of the input at the time step, the output of all units from the previous step, and its own state from the previous step, as shown in Figure 15.

LSTMs are particularly useful for learning from sequence input data and building models that rely on context and earlier states. The cell block retains pertinent information from previous states, allowing the network to keep track of long-term dependencies.

Gated Recurrent Units

GRU are a smaller variation of LSTM, they are smaller in size when compared to them, as they do not include the output gate. While LSTMs are suitable for tasks requiring long-term dependencies, GRU can perform better on simpler datasets [102]. Thus, GRU do not have separate memory cells, but instead, they utilize a single hidden state vector to maintain and update information across time steps. This hidden state vector is updated using two main gates. The update gate controls how much of the past information should be preserved and how much of the new information should be incorporated into the current hidden state. The reset gate determines how much of the past information should be forgotten or reset. These gates are also controlled by sigmoid activation functions, allowing them to selectively filter and update information based on the input at each time step.

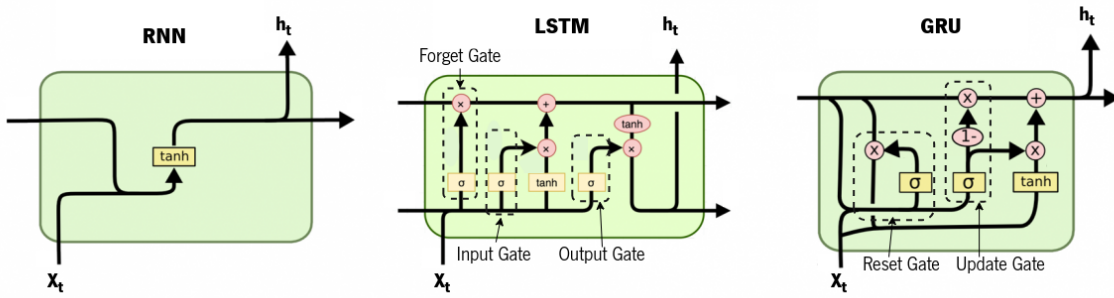


Figure 15: RNN/LSTM/GRU blocks, respectively, with memory cell and respective gates.

Graph Neural Network

Graph Neural Networks (GNNs) operate directly on graph-structured data, extending the capabilities of traditional neural networks to capture complex relationships and dependencies between entities. Unlike conventional neural networks that process structured inputs such as grids (e.g., pixels in images) or sequences (e.g., words in text), GNNs are designed to handle non-Euclidean data, where the underlying structure is represented as nodes and edges in a graph, making them particularly suited for applications where the data is inherently relational, such as social networks, molecular structures, and knowledge graphs.

A central feature of GNNs is their ability to iteratively update node representations by aggregating information from neighboring nodes, capturing both local and global graph structures. This neighborhood aggregation (or message-passing) allows the model to learn embeddings that reflect both the attributes of individual nodes and the topology of the graph. The general framework for GNNs was formalized by [105] with the introduction of Graph Convolutional Networks (GCNs), which extended convolution operations to graph domains by leveraging the graph Laplacian to define neighborhood aggregation. Key types of GNNs include GCNs, Graph Attention Networks (GATs) and GraphSAGE.

GCNs generalize the concept of convolutions from grids to graphs. They use the graph adjacency matrix and node features to compute new node representations by iteratively aggregating feature information from neighboring nodes [105].

GATs utilize attention mechanisms to adaptively weigh the importance of a node's neighbors during the aggregation step. This attention mechanism allows the network to focus more on relevant neighbors, providing flexibility in handling heterogeneous graphs [106].

GraphSAGE addresses scalability by sampling a fixed number of neighbors for each node during aggregation. This enables the model to be applied to large-scale graphs, making it suitable for inductive

learning where the graph structure might change dynamically [107].

4.5 Software, packages and tools

There are currently various software packages and libraries implementing relevant tools frequently used in ML and in DL. These libraries help with the reproducibility of published work and also ease the development of new approaches. Some examples that are also used further for the case-study are:

- **Scikit-Learn [108]:** ML module for Python, implements a broad toolkit for developing ML models, including tools for data preprocessing, model selection, and evaluation, making it a comprehensive library for traditional ML tasks;
- **TensorFlow [109]:** End-to-end open source symbolic math library available in Python, implementing many useful operations on tensors. Provides the most popular back end for DL computation. TensorFlow supports both high-level and low-level API access, which allows for flexibility in designing and training complex neural networks;
- **PyTorch [110]:** DL module for Python, implementing various popular DL architectures. Allows the easy development of custom DL architectures. PyTorch is known for its dynamic computational graph, which makes it intuitive to debug and modify neural network architectures on the fly;
- **Keras [111]:** DL module for Python, providing a high level abstraction of DL. Its focus is on allowing fast experimentation. Keras acts as an interface for TensorFlow, making it easier to develop and test neural network models quickly with a user-friendly API;
- **AutoGluon [112]:** An open-source AutoML toolkit designed to automatically produce high-quality ML models with little effort from the user. AutoGluon simplifies the process of model selection, hyperparameter tuning, and ensembling;
- **Hyperopt [113]:** A Python library for serial and parallel optimization over hyperparameters. It allows the use of Bayesian optimization algorithms to search for the best set of hyperparameters, making it an effective tool for model tuning. It also supports various optimization algorithms like random search, tree-structured Parzen estimators, and Gaussian processes;
- **Optuna [114]:** An open-source hyperparameter optimization framework designed to automate the process of hyperparameter tuning in ML, employing advanced optimizing algorithms. It supports pruning, visualization tools, and various storage options.

Chapter 5

Machine Learning for Toxicity Prediction

5.1 In silico Modeling Methods

According to [115], the term *In silico* toxicology is the application of computer technologies to analyze existing data, model, and predict the toxicological activity of a substance. Therefore, it refers more explicitly, to a computational experiment (relying on mathematical calculation, determined algorithms and scientific/biological analysis of xenobiotics), along with an organization of substance-related data, designed to predict toxicity for a specific endpoint, or for biological safety and testing purposes. The author emphasizes that the term is more comprehensive than the well-known computer-based analysis of the relationship of chemical structure and toxicological activity (SAR).

This type of toxicity assessment aims to complement existing toxicity tests, while minimizing *in vivo* and *in vitro* testing dependency, financial and time-related resources, and late-stage failures in drugs design [116].

Furthermore, *in silico* toxicology encompasses a wide variety of computational tools, which are aligned with its own definition, starting with the gathering of substance-related data in databases, and ending with the generation of molecular descriptors used to feed the several models, along with the application of modeling software and visualization tools, as Figure 16 presents.

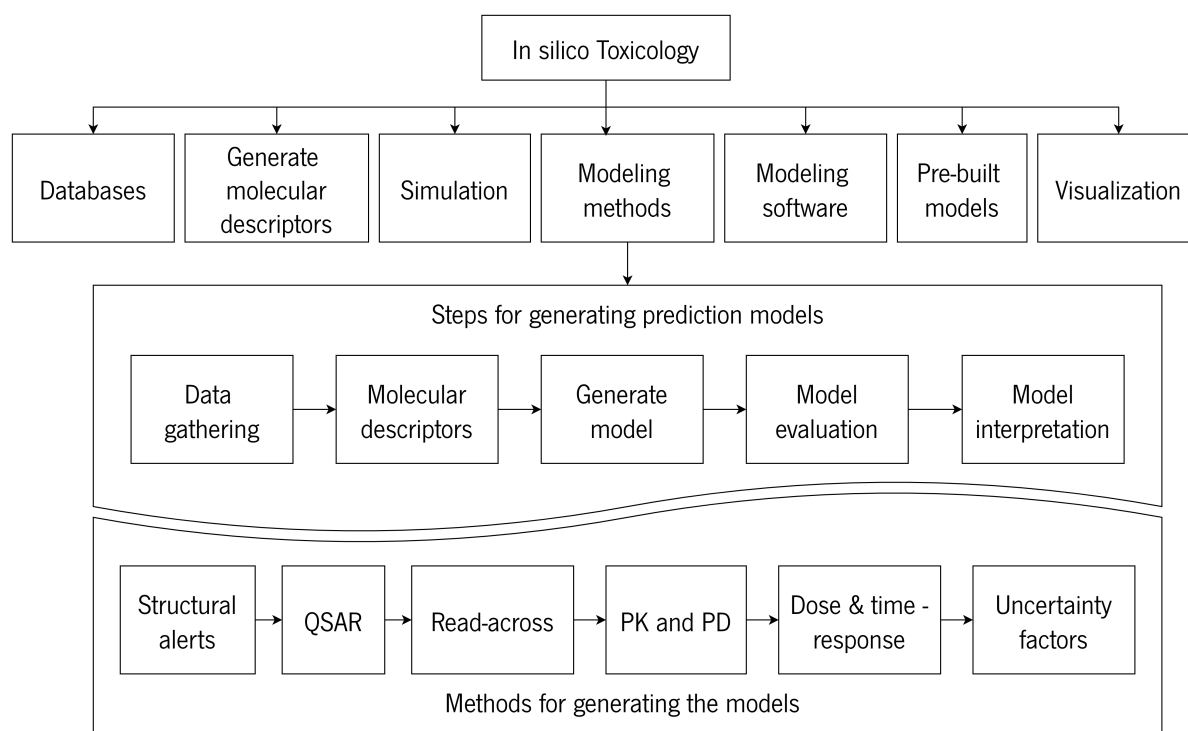


Figure 16: Conceptual tree of *in silico* technologies, illustrating the steps to generate prediction models, and prediction categories (adapted from [116]).

Quantitative Structure Activity Relationship (QSAR) is a computational modeling approach that can be employed to predict the toxicity of chemicals based on their molecular structure. It assumes that chemicals with similar structural features exhibit similar biological activities. The basic QSAR model predicts toxicity (T) using a set of molecular descriptors (Θ_P) and a function (f) that relates these descriptors to toxicity levels ($T = f(\Theta_P)$) [117].

There are two main categories of QSAR models: local and global. Local QSAR models are built from chemically similar compounds, providing tailored predictions for specific chemicals, hence offering higher accuracy. Conversely, global QSAR models are developed from diverse chemical datasets, making them more practical but potentially less accurate [118].

QSTR (Quantitative Structure-Toxicity Relationship) and QSPR (Quantitative Structure-Property Relationship) are both subsets of QSAR that use molecular structures to predict properties. However, QSTR specifically focuses on toxicity prediction, while QSPR encompasses a broader range of chemical properties, such as solubility, boiling point, or reactivity [118].

The development of a QSAR model involves two primary steps: generating molecular descriptors and constructing models to fit the data. Various molecular descriptors are utilized to characterize chemicals, including methods like simulated annealing, genetic algorithms, or PCA for feature selection [119]. These

descriptors are crucial in understanding the relationship between chemical structure and toxicity.

QSAR models can be generated using different algorithms, such as Logistic Regression (LR), partial least squares, ANNs, or SVM. While linear models are simpler and require fewer parameters, nonlinear models are commonly preferred due to the complex, often nonlinear relationships between chemicals and toxicity [118].

Visualizing structure-activity relationships is facilitated through Structure Activity Relationship (SAR) landscapes and SAR maps. SAR landscapes provide three-dimensional plots showing the relationship between molecular descriptors and response, while SAR maps offer a two-dimensional representation characterizing activity and structure similarity [118].

Historically, QSAR models date back to 1962, with [119] proposing an early model based on chemical concentration estimation using molecular descriptors. Since then, QSAR models have found application in various domains, however, their applicability is limited by the need for large datasets, challenges in feature selection, and limitations in biological interpretation.

QSAR models are widely used in various fields, including medicinal chemistry, environmental science, and toxicology. They serve as predictive tools for assessing the biological activity and toxicity of chemical compounds, aiding in drug discovery by identifying potential candidates with desired properties. These models assist in risk assessment by predicting the environmental impact of chemicals, thus informing regulatory decisions. These models can also be employed in virtual screening processes to prioritize compounds for experimental testing, streamlining research and development efforts.

These models have the advantages of being understandable, versatile and powerful: **understandable**, as they are easy to interpret if the descriptors are meaningful; **versatile**, as they can model categorical and continuous toxicity endpoints, and toxic and non-toxic chemicals; and **powerful**, as they allow to model complex endpoints, when using different types of descriptors [116].

However, these types of models also have disadvantages: they are dependent and limited: **dependency** comes from the fact that they require large datasets, as well as feature selection; and **limitation** in terms of not considering other important parameters that might take major importance when accessing toxicity, like dose and duration, and cannot be used for extrapolation between species, routes of exposure or doses, unless biological data are used [116].

5.2 Existent approaches for endpoint prediction

In the literature, different ML models are presented for toxicity prediction, depending on the type of endpoint to be predicted and using different methods. The most common pipeline for these types of models include four phases: definition of toxic endpoints, data selection and curation, ML construction and evaluation and interpretation of results.

5.2.1 Endpoint-specific

Cardiotoxicity

[120] employed DL based on multi-task DNNs to predict cardiotoxicity, with a particular focus on hERG channel inhibition, which is critical due to its potential to cause fatal arrhythmias. The dataset utilized comprised nearly 7889 compounds sourced from the PubChem and ChEMBL bioactivity databases, as well as information from the literature. The authors considered only those compounds with well-defined experimental hERG blocking bioactivities, excluding compounds with a half-maximal inhibitory concentration (IC_{50}) value exceeding 10 μM . The data were then segmented based on various decoy threshold values, ranging from 10 μM to 100 μM , with each range designated as a specific task. For each task, the data were further divided into training, validation, and final evaluation sets. Featurization involved the use of Molecular Operating Environment (MOE) descriptors and vector representation of molecules, utilizing Mol2Vec. The depicted DNN architecture consisted of an input layer that receives the multiple chemical features, followed by three fully connected hidden layers, and multiple task-specific output layers to predict whether the compound is in or out of the IC_{50} specific range (i.e., task one: $IC_{50} > 10$ or $IC_{50} > 10$, task two: $IC_{50} > 20$ or $IC_{50} > 20$, and so on). Finally, the evaluation metric applied was ROC AUC, and the DNN models using multi-task outperform other models, such as singletask DNN, NB, SVM, RF, and the graph convolutional neural network. The performance of the best model (MT-DNN) on the validation set was measured at 0.967.

[121] combined pharmacophore modeling with ML, as they used NB and SVM algorithms integrated with multiple representative pharmacophore hypotheses identified by a recursive partitioning (RP) approach. RP allows for the identification of the most important pharmacophores creating a decision tree that splits molecules into subsets based on independent properties. The authors used a dataset of 587 molecules (from PubChem and ChEMBL bioactivity databases [122, 71, 70]) for model construction and validation, with 527 exhibiting experimental hERG blocking bioactivities (IC_{50}) and 60 identified as hERG

nonblockers from a previous study. A 40 μM threshold distinguished hERG blockers from nonblockers. This dataset was split into training, validation, and testing sets, ensuring maximum diversity in the training set, evaluated by Tanimoto coefficients based on molecular fingerprints and logP. The optimal SVM model achieved prediction accuracies of 84.7% for the training set and 82.1% for the test set.

[123] also developed a predictive model for hERG binding, but instead, they used a hybrid approach of LASSO-regularized SVM and data from the ChEMBL bioactivity database [71] (8,337 entries). Their method combines structure-based drug design (molecular fingerprints and docking scores) with ML techniques. Different IC_{50} inactivity thresholds from 1 μM to 80 μM were used (positive molecules: $\text{IC}_{50} \leq 1\mu\text{M}$ and negative molecules: those with $\text{IC}_{50} >$ different inactivity thresholds). Evaluation included a ROC AUC of 0.86 and negative predictive values of 0.81.

Acute oral toxicity

The Acute Oral Toxicity (AOT) is frequently evaluated using the LD_{50} , a quantitative dose endpoint (as previously mentioned in subsection 2.3.1). Numerous ML models have been proposed to predict AOT, using both regression and classification methods. One such model is etoxPred, introduced by [124]. The model trains several classifiers with varied architecture and using different strategies, employing two primary predictors: SAScore and Tox-score.

Firstly, the input data is 1024-bit Daylight fingerprints, constructed using Open Babel [77]. This data is used as reference for SAScore values, which are calculated through an exact approach that combines a fragment-based score (representing historical synthetic knowledge) with a complexity-based score (that penalizes the presence of multiple chemical structures). This data is given to a Deep Belief Network (DBN) with a specific architecture consisting of a visible layer corresponding to the fingerprints input and three hidden layers with 512, 128, and 32 nodes, respectively. To mitigate overfitting, L2 regularization and adaptive learning rate decay are applied, as well as 5-fold CV using a large set of known datasets.

Secondly, the second predictor, Tox-score, is a binary classifier based on the Extremely Randomized Trees (ET) algorithm where the final class assignment is determined by majority voting among the individual trees. The training and CV processes utilize the FDA-approved dataset as positive (non-toxic) instances and the TOXNET [29] dataset as negative (toxic) instances (non-toxic instances present $\text{LD}_{50} < 500$ mg/kg). The toxicity predictor is subsequently trained on the entire FDA-approved/TOXNET dataset and independently tested against the KEGG-Drug [125] (positive, non-toxic) and T3DB [126] (negative, toxic) datasets. Additionally, the capability of the classifier to predict specific toxicities is assessed using several specific datasets, with 5-fold CV.

Finally, both ML predictors, SAScore and Tox-score, are applied to the Traditional Chinese Medicines (TCM) dataset [127]. The etoxPred model was evaluated with an accuracy of 0.85 and an AUC value of 0.80, demonstrating better performance comparable to other multiclassification models such as MLP, RF, and LDA.

It is possible to use molecular graphs in conjunction with Convolutional Neural Networks (CNNs). In this approach, the basic chemical information of molecular graphs serves as input, allowing the DL algorithm to automatically learn specific representations using graph convolutions, thereby eliminating the need for descriptors [128].

In this context, [129] presented the deepAOT model, which is based on a Molecular Graph Encoding Convolutional Neural Network (MGE-CNN) architecture. This model builds on the previous work by [130]. The input data comprised approximately 12200 compounds sourced from their previous AOT (acute oral toxicity) database, which includes data from the admetSAR database, the MDL Toxicity Database (version 2004.1), and the Toxicity Estimation Software Tool (TEST version 4.1) program from the U.S. Environmental Protection Agency (EPA).

Two external test datasets were used to estimate the predictive power of the models. Both regression and multi-classification models were developed. The regression models enable the quantitative prediction of LD₅₀ values (where a lower value indicates higher toxicity). For the multi-classification models, data were divided into balanced classes based on the U.S. EPA's AOT categories. The workflow begins with the standardization of canonical SMILES strings, which are then used to construct a molecular structural graph. Each sub-graph from each iteration is encoded into a fixed-sized vector, summed to create a unique fingerprint, and used as input to the neural network's output layer.

In terms of evaluation, the regression model demonstrated superior performance, exhibiting a higher Pearson Correlation Coefficient (0.864), lower Root Mean Square Error (RMSE) of 0.268, and lower Mean Absolute Error (MAE) of 0.195 compared to most state-of-the-art models. The deepAOT classifier also showed excellent performance, achieving an accuracy of 96%, which surpasses the best-reported accuracy of 80% from previous AOT classification methods by [130]

[131] introduced topology-based descriptors utilized with multitask deep neural networks (MT-DNN) for the prediction of LC₅₀ and LD₅₀ values. This method employs Element-Specific Persistent Homology (ESPH) as a novel molecular representation technique for toxicity prediction. ESPH is an algebraic topology approach that preserves essential chemical information during the topological abstraction of geometric complexity.

Four benchmark toxicity datasets involving quantitative measurements were used to validate the pro-

posed approaches. These datasets included oral rat LD₅₀. The data were sourced and curated from the ECOTOX aquatic toxicity database [132] and the ChemIDplus database [133]. Additionally, the Toxicity Estimation Software Tool (TEST) database was employed as a final test set to compare the results with previous studies. Supported by physicochemical descriptors and the MT-DNN architecture, the ESPH-based descriptors (ESTDs) demonstrated performance comparable to state-of-the-art predictions for quantitative toxicity datasets, achieving an R² value of 0.788. The study highlights that new molecular representation methods, such as ESTDs, merit further investigation to enhance the performance of ML models in toxicity prediction.

Hepatotoxicity

As specified in previously in subsection 2.3.2, Drug induced Liver Injury (DILI) is the toxicological endpoint of concern when predicting this type of toxicity, as it has been the leading cause of clinical trials failure and drug withdrawal from the market. [134] presented a DL model to predict DILI, using data from Liver Toxicity Knowledge Base (LTKB) [135] and other additional datasets. The authors developed a graph RNN, encoding molecules SMILES notation as undirected graphs that were then encoded to acyclic graphs for use in the network. In the graph, heavy atoms are represented as nodes and bonds between atoms are represented as edges. Several models were developed and each one required a separate training and validation set. The evaluation metrics included sensitivity and specificity, and the model of best performance was trained on 475 compounds and evaluated on 198 drug-like compounds. The three trained models registered a sensitivity and specificity of 81.4 and 61.3, 80.5 and 70.3, 82.5 and 92.9, respectively. On the other hand, [136] used 2144 chemicals from FDA-approved drugs and human DILI data from LTKB [135], to develop binary classification models. and the methodology consisted in preprocessing, filtering and application of the SVM algorithm using 12 different descriptors like MACCS keys and 7 types of fingerprints like Klekota-Roth fingerprints. Compounds were randomly divided into training and validation sets with a 4:1 ratio.

Finally, the evaluation metrics included accuracy, sensitivity, and specificity, and the performance achieved was 80.4% of accuracy, 88.2% of sensitivity, and 65.7% of specificity, showing similar results in validation and external test sets.

Mutagenicity

This endpoint is another property highly responsible for drugs candidates failure in drug discovery context, and its prediction aims to identify compounds causing DNA damage, which is highly related to carcino-

genicity (subsection 2.3.2). The most know assay for testing this endpoint is the Ames test. The process of conducting it depends on a large number of resources, mainly monetary, and identifying the existence of toxicity is still inefficient. For this reason, computational models have attracted attention in research groups. Starting with [137], who used Ames data collection by [138]. Their methodology involved combining molecular descriptors and fingerprints in order to train eight classification algorithms, including SVM, Random Forest (RF) and Extreme Gradient Boosting (XGB). These models achieved best results when compared to the previous work. The training and validation sets clustered 7617 and 731 molecules, respectively, and a separate balanced external set of 234 molecules were also used in order to remove duplicates. During feature importance analysis, authors concluded that some known physicochemical descriptors presented high relevance in model training combined with molecular fingerprints. Evaluation metrics included ROC AUC, and the RF model achieved ROC AUC > 0.90 on training set and ROC AUC > 0.75 on external test set.

The most recent AMPred-CNN model by [139] demonstrates another assessment of mutagenicity using Ames test. Authors use the benchmark mutagenicity dataset collected by [140] with 6512 compounds. They start by curate the data employing standardization and using a clear evidence criterion to avoid conflicting compounds: is given priority to positive cases when the same canonical SMILES (repeated entries) has different mutagenic results. A set of 6277 compounds remain and a random split of 80% for training and validation, and 20% for testing is done. PCA and t-SNE analysis are done. In the context of featurization, authors used five different types of structural features, including MACCS, ECFP6, 208 2D descriptors and 1826 Mordred descriptors. Each image had dimensions of 224x224 pixels with 3 channels per RGB encoding. The architecture used consisted of a convolutional layer (16 filters, kernel size of (5,5) and stride of (1,1)), followed by a 2x2 MaxPooling layer, another convolutional layer (32 filters, same kernel size and stride), a dropout layer with 0.5 rate for regularization, a set of three fully connected layers with 256, 84, and 2 neurons (for the binary prediction), with a LogSoftmax along dimension 1 for probability distribution. The training employed a CrossEntropy loss function with a SGD optimization. Five ML algorithms (CatBoost (CB), XGB, ET, RF, and Light Gradient Boosting Machine (LGB)) were chosen for their best performance among the many ML algorithms PyCaret supports when we evaluated 10-fold cross validation and tested on test sets corresponding to the following features: ECFP6, MACCS, RDKit2D, and Mordred. Model evaluation was done using accuracy, ROC AUC, sensitivity, specificity, mcc and f1-score. The previous five models consistently outperformed the SVM model from [140], boasting an AUC greater than 0.86. When comparing to other several models in literature, the proposed models still outperformed, highlighting the reported auc of 0.95. This study shows the correct use of molecular prediction tasks and

its effectiveness in capturing intricate patterns within molecular structures, encouraging future exploration of image-based approaches in chemoinformatics.

Carcinogenicity

[141] constructed the CarcinoPred-EL tool consisting of three ensemble classification models capable of predicting the carcinogenicity of chemicals from seven types of molecular fingerprints. The algorithms used were SVM, RF, XGB ensembles. Authors used 1003 compounds as training set from the Carcinogenic Potency Database (CPDB), and 40 compounds as external validation set from ISSCAN database [142]. The ensemble models can be formed by combining simple independent classifiers via voting or averaging to produce a more accurate and robust model than any of its constituents. This strategy is particularly relevant when dealing with unbalanced or reduced data. The registered performance was any ensemble was higher than any binary classifier, and the ensemble XGB model achieved an AUC of 76.5% (the best result at the time).

5.2.2 General Toxicity

One of the most known used database for various toxicities predictions is Tox21 with data of nuclear receptor (NR) and stress response (SR) panels [17]. DeepTox [143] was the DL winner model of the respective challenge. The authors used thousands of physicochemical descriptor and molecular fingerprints to represent data (e.g. PCFP, ECFPs and MACCS keys), adding to it nearly 2500 toxicophore features (substructures previously reported as toxiphores). They have also included nearly 40 similarity features calculated using natural ligands of the receptors. The existent data imbalance was corrected by oversampling positive samples with PubChem and ChEMBL databases, searching for compounds and assays of the databases that were similar to the ones of the project data, using pharmacological analoging (Kernel-based Structural and Pharmacological Analoging (KSPA)). In DeepTox, for 10 out of 12 assays Multitask-DNN models outperformed singletask networks. An average ROC AUC of 0.86 was obtained for the NR test set and 0.83 for the SR test set. Some compounds were not tested across the entire target set, resulting in sparse arrays. To address this issue, compounds with missing activity values were assumed to be inactive, which could lead to false negatives. This challenge of missing data in multitarget matrices is an active area of study within ML.

Furthermore, [144] presented a model called TOP, also trained and evaluated on the Tox21 data. The model architecture included word embedding and a RNN based on a bidirectional GRU, followed by fully connected neural networks. Featurizers include a mixing of SMILES (with augmentation performed) and

physicochemical descriptors. The model performed better than DeepTox in 11 out of 12 tasks, with an average ROC AUC value of 0.95 (vs. 0.85 in DeepTox).

A simpler model is yet consider, proposed by [145]. The study utilizes benchmark toxicity datasets (e.g. Ames mutagenicity (AM) dataset, Tetrahymena pyriformis IGC50 dataset, oral rat LD₅₀ dataset, and LC₅₀ datasets for both fathead minnow and Daphnia magna), which are then employed to validate the performance of the proposed models. Chemical compounds are represented using 2D descriptors, which are computationally less intensive than 3D descriptors. A total of 1422 2D descriptors were calculated with PaDEL tool [146] and then filtered using a decision tree to select the most relevant features for model training. The training method involves a hybrid approach that integrates a shallow neural network (SNN) with a decision tree (DT). The SNN consists of one hidden layer with 10 neurons. Parameters for both the DT and SNN are jointly optimized. The optimization is performed in both series and parallel modes, with series optimization demonstrating marginally better performance. The SNN's role is to extract relevant features and train on a reduced feature set. The model is compared against traditional ML methods like K-Nearest Neighbors (KNN), SVM, Naive-Bayes (NB), and RF, as well as DNNs. The performance of the models is evaluated using metrics such as area under the curve (ROC AUC), precision-recall curves (AUC-PR), and F1 scores. The hybrid model achieved high accuracy, outperforming existing methods on several toxicity benchmark tasks. For instance, it obtained an ROC AUC of 0.95 for the SR-MMP task, which indicates strong predictive power using 2D features. The simplicity of the model allowed for better interpretability of the features and decisions, which is critical in cheminformatics.

[147] presents a novel method for predicting the toxicity of small molecules by leveraging a stacked ML model. Utilizing a diverse dataset from sources like FDA-approved drugs, KEGG database, and TOXNET, the model incorporates LGB, RF and Multilayer Perceptron (MLP) as base classifiers, with LR as the meta-classifier. Feature selection is rigorously performed using chi-square and mutual information statistics to enhance model accuracy. The stacked model achieved superior performance metrics, including an ROC AUC of 87.8% on the test set and 88.9% on external validation, outperforming individual base models and existing tools.

[148] is a recent study that implemented several sophisticated models to predict toxicity many toxicity endpoints, each designed to leverage different aspects of molecular representations. The study utilized multiple datasets for training and evaluation, including the Tox21, various MoleculeNet datasets [149] (BACE, SIDER, ClinTox, BBBP, HIV, and a redistribution of Tox21), and the Cytotoxicity dataset from the *Leibniz-Forschungsinstitut für Molekulare Pharmakologie*, which measures the cytotoxicity of 34848 compounds in HEK292 and HepG2 cell lines. The various endpoints predicted are: Inhibition of human

b-secretase 1 (BACE-1), associated with Alzheimer’s disease; adverse drug reactions; clinical trial toxicity status and FDA drug approval status; blood-brain barrier permeability, which can be related to neuro-toxicity; ability of compounds to block HIV replication, which is not directly a toxic endpoint but relevant for drug efficacy, and Cytotoxicity in HEK292 (kidney) and HepG2 (liver) cell lines, indicating potential organ-specific toxicity.

The primary models are presented in Table 10.

Table 10: Several types of models addressed by [148].

Model Type	Model Name	Model(s) Description
String-based	Chemical Language Model (CLM)	Attention-based multiscale convolutional neural network inspired by prior work in drug sensitivity prediction. It processes SMILES strings through a series of embedding layers, followed by three parallel 1D convolutional layers with kernel sizes of 3, 5, and 11, along with a residual connection. The model uses multiple attention heads to focus on relevant molecular substructures, and the outputs from these layers are concatenated and processed through dense layers to generate the final prediction.
	RNNs	Two models employed. First, a GRU with bidirectional layers with 256 units, followed by a dense network with three layers (1024, 1024, and 512 units) and a 50% dropout rate. Second, a Neuromodulated Bistable Recurrent Cell (nBRC): a biologically inspired modification of the GRU capable of exact memorization and counting, crucial for handling SMILES sequences with ring opening and closure symbols. It employs the same bidirectional and dense layer structure as the GRU
Fingerprint-based	ML-based	Two models employed: RF that uses ECFPs as input features. Hyperparameters such as the number of trees were optimized, with 500 trees found to be optimal. The KNN uses ECFP fingerprints and employs the Tanimoto similarity measure as the distance function. The parameter k was set to 23 based on performance on the Tox21 dataset.
	DL-based	A simple, four-layered, fully connected neural network with 512, 1024, 2048, and 1024 units with a sigmoid activation function.
Graph-based	Graph Convolutional Network (GCN)	Model that represents molecules as graphs, with atoms as nodes and bonds as edges. The architecture includes two graph-convolutional layers with 64 units each and one dense layer with 128 units, using a sigmoid activation function for the final output.

The performance of the models was evaluated using ROC AUC, balanced accuracy, sensitivity and

specificity. Stratified 5-fold CV was used to ensure balanced class distribution. First, the authors examine the impact of using different molecular representations on Tox21 and then they pick the best model and extend the analysis to MoleculeNet data. The graph kernels demonstrate worse performance given the large amount of data used. However, the GCN yielded second best performance with a ROC AUC of 0.83. Still, their proposed CLM model with augmented SMILES representation achieved the highest ROC AUC, demonstrating superior performance across all models and datasets.

Chapter 6

Methods

6.1 Methodology

For the execution of this research project, a methodology based on a case study analysis was employed. The case study method is a research approach that utilizes various elements for the classification, planning, data collection, and data analysis, with the aim of evaluating, understanding, or describing dynamic situations or events in which human involvement is present [150]. In this particular case, public real-world data was used, specifically from the Chemical Hazards Database (OpenFoodTox) provided by EFSA.

Each time a new food ingredient is created, it becomes necessary to test it in order to determine whether the product is viable and non-toxic for potential users. This process involves several entities and specific procedures, often being time-consuming, financially costly, and reliant on *in vivo* testing. This case study proves useful in addressing the problem of resource dependency, making it possible to predict whether a food ingredient is toxic or not solely by using AI and chemical information from previously categorized trials. For this study, only computational data of compounds were considered, specifically their SMILES notation.

As such, the possibility of integrating this dataset into the OmniumAI database was explored, with the goal of making the toxicity information available for future analysis and reference. Once the data was gathered, the next step involved developing a predictive model based on AI. The models, after being developed, were tested using performance metrics appropriate to the nature of the problem and the data. The development of the predictive work, including data analysis, model construction, and data integration, was carried out using Python, the fastest-growing programming language in the world due to its ease of use, rapid learning curve, and numerous high-quality packages for data science and ML [151]. The application itself was developed using Python and JavaScript.

In order to meet the proposed objectives, and following the study of various methodologies, the decision was made to adapt the principles of Data Mining (DM) methodologies to this case study. The

methodologies that served as the foundation for the approach used in this work were: Cross Industry Standard Process for Data Mining (CRISP-DM), consisting of six phases—business understanding, data understanding, data preparation, modeling, evaluation, and deployment; Sample, Explore, Modify, Model, and Assess (SEMMA), divided into five phases—sampling, exploration, modification, modeling, and evaluation; and Catalyst or Product, Price, Place, Time, Quantity (P3TQ), formulated based on two models—the Business Model and the Data Exploration Model. Furthermore, a specific architecture was considered, as the project was developed in an environment that included *Databricks* and its inherent data organization structure.

Thus, in the case study developed for this research project, the following phases were considered:

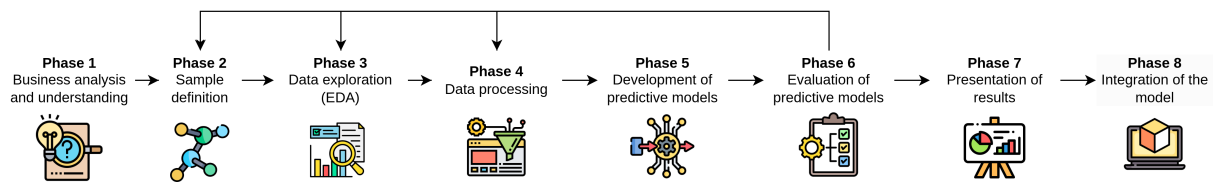


Figure 17: Phases of the adopted methodology.

Accordingly to Figure 17, **(1)** First, a business analysis is conducted to understand the problem context and define objectives. **(2)** Then, the sample is defined by selecting relevant datasets that accurately represent the population. **(3)** Next, data exploration is performed using Exploratory Data Analysis (EDA) techniques to identify trends and relationships within the dataset. **(4)** After that, the data is cleaned and preprocessed, addressing missing values and normalizing features to prepare for modeling. **(5)** Following this, predictive models are developed by selecting appropriate algorithms and tuning hyperparameters for optimal performance. **(6)** Then, the models are evaluated using appropriate metrics to assess their predictive capabilities. **(7)** Once evaluated, results are presented through visualizations and analyses to inform stakeholders. Finally, **(8)** the top best-performing models are integrated into a production environment (in the developed application), ensuring usability for real-world applications and providing ongoing support for monitoring and updates. It is important to note that the development of the app was done in parallel with the ML tasks.

6.2 Toxicity of compounds in OpenFoodTox dataset

OpenFoodTox [32] is the EFSA chemical hazards structured database, constituted by 9 sheets that contain a variety of toxicological information. As observed by Figure 18, at the core of the dataset (5 sheets) is the **Components** Sheet (8k rows), which catalogs chemical information for each compound, including relevant data like SMILES, CAS numbers, International Union of Pure and Applied Chemistry (IUPAC) names, and molecular formulas. This sheet serves as the foundation, linking each chemical to its toxicological data and establishing reference points for safe dosage. The **Genotoxic** Sheet (222 rows) focuses on genotoxicity assays, which reveal whether substances have the potential to cause genetic damage or mutations, it captures essential attributes like the study category, compound IDs, and guideline qualifiers. Further insights are provided in the **EndpointStudy** Sheet (11.4k rows), which records critical toxicological endpoints observed in various studies, covering adverse effects on specific organs and systems, and dose information. The **Study** Sheet (50k rows) aggregates data from multiple assay types, offering a comprehensive view of a compound's toxicological profile. This includes results for genotoxic, carcinogenic, and mutagenic effects. Finally, the **Opinion** Sheet (25.6k rows) provides EFSA's scientific opinions, offering expert evaluations and statements based on the dataset's findings. This sheet consolidates regulatory information, giving context to experimental results.

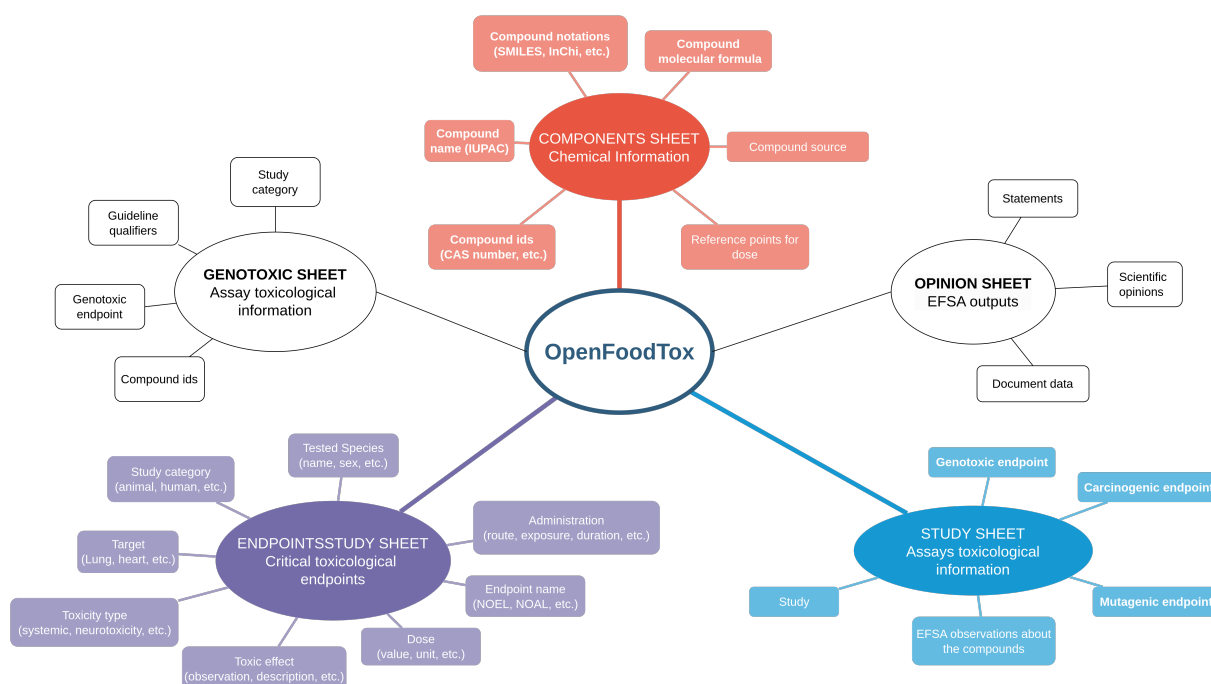


Figure 18: Overview of the OpenFoodTox data. Relevant sheets are colored, important features are in bold.

6.2.1 Data Collection and Processing using Medallion Architecture

The first phase of the work was to research the different databases (see section 2.4) with data related to toxicity, with a study focused on the type of data available, the type and number of toxicity endpoints and the type of license of each one, as Table 3 suggests. After the analysis of each dataset available, it was given more relevancy to food-related data, as it was the industry of interest that better aligned with the main purposes of the company and the respective intern project. This dataset opened opportunity for improvement in the field because it is recent data and with few referenced works.

The collection and processing of data were conducted following the medallion architecture data pattern within a lakehouse environment. This architecture, developed by [Databricks](#), aims to progressively enhance the structure and quality of data as it traverses through each layer of the architecture, beginning from the bronze stage, progressing to the silver stage, and culminating at the gold stage. Each layer has a distinct purpose and specific operations that can be performed [152].

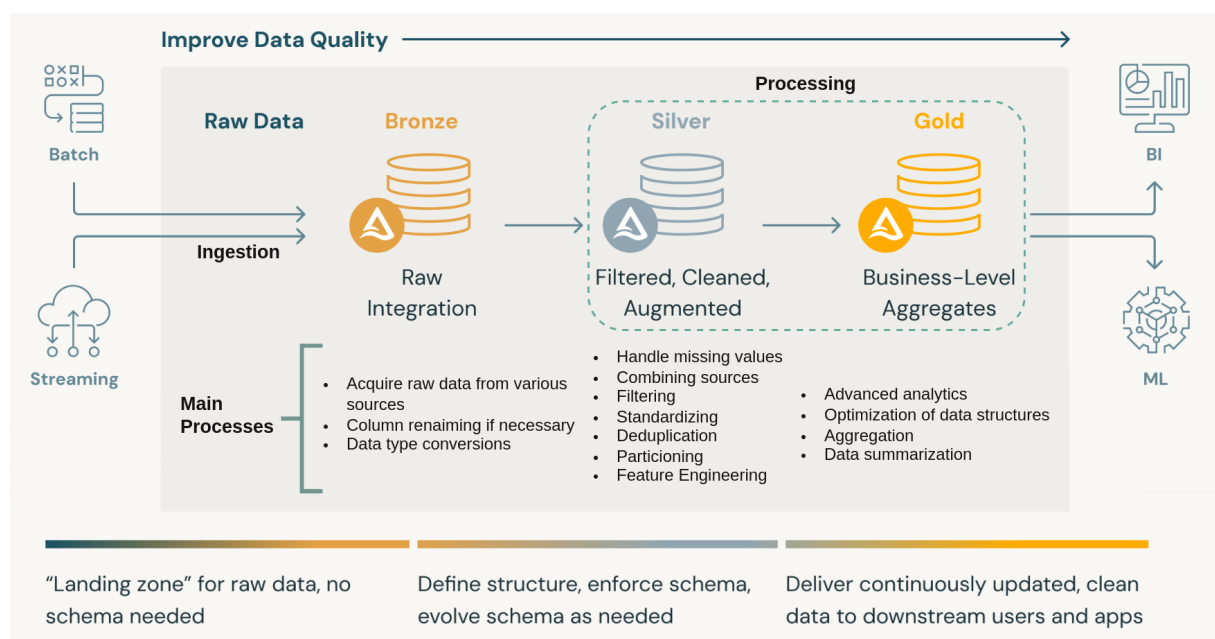


Figure 19: Stages of medallion architecture and the main processes of each phase (adapted from [153]).

As illustrated in Figure 19, the ingestion process leads to the bronze stage, commonly referred to as the "landing zone" where raw data is stored without any schema requirements. During the processing stages (silver and gold), the data structure is defined, schemes are applied, and data quality is continuously improved. The bronze stage serves as the initial point for data ingestion and storage, with data preserved in its raw form without processing or transformation. In the silver layer, data is cleaned, filtered, and transformed into a more usable format. Transformations at this stage include light modifications but

do not involve aggregations or enrichments. Finally, in the gold stage, data is refined to meet specific business and analytical requirements. This may involve aggregating data to a certain granularity, such as daily or hourly, or enriching data with information from external sources. After the gold stage, data should be prepared for consumption by downstream teams, including analytics, data science, or ML operations [152].

Bronze stage

This stage included fetch and initial raw loading of the data to Databricks Delta Lake. Delta tables are a type of table in Databricks Delta Lake, designed for large-scale data processing with support for Atomicity, Consistency, Isolation, Durability (ACID) transactions, version control, and schema evolution. They enable efficient data storage and management, allowing both batch and streaming data to be processed with high reliability and speed. After the loading of the dataset, some columns were renamed to successfully store data into the delta tables. Each sheet of the document was loaded into a separate delta table. Auxiliary tables with metadata were also created only for statics and practical matters.

Silver stage

This stage included merging and processing the data. The dataset contained 9 sheets, however, only the information about each compound (sheet COMPONENT) and endpoint study details (sheet STUDY) were relevant and therefore used. An inner join on the ID of each compound was performed, between the two sheets, so the final dataset aggregated only the rows that have both the chemical and study information. This merged data was used for a preliminary visualization and exploration of the data.

The next phase was to get a dataset without duplicates. The duplicates of the merged data, more precisely, chemicals with the same ID, same SMILES notation, and that had the same label, were removed. Having the same chemicals with the same label does not contribute to model knowledge as they only introduce noise to future training. It was observed while investigating the duplicates, that few chemicals, despite having a different ID (meaning that they were marked and were a different compound), they had the same SMILES notation and even the same label, so they were removed, as well as the duplicates with the same SMILES, different ID, and different label.

The next step was to proceed to handle the existent ambiguous labels. On the current dataset, each compound could have more than one entry because the STUDY sheet could contain more than one study/assay per chemical, therefore, resulting in duplicated lines in the final dataset, but with different labels (see Table 12). The method for treatment was to count the number of positive and negative labels

and keep the most common to that entry. If the number of positives equals the number of negatives, or if this number is 0 (there are only labels with no relevant information), the entry is marked with “No data”.

Next, the method explained in more detail:

1. Count Positive and Negative values for each compound for each label;
2. Determine the most common value for each compound;
3. Replace all the entries of that compound with the most common value for the label. In cases where the amount is equal, or even non-existent a No data label was put.

These steps resulted in a dataset with unique compounds and no duplicates. This data was loaded into the silver schema.

Gold stage

This final stage prepares the dataset to be ready for the ML training, noticing that labels are handled pre-supposing a multitask/singletask binary classification problem. This phase involved using the treated data to fetch the ones with missing SMILES, using PubChem PUG rest API [154] and National Cancer Institute (NCI) Chemical Identifier Resolver API [155]. The process consisted in sending a request for each API per compound, using the identifiers available, in a specific order: CAS number, InChi, IUPAC name, molecular formula, and compound name. The remaining compounds without SMILES were dropped, and data was numerically encoded (positive labels to 1 and negative to 0, “No data” cases were imputed with a NaN). Cases where all labels were missing were removed. The final processed dataset had about 3100 rows, and 4 columns (SMILES (only feature), IS_MUTAGENIC, IS_CARCIINOGENIC, and IS_GENOTOXIC (labels)). This dataset was used for the multitask training. For the singletask, this dataset was still split (for SL purposes) in one dataset per label, with about 2200 rows per task.

6.2.2 Preliminary Data Visualization and Exploration

Before even proceeding to the proper processing where the majority of modifications occurred, data was analyzed, to get a more comprehensible idea of the information that the dataset could provide, and then modify it accordingly. Considering also the initial merged dataframe present in Table 11, several relevant information was observed.

Table 11: Some rows and columns of the initial merged dataframe (silver stage data).

COMPARAMNAME	COM_CASNUMBER	IUPACNAME	SMILESNOTATION	IS_MUTAGENIC	IS_GENOTOXIC	IS_CARCIINOGENIC
Vanadate	NaN	NaN	NaN	Positive	Positive	Not determined
Vanadate	NaN	NaN	NaN	Positive	Positive	Not determined
Vanadate	NaN	NaN	NaN	Positive	Positive	Not determined
Sodium vanadate	13178-26-8	sodium oxido(dioxo)vanadium	NaN	Positive	Positive	Not determined
Sodium vanadate	13178-26-8	sodium oxido(dioxo)vanadium	NaN	Positive	Positive	Not determined

- The initial number of rows of the dataframe (bronze stage) was near to 50k, indicating that more than 50k studies were present. However, as visible, one compound could have more than one entry (e.g. Vanadate had 3 studies with the same label values), and some compounds present the same chemical properties and labels but have a different SMILES notation;
- There are compounds with missing information in the SMILES column, crucial for the future training process, that must be imputed using other present chemical information;
- As Table 12 shows, the same compound $C_6H_8O_6$ can have different studies and each one with a different value for each label, inevitably forcing the application of a method that brings coherence and defines the final value. Two compounds can have the same molecular formula and still be a different compound;
- There are label values that cannot be used as they are ambiguous and do not allow to determine the presence or absence of toxicity. The dataset did not contain any information about the meaning of the different labels. There was no more than one available article to describe their meaning, therefore, one can only make an assumption of it. The label `No data` infers that no data was available to determine toxicity, and `Ambiguous` means that the study presented both positive and negative values for toxicity of the compound. `Not determined` induces us that the investigated endpoint toxicity cannot be determined for some reason (insufficient data or no conclusion allowed), `Not Applicable` remain in doubt, and `Other` is refereed by [156], indicating waived endpoint values and other justifications. Thus, all of these labels could not be used for a correct toxicity assessment.

Table 12: Presence of several studies for the same compound.

SUB_COM_ID	MOLECULARFORMULA	SMILESNOTATION	IS_MUTAGENIC	IS_GENOTOXIC	IS_CARCIINOGENIC
36	C6H8O6	C([C@@H]([C@@H]1C(=C(C(=O)O)O)O)O)O	Ambiguous	Ambiguous	Ambiguous
37	C6H8O6	C([C@@H]([C@@H]1C(=C(C(=O)O)O)O)O)O	Ambiguous	Ambiguous	No data
51	C6H8O6	C([C@@H]([C@@H]1C(=C(C(=O)O)O)O)O)O	Negative	Negative	Negative
52	C6H8O6	C([C@@H]([C@@H]1C(=C(C(=O)O)O)O)O)O	No data	No data	No data
56	C6H8O6	C([C@@H]([C@@H]1C(=C(C(=O)O)O)O)O)O	No data	No data	Not determined

When creating a heatmap with the initial merged data (10312 entries, after removing compounds with the same ID and labels) - only for visualization purposes of the missing data, this was not the final dataset used-, it was observed that there were many compounds without SMILES information (1538), Molecular formula (1635), IUPAC Name (1685), CAS Number (1472), and other chemical/toxic properties. Determining the number of missing data for each property was essential in order to have a comprehensive idea of which compounds could be used and, in the worst case, how many compounds would be viable considering that the missing information would be inputted through the feature with fewer or no missing cases.

Table 13: Number of missing values for each column.

Column	Number of missing values
COMPARAMNAME	0
COM_CASNUMBER	1472
IUPACNAME	1685
MOLECULARFORMULA	1635
SMILESNOTATION	1538
INCHI	1477
GENOTOX_ID	10145
TOX_ID	7414
IS_MUTAGENIC	0
IS_GENOTOXIC	0
IS_CARCIINOGENIC	0

Analyzing Table 13, in the worst case, the name of the component could be used to find the chemical, even though it is not the best option since compounds with the same name could have different variations, leading to repeated compounds that originally were different. All rows have labels for all 3 endpoints. It is clear a large amount of data is missing for toxicity features, and most compounds do not present genotoxic information (they do not have genotoxic ID, meaning that no genotoxic assays were done for the compound). After counting the missing data per feature, it was determined that the CAS_NUMBER column had the lower number of nan-values, so it could be used, in the worst case, to get the SMILES

strings using PUBCHEM API.

It was also concluded that the majority of the missing values were of duplicated compounds, so, after handling it, the missing data was expected to be substantially low.

Figure 20 allows to better understand the distribution of compounds across different labels (columns). The plot illustrates that a considerable portion of the data is impracticable for analysis, exceeding 50% for each label. Among these labels, the IS_CARCIINOGENIC target stands out as having the fewest viable compounds.

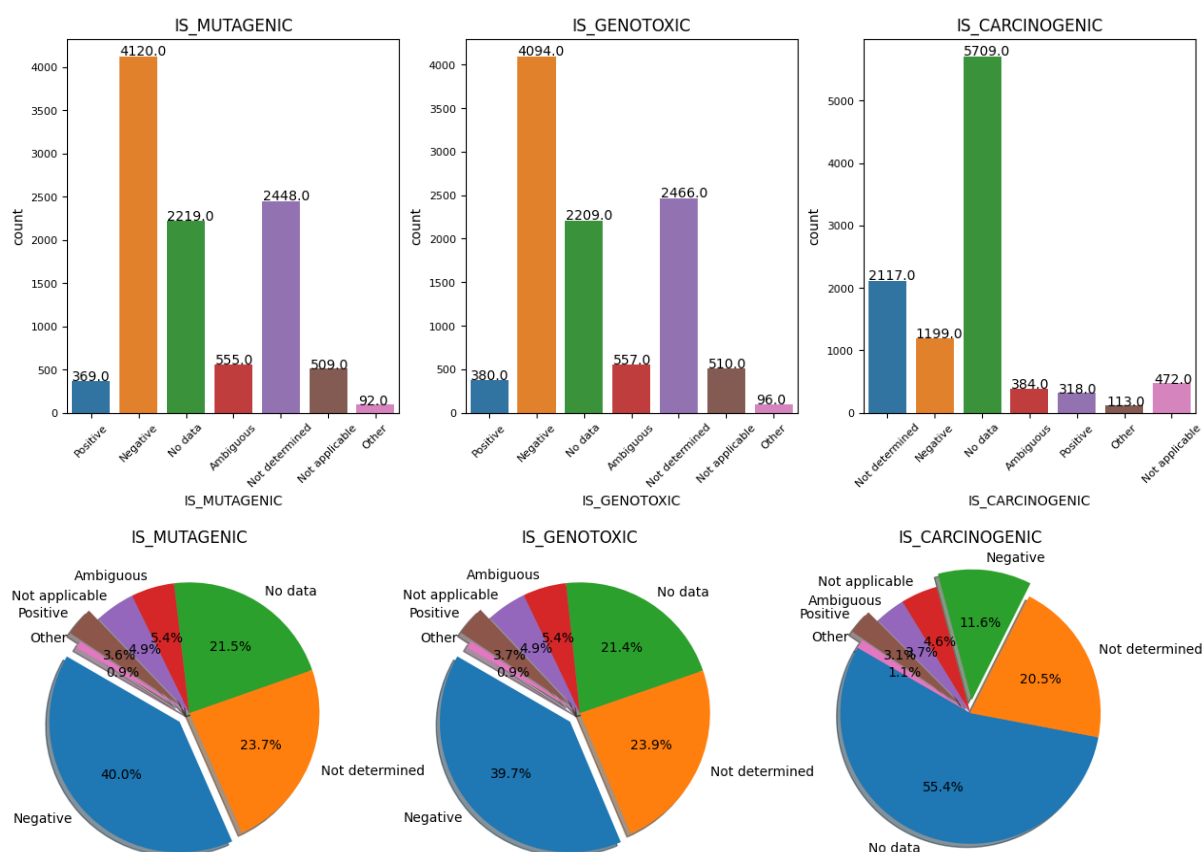


Figure 20: Distribution of data for each endpoint. Each graph shows the number of rows for each different endpoint. Top graph row presents the counts and bottom one presents the percentages. Note that Positive and Negative labels are highlighted.

The Figure 20 manages to summarize the most important information analyzed from previous bar graphs. Overall, there were many negative cases ($\approx 30.4\%$) and fewer positive cases ($\approx 3.5\%$), clearly highlighting the positive minority and so the imbalanced distribution of the data.

Furthermore, was also performed an analysis of the different study categories included in the dataset (Figure 21) to check if it would be viable to only use the data about human health. Considering that the final number of compounds could even decrease after processing, the strategy was not followed.

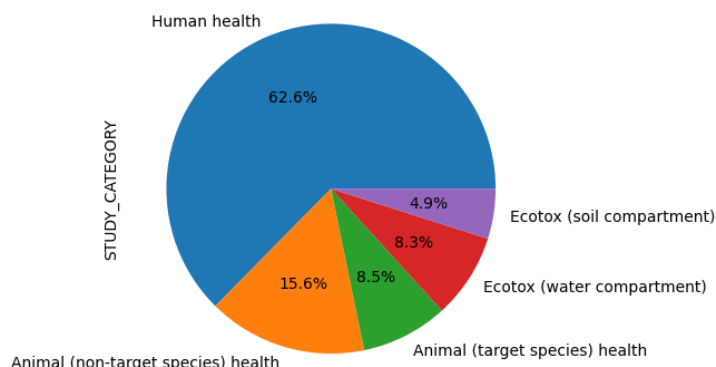


Figure 21: Several study categories addressed by OpenFoodTox dataset and percentages. Only viable data is being considered. The majority of the compounds (1835) concerns to human health, 708 compounds are about animals and the remaining 388 are included in the Ecotox Knowledgebase compartments [157].

6.2.3 Split of dataset

This section provides a more detailed explanation of the data distribution for training, validation, and testing. By the end of the gold phase, most of the data preprocessing had been completed. However, there were still some issues that needed to be addressed, including missing data in certain labels, the limited number of cases for specific classes, the definition of the split strategy for singletask and multitask models, and the percentage allocated to each data set.

Regarding missing values, they appeared only in specific labels, as cases with missing values in all labels had already been removed earlier. The missing values in question were intentionally kept as a way to distinguish instances where the label was neither positive (toxic) nor negative (non-toxic), but ambiguous due to lack of data. Initially, the idea was to leave these cases as NaN and use a custom loss function during model training. However, because the internal pipeline was not yet prepared to handle a customized loss function, and due to practical and time constraints, it was decided to impute these values with 0 (the majority class). While this approach placed greater emphasis on negative values, exacerbating the class imbalance, it was considered less detrimental than imputing with 1, which would classify a molecule as toxic, potentially leading to errors if the molecule was not actually toxic. Both methods would introduce some error and noise to the data, as imputing a value of 0 might misclassify a molecule that could indeed be toxic. This method was applied only to multitask models, but no imputation was performed on the test set to prevent data leakage (missing values were removed).

The final dataset, after preprocessed and ready for model input, consisted of 3102 cases with class distributions varying across labels. For the singletask models, this final dataset was split into training,

validation, and test sets. Fifteen percent (15%) of the data from the initial split was used for testing, and 15% of the remaining training data from the second split was used for validation. The splits were performed using stratification to reduce the impact of class imbalance.

For the multitask models, the process was exactly the same. However, to perform stratification, a new column was created by concatenating each label to generate a string that could be used to identify the different classes. Classes with a frequency of less than two were removed from this split and appended to the training set after the split. The three datasets (training, validation, and test) were split using the same seed and stored for future use in model training.

6.2.4 Model training

Model training is a crucial step in the development of ML systems, where a model learns to identify patterns and make decisions based on data. The effectiveness of the training process directly impacts the model's performance on specific tasks, and different approaches can be employed to optimize this process.

In this section, two training approaches are discussed: singletask training, which focuses on solving a single specific task, and multitask training, where the model is trained to learn and perform multiple tasks simultaneously.

Table 14: Set of options for parameters of light and medium presets.

Parameter	Possibilities
Models	DeepChem- graph-based neural networks and SMILES-based architectures (Smiles2Vec, TextCNN, etc.) Autogluon models- traditional ml algorithms like decision trees, gradient boosting, and basic neural networks (using tabular format) Omnia- mix of traditional classifiers/regressors (like RF, LR) and DL models (AutoEncoders, CNNs, RNNs)
Featurizer	2D Descriptors, MORSE, Mol2Vec Embeddings, 3D Descriptors, MACCS Keys Fingerprint, Morgan Fingerprint, Atom Pair Fingerprint, Topological Fingerprint, Layered Fingerprint
Standardizer	Basic ChEMBL, Custom (simple, heavy, super heavy), None
Scalers	Standard

To automate the training process, OmniumAI's internal pipelines were used to train the models. The company provides a set of presets that define a range of possible training configurations, including the selection of molecular descriptors, standardization and scaling methods, as well as the appropriate models. As shown in Table 14, this set is extensive and contains several interesting options that can be utilized. Given the available resources and the time required for training, the light or medium presets were selected.

The entire process was primarily conducted using notebooks on *Databricks*, with the creation of appropriate GPU-enabled clusters to reduce training time and allow for the testing of multiple combinations.

In singletask training without CV, during the initial phase of the work, internal pipeline presets from OmniumAI were used to obtain a preliminary set of pipelines of interest that could serve as a starting point for future model training. Initially, only binary classification was performed for three generic endpoints: IS_MUTAGENIC, IS_CARCIINOGENIC, and IS_GENOTOXIC.

During model training, each run explored unique combinations of initial pipeline parameters, selected randomly from a predefined set of featurizers, standardizers and scalers (see Table 14). This approach aimed to identify the optimal pairing of features that best suit the dataset. To enhance robustness, the top three performing runs were incorporated into a soft voting ensemble. Here, each model contributed its weighted prediction probability, with the final classification determined by the aggregated probabilities across these selected models.

For hyperparameter tuning, the Optuna framework [114] was employed, with the Tree-structured Parzen Estimator (TPE) selected, a probabilistic model that estimates which hyperparameter values are more likely to lead to good performance, based on the results of previous trials. During hyperparameter optimization, 100 evaluations (runs or trials) were performed, with optimization beginning at the 60th trial. The metric used for optimization was balanced accuracy. However, to better understand potential overfitting and assess generalization capacity, additional relevant metrics were calculated on the test set.

Different optimizers were also compared, namely the random one, to test whether more different combinations could be useful for the models, replicating the same process just by changing the optimizer. However, the results were even worse, so it was decided not to proceed with this strategy.

The singletask training with CV, involved splitting the dataset into k-folds (5 splits), where the model was trained on k-1 folds and validated on the remaining fold, iterating over all folds to ensure each part of the dataset was used for both training and validation. Given the longer runtime of CV, the light preset configuration was employed for the model pipelines, prioritizing efficiency while still exploring a variety of model types and configurations, with minimal tuning to reduce computation time. Voting was not used, due to the already extended duration of the CV process. These models were then evaluated on each fold to compute the average performance across the folds, providing a more reliable estimate of the model's capacity for generalization. The key metric for optimization remained balanced accuracy, but similar to the earlier process, additional metrics were computed on the final test set to examine overfitting and overall generalization performance.

Multitasking was conducted to compare the influence of using additional labels on the current

prediction. Given the poor results obtained for singletask without CV, only multitask with CV was performed, following the exact same strategy as the singletask with CV.

6.2.5 Overcoming the impact of imbalanced data

Addressing the challenges posed by imbalanced data is critical in ML. Several strategies were implemented across each phase of the methodology to effectively mitigate the impact of this imbalance. Starting in **data preprocessing**, to ensure that the model was exposed to a representative sample of the data, a stratified split on each label was employed, ensuring that a representative subset of different classes was present on each set, and allowing the model to learn from both majority and minority classes during training and validation. The missing SMILES were imputed using relevant APIs, to preserve the largest number of instances. During **model training**, robust evaluation metrics that are particularly effective for imbalanced datasets were utilized, such as balanced accuracy, ROC AUC, and recall. Balanced accuracy provided a clearer view of model performance by considering the accuracy on both the majority and minority classes. ROC AUC assessed the trade-off between true positive rates and false positive rates across different thresholds, offering insight into the model's ability to distinguish between classes, especially the minority class. Special attention was given to recall for the minority class. Finally, in **model evaluation** stratified k-fold CV was applied to assess the generalization performance of the models on unseen data (robustness). Having implemented these strategies to effectively address the challenges posed by imbalanced data throughout the ML pipeline, the subsequent results and discussion will provide insights into the performance of the models.

Chapter 7

Results and Discussion

Considering the overview of the experimental setup already explained (subsection 6.2.4), the results from each experiment are presented alongside a detailed discussion of their implications. This integrated format enables an immediate and thorough analysis of the impact of each modeling approach or configuration adjustment, facilitating a comprehensive understanding of their effects on model performance.

7.1 Unsupervised Learning Analysis

In order to inspect if UL could be appropriate, was performed an unsupervised analysis that included some well-known algorithms such as PCA, t-SNE and K-means clustering. This analysis could help to reveal underlying patterns or structures within the data, relationships or groupings previously unknown, making use of dimensionality reduction of the data. As previously presented (section 3.1), 2D descriptors can capture essential chemical and structural properties, like molecular weight, bond connectivity, and specific substructures. These descriptors, unlike 3D descriptors, don't require spatial orientation, making them computationally efficient, and ideal for initial data exploration through unsupervised methods.

7.1.1 Principal Component Analysis

The processed dataset with nearly 3100 compounds was utilized. The SMILES depictions of these compounds were primarily standardized, the 2D descriptors were generated from these SMILES using RDKit. During the generation of the descriptors, missing data was generated, so the KNN algorithm with 3 neighbors was used to perform the imputation. Then, the descriptors were scaled, resulting in exactly 208 descriptors for each compound. Data was then ready to feed the PCA algorithm, and components from 1 to 150 were computed until a maximum explained variance of 0.95 was reached. The core of PCA is the choice of the number of components k , which is investigated by analyzing the cumulative explained variance ratio over component, as shown in Figure 22.

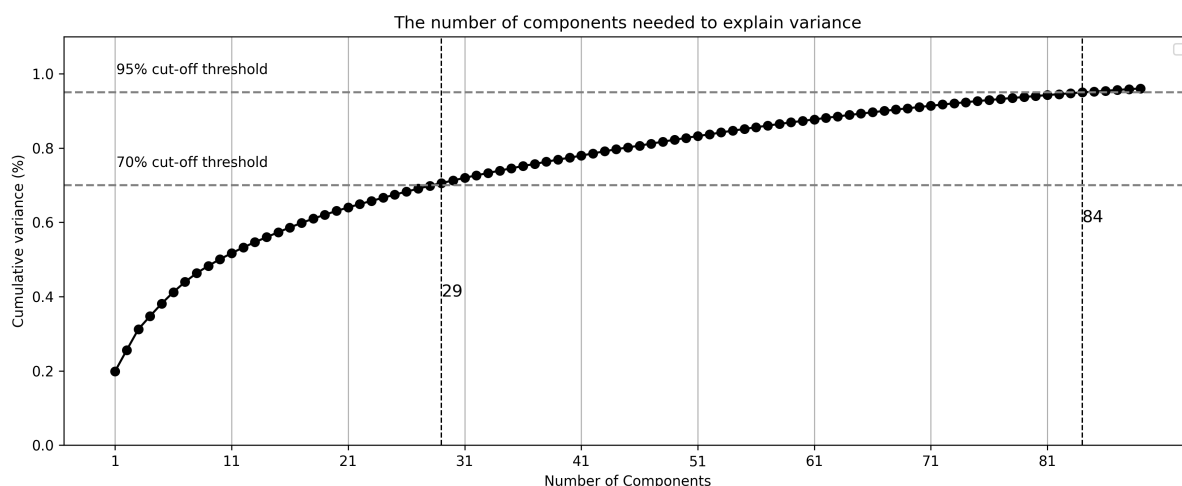


Figure 22: Evolution of the cumulative variance percentage with the number of components. It is represented 70% and 95% cut-off threshold for more practical analysis.

In the PCA plot (Figure 22), the cumulative variance explained by the principal components is shown as a function of the number of components. To reach a threshold of 95% cumulative variance, 84 principal components are necessary. This reflects that a significant number of components (84) is required to capture almost all variance present in the dataset, indicating the data's inherent complexity. After computing the PCA with 84 components, it was possible to use the labels as metadata to color the PCA, inspecting if there is any relationship with the real classes.

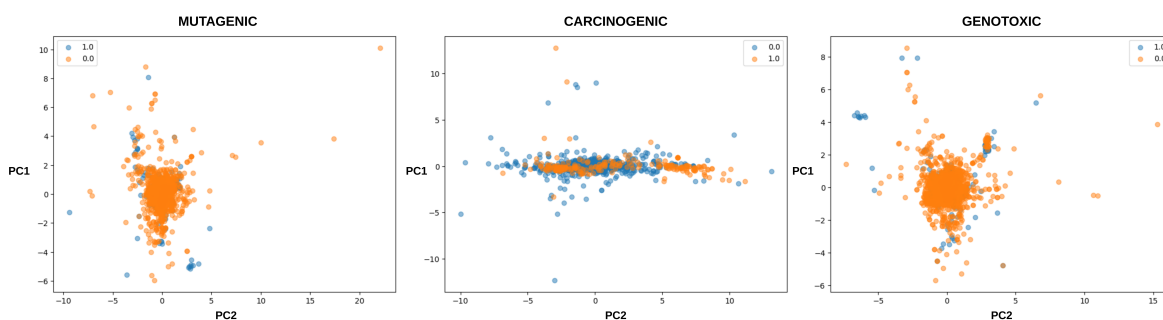


Figure 23: 2D plot between components 1 and 2 for each data (0 represents the negative values and 1 represents the positive ones).

In the mutagenic data (Figure 23), the two classes are not perfectly separated, with concentration near the center of the plot. A few blue points are scattered more broadly, suggesting that the mutagenic data is more dispersed compared to the non-mutagenic data. Several outliers are visible in the plot, primarily orange (non-mutagenic) points, which could indicate variability in the non-mutagenic data. In the carcinogenic data, there is a high degree of overlap between classes and no separation, as well as on the

genotoxic data, also highlighting the necessity of further exploration with different clustering methods to improve class separation (the classes of each label might not be non-linearly separable).

7.1.2 T-Distributed Stochastic Neighbor Embedding

Proper tuning of perplexity and monitoring the convergence of KL Divergence are crucial for generating meaningful and accurate t-SNE visualizations. The next analysis is centered in finding the best perplexity value using the elbow method. The first step was to calculate the KL Divergence for perplexity values from 5 to 50 with a step of 5.

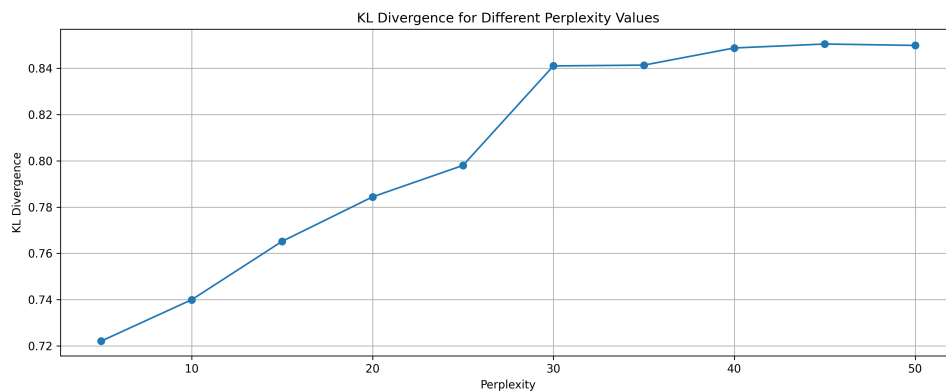


Figure 24: Evolution of perplexity values with KL Divergence.

As illustrated in Figure 24, there is a clear increase in KL Divergence up to a perplexity of 30, after which the curve starts to slow down. The elbow method suggests choosing the point where the rate of increase in KL Divergence starts to slow down, indicating diminishing returns from higher perplexity values. In the plot, the increase in KL Divergence begins to level off around a perplexity value of 30. After this point, the changes become less significant, so a perplexity value of 30 would be a reasonable choice.

t-SNE was then computed with a perplexity value of 30. The result of the algorithm for the first 2 components is illustrated in Figure 25.

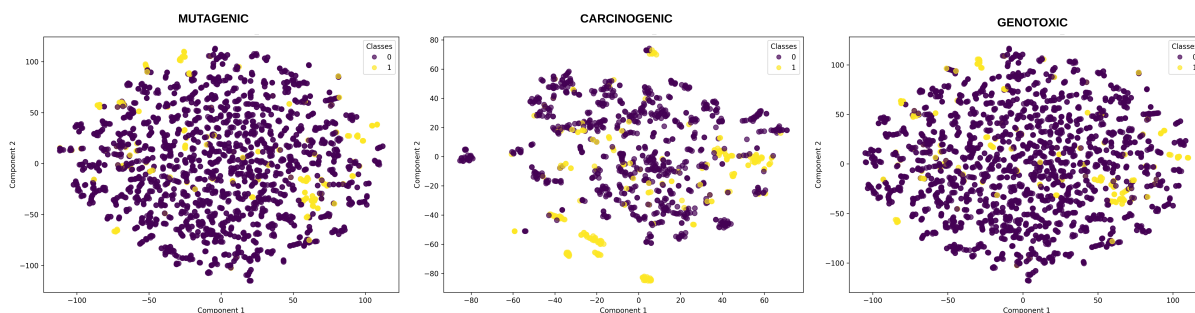


Figure 25: t-SNE first two components for each data (0 represents the negative values and 1 represents the positive ones).

Similar to PCA, there appears to be no obvious separation of classes between the components presented. As expected, the carcinogenic data is slightly different.

7.1.3 K-means clustering

First, 2 clusters were used with the algorithm since there were 2 types of compounds (toxic and non-toxic). Figure 26 show that the clustering patterns are similar across all three labels, with a tendency for the positive values to be more spread out, while the negative values cluster more densely in certain regions. There is a notable overlap between the clusters for both positive and negative label values in all three cases, suggesting that K-means clustering might not be fully effective at distinguishing between the two classes for these specific features.

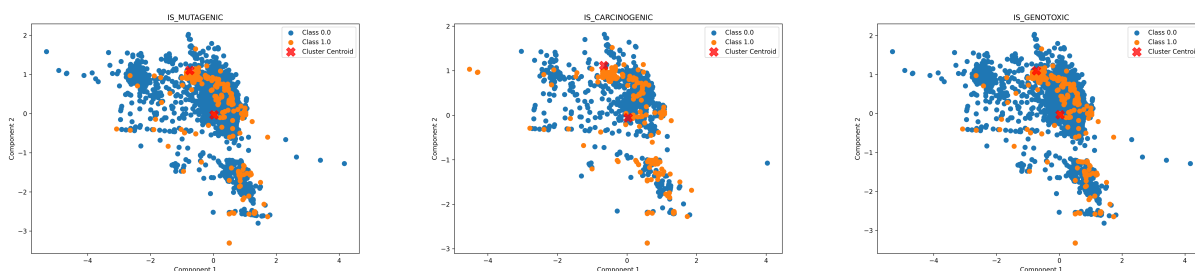


Figure 26: K-means components 1 and 2 for each data (0 represents the negative values and 1 represents the positive ones), with 2 clusters.

Since there is no obvious separation of classes, it was tried to understand whether there was an optimal number of clusters that would bring greater class separation. In a similar way, it was analyzed how the number of clusters influence the WCSS. It is clearly visible in Figure 27 that there is an abrupt descent from cluster 3 to **4**, marking our elbow.

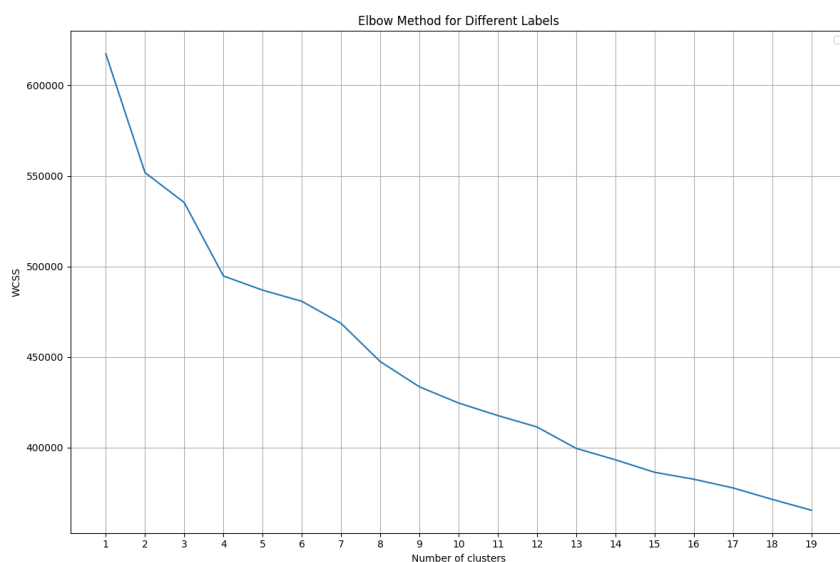


Figure 27: Evolution of number of clusters with WCSS, for each label.

After calculation the algorithm with 4 clusters and plotting the result in 2D, it is possible to conclude that the clusters vary in shape, showing some degree of mixing, which could suggest that the data is not easily separable using linear methods, potentially hinting at complex decision boundaries.

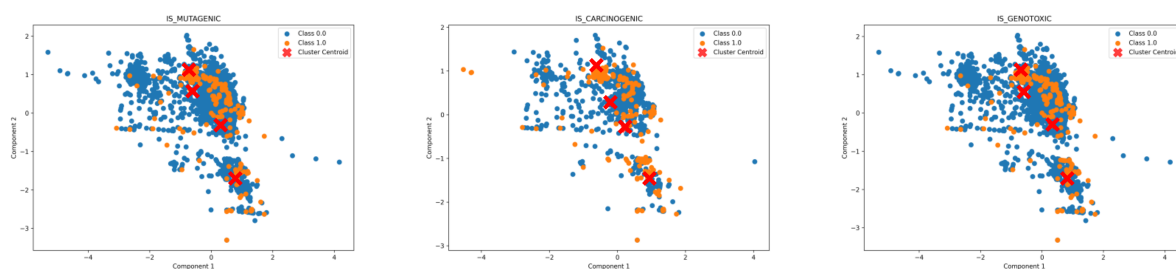


Figure 28: K-means components 1 and 2 for each data (0 represents the negative values and 1 represents the positive ones), with 3 clusters.

Comparing the real classes of the data (colored in Figure 28), with the clusters (marked with a cross), there are no evidence of a relation between each cluster with the toxicity of each compound.

The separation was also not evident, what allows to infer that traditional ML algorithms may be the most suitable approach for this problem, as well as DL, due to its ability to automatically learn complex patterns and representations from large datasets.

7.2 Singletask Learning

The performance of the singletask models on the binary classification task of toxicity prediction is presented. Each model was trained independently for the three binary labels. Upon completing the experiment, trials in a “complete” state, and free of errors, were filtered and sorted in descending order based on performance in the validation set. Duplicated trials were removed, yielding a final set of 10 unique trials, each representing different and relevant training configurations. The subsequent tables present the results of various ML models trained for each label. Each table includes performance metrics for both the test and validation sets, along with details of the best configuration employed. The `val` score indicates the model’s performance on the validation set, which is used to monitor and adjust the model during training. This step is crucial for hyperparameter tuning and preventing overfitting, as it evaluates the model on unseen data during the training process. The `test` score represents the model’s performance on the test set, which is an entirely separate dataset that the model encounters only after training and validation are complete. For model selection, the validation score is used, however, to better assess their performance, some observations are done with the test score.

7.2.1 Without cross-validation

The analysis of mutagenic (Table 15), carcinogenic (Table 16), and genotoxic (Table 17) results indicates generally low performance across singletask models, particularly in balanced accuracy scores, suggesting inefficiencies in capturing data patterns. The results highlight a trend where models tend to favor the majority class, resulting in low recall for positive cases, and in some instances, they perform worse than random classifiers.

For **mutagenic** data, RF (trial 69) achieves the highest validation score but struggles with test performance. In contrast, the Autoencoder MLP models (notably trials 46 and 40) perform better on the test set, especially when using Topological Fingerprint featurizers, indicating a potential advantage in feature representation.

In the **carcinogenic** and **genotoxic** analyses, the highest-performing models (both trial 12) employ the Layered Fingerprint featurizer, showing modest ROC AUC scores around 0.4551. However, challenges in distinguishing minority classes persist, primarily due to **dataset imbalance** and its reduced number of instances. The AutoencoderMLP’s flexibility with feature distributions with and without standardization is noted.

Table 15: Singletask binary classification results from training with medium preset for mutagenic label.

In green is the best model.

IS_MUTAGENIC									
Trial	BAL_ACC		F1	RECALL	ROC_AUC	Model	Featurizer	Standardizer	Scaler
	Val	Test							
69	0.5000	0.4697	0	0	0.1842	Random Forest	MORSE	Custom (simple)	None
46	0.4835	0.5012	0.0047	0.0024	0.4650	Autoenconder MLP	Topological FP	Custom (simple)	None
31	0.4821	0.4815	0	0	0.4426	Autonconder MLP	Topological FP	Basic	None
40	0.4821	0.500	0	0	0.4924	Autonconder MLP	Topological FP	Custom (heavy)	None
27	0.4657	0.4071	0.0276	0.0142	0.3574	Autonconder MLP	Atom Pair FP	Basic	None
108	0.4464	0.4848	0	0	0.1816	Random Forest	MORSE	None	None
26	0.4314	0.37612	0.0364	0.0189	0.3236	Autonconder MLP	Atom Pair FP	None	None
104	0.3214	0.3226	0	0	0.1507	Random Forest	Morgan FP (radius: 4)	Custom (simple)	None
51	0.3109	0.4706	0.0619	0.0322	0.3083	ChemCepcion	None	Custom (simple)	None
23	0.3036	0.3000	0	0	0.1173	Random Forest	Atom Pair FP	ChEMBL	None

Table 16: Singletask binary classification results from training with medium preset for carcinogenic label.

In green is the best model.

IS_CARCIINOGENIC									
Trial	BAL_ACC		F1	RECALL	ROC_AUC	Model	Featurizer	Standardizer	Scaler
	Val	Test							
12	0.4644	0.4583	0.0096	0.0049	0.4551	Autonconder MLP	Layered FP	Custom (heavy)	None
59	0.2771	0.2432	0.0095	0.0050	0.1156	Gradient Boosting	MACCS Keys FP	Custom (heavy)	None
18	0.2361	0.2803	0.0096	0.0050	0.0998	Logistic Regression	MACCS Keys FP	Custom (heavy)	None
102	0.2316	0.2234	0.0047	0.0025	0.1225	Gradient Boosting	MACCS Keys FP	Custom (simple)	None
14	0.2217	0.2334	0.0047	0.0024	0.0887	Gradient Boosting	MACCS Keys FP	Basic	None
81	0.2217	0.2334	0.0047	0.0024	0.0879	Gradient Boosting	MACCS Keys FP	None	None
55	0.2217	0.2802	0.0095	0.0049	0.0997	Logistic Regression	MACCS Keys FP	Custom (simple)	None
5	0.2113	0.2977	0.0138	0.0072	0.2651	Autonconder MLP	Atom Pair FP	Basic	None
10	0.2098	0.3271	0.0139	0.0072	0.3082	Autonconder MLP	Atom Pair FP	Custom (heavy)	None
33	0.2029	0.3048	0.0047	0.0024	0.0892	Logistic Regression	MACCS Keys FP	None	None

Table 17: Singletask binary classification results from training with medium preset for genotoxic label. In green is the best model.

IS_GENOTOXIC									
Trial	BAL_ACC		F1	RECALL	ROC_AUC	Model	Featurizer	Standardizer	Scaler
	Val	Test							
12	0.4658	0.4718	0.0048	0.0025	0.4551	Autonconder MLP	Layered FP	None	None
35	0.4286	0.4412	0	0	0.4445	Autonconder MLP	Layered FP	Custom (heavy)	None
19	0.3765	0.3395	0.0048	0.0024	0.3321	Autoenconder MLP	Topological FP	None	None
18	0.3768	0.1702	0.048	0.0261	0.1336	ChemCepcion	None	None	None
4	0.3260	0.1750	0.0136	0.0072	0.0996	Autoenconder MLP	Atom Pair FP	ChEMBL	None
37	0.3209	0.2449	0.0093	0.0049	0.0455	Logistic Regression	MACCS Keys FP	None	None
13	0.3082	0.2012	0.0046	0.0024	0.1762	Autoenconder MLP	Atom Pair FP	None	None
55	0.2993	0.1830	0.0046	0.0024	0.0884	Autoenconder MLP	MACCS Keys FP	None	None
54	0.2546	0.1820	0.0035	0.0024	0.0700	Random Forest	Atom Pair FP	None	None
69	0.2516	0.1800	0.0031	0.0023	0.0712	Directed Acyclic Graph (DAG)	ConvMol	None	None

In general, the results obtained across labels are **suboptimal**, as show in Figure 29. **Mutagenic** results present **higher values** when compared to **genotoxic** and **carcinogenic** results, with test scores ranging between 0.4 and 0.5. These scores also suggest that the classifiers are performing at a **near-random level**, offering limited practical utility for this study. Nevertheless, carcinogenic models demonstrate a better ability to correctly classify toxic compounds, as indicated by their superior, though still modest, **recall** values. This improved performance could be attributed to the use of standardizers, in conjunction with **Atom Pair Fingerprints**, once the best trials use this configurations, and they are not present in the configurations of top trials for other labels. In this context, the model itself does not appear to be the critical factor, as across all top trials, the **model architecture** remained constant (Autoencoder MLP), yet with fewer evidences.

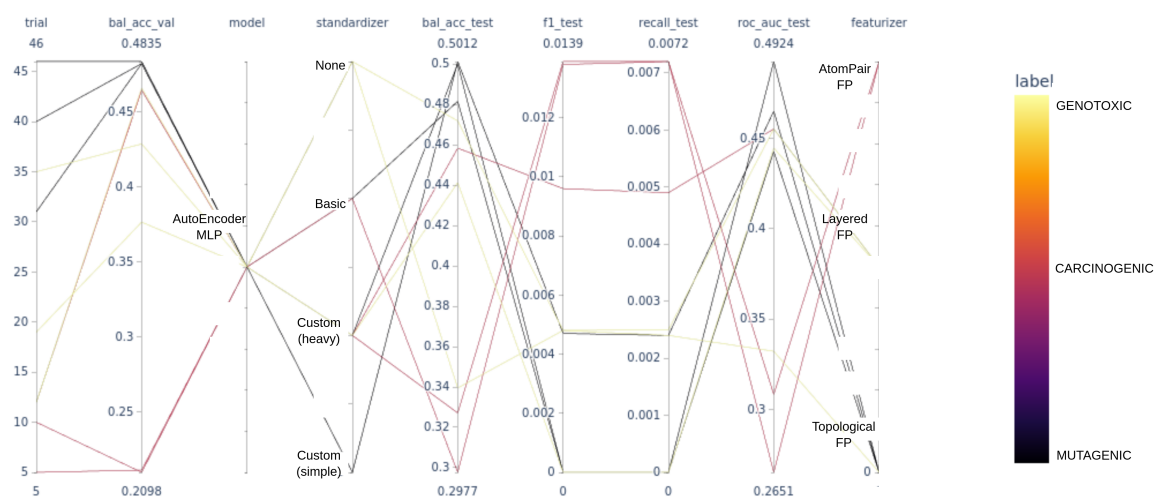


Figure 29: Parallel Diagram with top 3 trials without CV for each metric.

7.2.2 With cross-validation

The setups for singletask classification differ significantly: while this approach provides a robust assessment of model performance by training on multiple data splits, the non-CV method risks overfitting or underfitting due to reliance on a single train-test split.

This setup used 5-split CV with the balanced accuracy as validation metric.

When analyzing the **mutagenic** results (Table 18), there is a significant improvement of the metrics, with validation scores ranging between 0.8 and 0.9. Trial 27 is the top-performing model (validation score of 0.9499), with a ROC AUC of 0.7518, demonstrating its superiority in distinguishing classes. This model's robust performance is further supported by its strong balanced accuracy in test data (0.7518) and F1 score (0.6182), and recall of 0.5152, reflecting its proficiency in predicting the positive class. Besides having the best validation score, trial 17 outperforms in terms of test score, what can evidence the utility of graph-based featurizers and models.

Based on the validation scores, trial 28 follows with a ROC AUC of 0.8310, which, although slightly lower than trial 17, still signifies a strong predictive capability. The use of TextCNN, as predicted, is effective in processing molecular sequence data in a way that captures the key structural features related to mutagenicity. The test balanced accuracy (0.8253) and recall (0.6667) suggests that while this model performs well overall, it slightly underperforms in predicting the minority class when compared to trials 27 and 17.

Trial 8 follows closely with an ROC AUC of 0.8253, using the same Pagtn architecture as trial 17. This

similarity in performance between trials 8 and 17 reinforces the effectiveness of graph based models in capturing mutagenic signals from molecular data.

The **carcinogenic** data results (Table 19) shows strong model performance, also indicating effective differentiation between carcinogenic and non-carcinogenic compounds. Trial 29 is the top performer, achieving a balanced accuracy on the validation set of 0.9321, and an F1 score on the test set of 0.9638, showcasing excellent recall for the minority class at 0.9801. This model uses a CNN architecture with a Layered Fingerprint featurizer, demonstrating effective feature extraction with further additional preprocessing. Additionally, graph-based architectures still stood out.

The **genotoxicity** data results (Table 20) also shows strong model performance. Trial 32 is the top performer, achieving a test ROC AUC of 0.8859 with a TextCNN architecture, balanced accuracy of 0.9362 (validation set), and an F1 score of 0.9823 (test set). Its recall of 0.9930 highlights its effectiveness in identifying genotoxic compounds and detecting the minority class. Overall, TextCNN models outperformed graph-based models in this context, suggesting that increased complexity in data may detract from performance.

Table 18: Singletask binary classification results from training with light preset for mutagenic label. In green is the best model.

IS_MUTAGENIC									
Trial	BAL_ACC		F1	RECALL	ROC_AUC	Model	Featurizer	Standardizer	Scaler
	Val	Test							
27	0.9499	0.7518	0.6182	0.5152	0.7518	TextCNN	None	None	None
26	0.9499	0.7693	0.6667	0.5455	0.7693	TextCNN	None	None	None
28	0.9446	0.8310	0.7719	0.6667	0.8310	TextCNN	None	None	None
8	0.9395	0.8253	0.7097	0.6667	0.8253	Pagtn	PagtnMolGraph	None	None
5	0.9301	0.8078	0.6667	0.6364	0.8078	Pagtn	PagtnMolGraph	None	None
6	0.9275	0.7681	0.6545	0.5455	0.7681	Pagtn	PagtnMolGraph	None	None
21	0.9254	0.7821	0.6667	0.5758	0.7821	GraphConv	MolGraphConv	None	None
29	0.9230	0.7238	0.5882	0.4545	0.7238	TextCNN	None	None	None
17	0.9227	0.8661	0.7246	0.7576	0.8661	Pagtn	PagtnMolGraph	None	None
15	0.8992	0.7075	0.5490	0.4242	0.7075	Directed Acyclic Graph (DAG)	ConvMol	Basic	None

Table 19: Singletask binary classification results from training with light preset for carcinogenic label. In green is the best model.

IS_CARCIINOGENIC									
Trial	BAL_ACC		F1	RECALL	ROC_AUC	Model	Featurizer	Standardizer	Scaler
	Val	Test							
29	0.9321	0.8285	0.9638	0.9801	0.9144	CNN	Layered Fingerprint	None	None
5	0.9240	0.8085	0.9781	0.9883	0.9003	Pagtn	PagtnMolGraph	None	None
7	0.9226	0.7907	0.9651	0.9814	0.8832	Directed Acyclic Graph (DAG)	ConvMol	Basic	None
13	0.9178	0.7930	0.9737	0.9860	0.9022	Pagtn	PagtnMolGraph	None	None
30	0.9168	0.8420	0.9595	0.9781	0.9149	CNN	Layered Fingerprint	None	None
1	0.9154	0.8273	0.9594	0.9781	0.8817	Pagtn	PagtnMolGraph	Basic	None
28	0.9083	0.7883	0.9565	0.9767	0.8466	Directed Acyclic Graph (DAG)	ConvMol	Basic	None
4	0.9068	0.7918	0.9694	0.9837	0.8809	GraphConv	MolGraphConv	None	None
9	0.9047	0.7467	0.9605	0.9790	0.8957	Directed Acyclic Graph (DAG)	ConvMol	None	None
12	0.9041	0.7953	0.9824	0.9907	0.8840	TextCNN	None	None	None

Table 20: Singletask binary classification results from training with light preset for genotoxic label. In green is the best model.

IS_GENOTOXIC									
Trial	BAL_ACC		F1	RECALL	ROC_AUC	Model	Featurizer	Standardizer	Scaler
	Val	Test							
32	0.9362	0.7108	0.9866	0.9930	0.8859	TextCNN	None	None	None
26	0.9264	0.7382	0.9823	0.9907	0.9061	TextCNN	None	None	None
27	0.9246	0.6988	0.9955	0.9977	0.8793	TextCNN	None	None	None
23	0.9234	0.7478	0.9649	0.9814	0.8909	TextCNN	None	None	None
36	0.9229	0.7764	0.9651	0.9811	0.8666	Pagtn	PagtnMolGraph	None	None
25	0.9216	0.7131	0.9955	0.9977	0.8602	TextCNN	None	None	None
14	0.9140	0.7919	0.9694	0.9837	0.9001	GraphConv	MolGraphConv	Basic	None
24	0.9140	0.8006	0.9908	0.9951	0.8787	CNN	MACCSKeys FP	None	None
19	0.9106	0.7776	0.9694	0.9837	0.8831	GraphConv	MolGraphConv	Basic	None
29	0.9102	0.7513	0.9779	0.9884	0.8328	Directed Acyclic Graph (DAG)	ConvMol	None	None

Focusing on the **balanced accuracy** score from the validation set, the results (Figure 30 show that singletask models trained on **mutagenic data** tend to exhibit superior performance, further confirmed by the test score, even though some trials yield slightly better results on the test set for **carcinogenic data**. On the other hand, models applied to **genotoxic data** display comparatively lower performance in terms of balanced accuracy. However, they outperform the other models in metrics such as **recall**, **F1 score**, and **ROC AUC**, suggesting a greater ability to distinguish between the two classes and

accurately predict the minority class. Only the **carcinogenic** models show comparable performance in this regard.

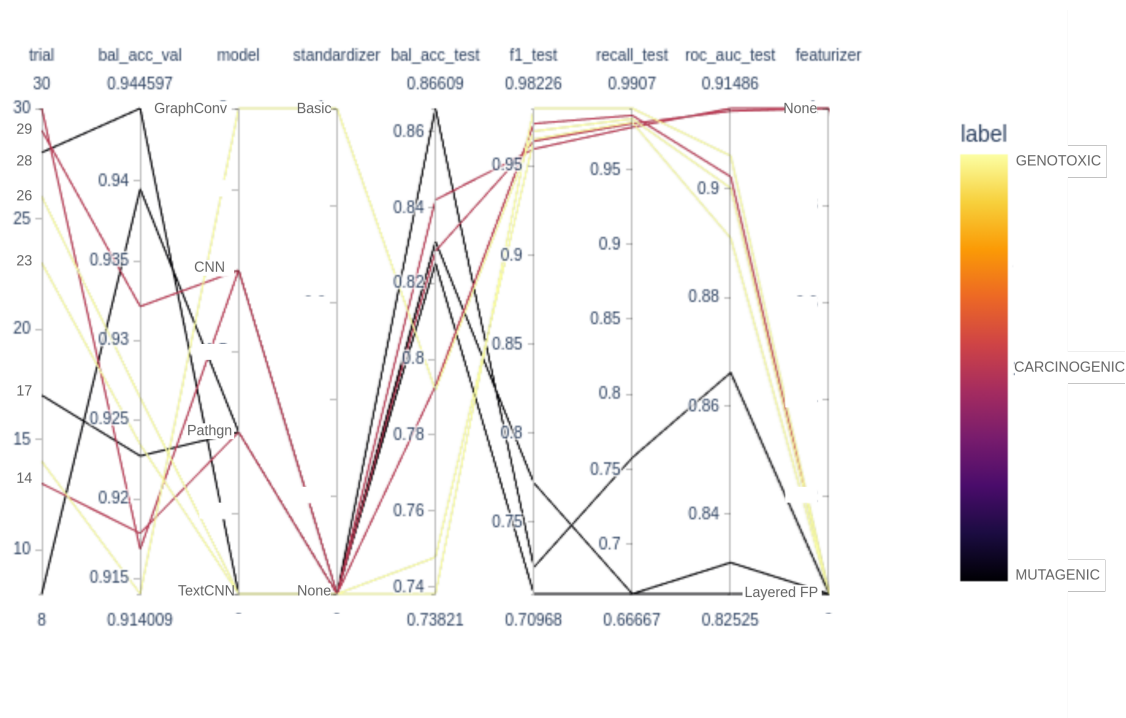


Figure 30: Parallel Diagram with top 3 trials with CV for each metric.

7.2.3 Comparative Analysis (with and without cross-validation)

Across all three tasks, trials employing CV demonstrate substantially better performance, particularly in terms of the F1 score and ROC AUC. Notably, in all cases where CV was not used, the F1 score plummeted to near-zero, while the ROC AUC either bottoms out or is negligible. This was expected, as CV is a more robust method to evaluate the performance of a model, and given the reduced size of the validation set (without CV). The non-CV models are likely highly overfit to their respective training sets, failing to generalize for the minority class.

Overall, some reasons why CV works better than without could be because in CV **all the data** is used for training while without there is always a validation set that is left out. Another reason is that for CV the optimizing process do not occurs for a single validation set, making it more difficult to overfit this set, which seems to be exactly what happens without CV (overfitting). The use of graph-based featurizers and models seems to be relevant for prediction: most of the top trials for the 3 labels use these architectures, which are able to directly capture the molecular structures, preserving spatial relationships and connectivity patterns, crucial for understanding chemical properties and interactions.

7.3 Multitask Learning

The performance of the multitask models on the binary classification task of toxicity prediction is presented. Each model was trained using the information of the three binary labels. The strategy for obtaining the results was similar to singletask with CV.

7.3.1 With cross-validation

Observing Table 21, **balanced accuracy** on the validation set ranges between 0.8 and 0.9, while on the test set, the values vary between 0.6 and 0.8. The top-performing model, trial 55 (CNN model), achieves the highest **balanced accuracy** of 0.9232 on the validation set and the second highest on the test set. Trial 30 (Weave model) has the best test set **ROC AUC** of 0.9427.

Equally important is the high **recall** observed in these models, as it reflects the models' ability to correctly predict positive cases. For instance, trial 8 (CNN model) achieves a **recall** of 0.7006, indicating a good performance predicting the positive class.

The trials with standardization do not show a consistent improvement over those without it, indicating that standardization might not provide a clear advantage in this setting.

Table 21: Multitask binary classification results from training with light preset and CV for each label. In green is the best model.

Multitask									
Trial	BAL_ACC		F1	RECALL	ROC_AUC	Model	Featurizer	Standardizer	Scaler
	Val	Test							
55	0.9232	0.8403	0.7348	0.6962	0.9206	Pagtn	PagtnMolGraph	None	None
8	0.9106	0.8393	0.7006	0.7039	0.9047	CNN	Topological FP	None	None
0	0.9096	0.8244	0.6821	0.6742	0.8976	CNN	Topological FP	None	None
26	0.9093	0.8203	0.7040	0.6569	0.9292	Pagtn	PagtnMolGraph	Basic	None
48	0.9077	0.8306	0.6944	0.6843	0.9049	CNN	Layered FP	None	None
7	0.9059	0.8301	0.7164	0.6768	0.9061	CNN	Morgan FP (radius: 2)	None	None
9	0.8852	0.8365	0.7012	0.6959	0.9116	CNN	Morgan FP (radius: 2)	Basic	None
22	0.8791	0.8311	0.7276	0.6763	0.8953	DMPNN	DMPNNFeaturizer	None	None
50	0.8728	0.8642	0.7574	0.7455	0.9153	DMPNN	DMPNNFeaturizer	None	None
30	0.8586	0.8322	0.6968	0.6861	0.9427	Weave	WeaveFeaturizer	Basic	None

7.4 Comparative Analysis (Singletask vs. Multitask)

In **Singletask with CV MUTAGENIC (Trial 27 - TextCNN Model)**, the model achieves a high balanced accuracy of 0.7518 for mutagenicity, not outperforming the multitask model, which has a balanced accuracy of 0.8252 in that task. This result suggests that for mutagenicity, the multitask approach with the Patgn model may capture specific patterns more effectively than the multitask setup.

In **Singletask with CV CARCINOGENIC (Trial 29 - CNN Model)**, the **CNN model**, utilizing a **Layered Fingerprint (FP) featurizer**, attains a balanced accuracy of 0.8285 for carcinogenicity in the singletask configuration. However, this is lower than the 0.8742 achieved by the multitask model, indicating that while the layered FP featurizer aids the model (boosting the test score when compared to other singletask data), the multitask learning approach likely benefits more from shared patterns across tasks, resulting in superior performance (validation score).

In **Singletask with CV GENOTOXIC (Trial 32 - TextCNN Model)**, the **TextCNN model**, without any featurizer or standardizer, achieves a balanced accuracy of 0.7108 for genotoxicity. This is lower than the multitask model's balanced accuracy of 0.8216, as well. Featurizers and standardizers do not significantly influence model performance, and alone cannot surpass the gains made by multitask learning, which leverages information across related tasks.

As Table 22 suggests, multitask performs better overall, and featurizers can help, but they are not decisive.

Table 22: Overall top configurations.

Method	Trial	BAL_ACC			F1	RECALL	ROC_AUC	Model	Featurizer	Standardizer	Scaler
		Test (per task)	Val	Test							
Multitask with CV	55	MUTAGENIC: 0.8252	0.9232	0.8403	0.7348	0.6962	0.9206	Patgn	PatgnMolGraph	None	None
		CARCINOGENIC: 0.8742									
		GENOTOXIC: 0.8216									
Singletask with CV MUTAGENIC	27	N/A	0.9499	0.7518	0.6182	0.5152	0.7518	TextCNN	None	None	None
Singletask with CV CARCINOGENIC	29	N/A	0.9321	0.8285	0.9638	0.9801	0.9144	CNN	Layered FP	None	None
Singletask with CV GENOTOXIC	32	N/A	0.9362	0.7108	0.9866	0.9930	0.8859	TextCNN	None	None	None

7.4.1 Comparative Analysis with Literature

This section discusses the comparative analysis of the experimental results obtained in this study with findings from existing literature on ML approaches for toxicity classification. The used dataset is not present in any article, because this dataset was never used for this purpose yet, probably due to the many problems previously identified. However, a comparison is performed with some of the referenced studies.

For the **mutagenic** endpoint, our observed results are consistent with those in the literature, support-

ing the validity of our chosen methodology. Similar to [137], we use the ROC AUC metric to evaluate model performance, where their best-performing model, a RF, aligns with our top models in singletask learning. These studies highlight the predictive value of fingerprints, which were also key in our best configurations. While prior studies employed a singletask approach, our multitask strategy demonstrates superior overall performance. Additionally, findings from [140], which used CV with MACCS and 2D descriptors, affirm the effectiveness of tree-based models, achieving a singletask ROC AUC slightly above ours.

For the **carcinogenic** endpoint, our multitask model differs in architecture, dataset, and evaluation metrics compared to the model from [141], even though both aim to classify carcinogenicity. Their CarcinoPred-EL tool employs an ensemble of SVM, RF, and XGB models— a suitable choice for smaller, imbalanced datasets. In contrast, our model integrates neural and graph-based architectures, including CNN and D-MPNN, with CV. In our multitask framework, balanced accuracy, F1 score, and recall also demonstrate consistent and robust performance across trials.

For the **genotoxic** endpoint, the multitask models from [143] achieved an average ROC AUC of 0.83 for genotoxic Nuclear Response endpoints using SMILES augmentation, a technique that presents certain challenges. However, similar to these studies, we found that fingerprints and 2D descriptors were essential as feature sets. Our study's use of graph kernels further reinforced this approach, producing strong results consistent with those reported by [148].

Chapter 8

Application Development

The application developed throughout this project aims to assist in predicting the toxicity of chemical compounds for the three selected endpoints: mutagenicity, carcinogenicity, and genotoxicity. By inputting a compound's SMILES notation, the application can predict whether the compound will be toxic at these levels. This development fulfills the initial goal of integrating the constructed models into a user-friendly platform.

8.1 Concept and Features

This application is mainly targeted at users in the field of bioinformatics, although it is not restricted to a specific audience. In general, in terms of features (Table 23), the application allows for the input of chemical compounds (molecules) either manually or via file upload. Once the compounds are submitted, users can visualize their 2D structure. The data set is then processed through the prediction model selected by the user for each endpoint, and the application generates the predictions. These results are made available for download in CSV or PDF format, allowing users to perform a detailed analysis of the compound's toxicity.

The main features of the application are summarized in the following table:

Table 23: Key features of the developed app.

Feature	Description
Manual compound input	The app allows users to manual submit a compound SMILES via single input.
File compound input	The app allows users to submit a csv file containing a header with ID and SMILES.
Compound data deletion	The app allows users to delete the data submitted manually or via file.
2D visualization	The app allows users to visualize of the 2D structure of each compound.
Model selection	The app allows users to select a prediction model for mutagenicity, carcinogenicity, or genotoxicity.
Export prediction output	The app allows users to generate predictions and export them in a CSV or PDF format.

8.2 Technological stack

The entire application was developed **locally** using various technologies. For the **frontend**, React was used to build the user interface. Initially, **mockups** were created to visualize the design and flow of the application, which were then implemented. The **backend** was developed using Python, specifically with the **Flask** framework, which provides the necessary tools to manage server-side logic and handle requests from the frontend. No database system was used for this project, as the scope and complexity did not require persistent data storage at that level. All data is handled locally, relying on the memory of the user's machine. For version control, a **GitHub** repository was created to manage the code base and track changes throughout the development process. The backend runs in a **conda environment** [158], which includes all the required libraries to generate the 2D molecular structures and manage the **internal operations** of the application, such as using OmniumAI code to load the pipelines and predict.

The choice of **technologies** was based on the developer's familiarity with them, ensuring faster and more efficient implementation.

8.3 Architecture and Design

The architecture of the application is based on a client-server model, where the frontend (client) interacts with the backend (server) to perform predictions and visualize results. The design is modular and scalable, allowing for future enhancements while maintaining clear separation between the frontend and backend. Below is a breakdown of the architecture and design principles followed during development.

8.3.1 System Overview

The following diagram (Figure 31) provides a high-level overview of the system architecture. The system is generally composed of two primary components: the frontend, which provides the user interface, and the backend, which processes the molecular data, runs the prediction models, and serves the results to the user.

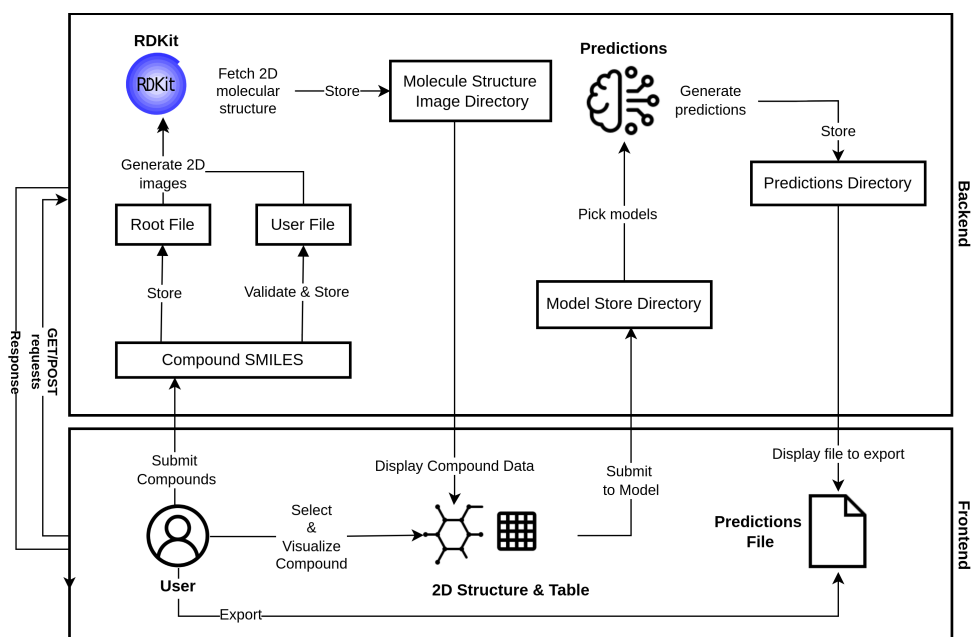


Figure 31: High-level system architecture

Frontend (Client)

The frontend was developed using React, a JavaScript library for building user interfaces. The user interface is designed to be intuitive and simple. Users can easily upload SMILES files or manually input them and receive results in a few steps. The frontend is responsible for:

- Accepting user inputs (manual or file-based SMILES notation);
- Displaying 2D molecular structures using third-party libraries (e.g., RDKit);
- Allowing users to select the prediction model (mutagenic, carcinogenic, or genotoxic);
- Communicating with the backend to send data and retrieve predictions;
- Displaying results to the user and providing options for downloading them in CSV or PDF format;

Some interfaces are shown in Figure 32, Figure 33 and Figure 34.

Insert
Select
Export

Insert the SMILES Notation of each compound

Type SMILES
Submit

100%
Upload Complete

ID	SMILES
1	C(C(=O)O)N
2	CC(C(=O)O)N
3	CC(=O)O
4	C1=CC=CC=C1
5	C1=CC(=O)C=C1C(=O)O
6	CC(=O)C(=O)O)N
7	C1CCCCC1
8	C1=CC=C(C(=C1)C(=O)O
9	C1CCCC1

Figure 32: Compound insertion page.

Insert
Select
Export

Select the model to use for each label

Choose the file to use:

☐ dataset_root.csv
☒ valid_smiles.csv

Choose a model for each label:

IS_MUTAGENIC
singletask_PAGTN

IS_CARCIINOGENIC
singletask_TEXTCNN

IS_GENOTOXIC
singletask_CNN

Predict

Figure 33: Model selection page.

Insert
Select
Export

Export Predictions

Select a file to preview

IS_CARCIINOGENIC_singletask_PAGTN
IS_MUTAGENIC_singletask_TEXTCNN
IS_GENOTOXIC_singletask_CNN

IS_CARCIINOGENIC
0
0
0
0

Export as CSV
Export as PDF

Figure 34: Export predictions page.

Backend (Server)

The backend, built with Flask (Python), handles the logic for receiving input data, processing molecular structures, and predicting toxicity. It consists of the following core tasks:

- **Data Handling:** Responsible for receiving and validating input from the frontend. Ensures that the SMILES are correctly formatted before passed to the prediction models;
- **Prediction Engine:** Contains the pre-trained models for the three endpoints (mutagenicity, carcinogenicity, and genotoxicity). Once a valid input is received, it is fed into the selected model, and the output is processed to be returned to the user;
- **2D Visualization Generator:** Using the RDKit library, this component generates 2D molecular structures based on the SMILES inputs, which are then displayed in the frontend;
- **Results Formatting:** Formats the prediction results into the chosen format (CSV or PDF), allowing users to download the output.

8.3.2 Data/work Flow and Interaction

All interactions between the frontend and backend are conducted via RESTful APIs, ensuring that the system is stateless and scalable.

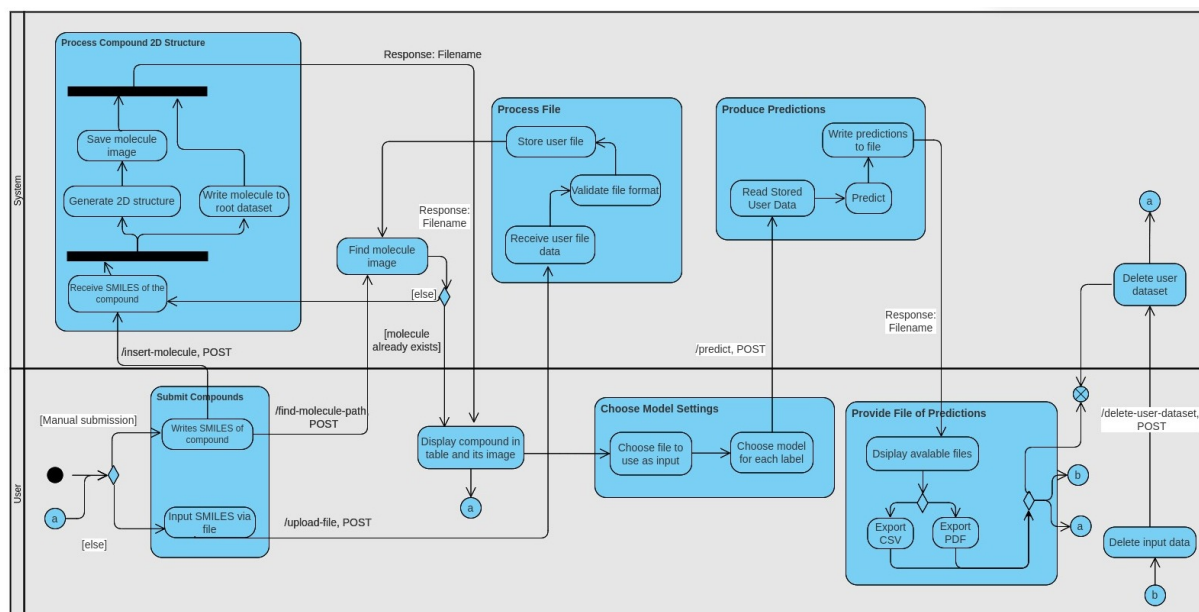


Figure 35: Activity Diagram that describes the flow between the user and the system.

The application workflow illustrated in the activity diagram in Figure 35, that comprises several distinct processes, each one associated with the handling, processing, and analysis of molecular data, with an emphasis on structure-based predictions.

The workflow initiates with the user submitting the compounds, either manually or by uploading a file containing the SMILES representation of the chemical structure and the ID of each compound. In the case of manual submission, the SMILES string is provided directly via input form. If the user uploads a file, the SMILES data is extracted from the file. In both cases, the SMILES are sent to the system via `/insert-molecule` (POST) or `/upload-file` (POST) endpoints, depending on the mode of submission.

Upon receiving the SMILES manually, the system processes the compound's 2D structure. This involves the generation of the compound's 2D structural image (using RDKit), which is saved and linked to a root dataset for subsequent reference. If the molecule has already been processed (i.e., a duplicate submission), the system immediately retrieves the corresponding image from the existing dataset instead of generating it anew. The response from this step includes the filename of the molecule image, which is used to display the compound in a table format on the frontend, along with its molecular representation.

On the other hand, users can upload files containing data related to the compounds. The file is stored via `/upload-file` (POST), and its format is validated to ensure compliance with system requirements. The user file data is then received and prepared for prediction, ensuring that all necessary inputs for further analysis are available.

Once the molecular data has been prepared, the user proceeds to the model selection phase. The user selects the appropriate input file for prediction from the stored dataset, then specifies which prediction model to use for each classification label. After the `/predict` (POST) request, this selection is processed, and the system reads the corresponding user data from storage. Subsequently, predictions are generated based on the chosen model and input data, and written to a file. After the predictions have been computed and saved to a file, the system allows the user to retrieve the file of predictions, in CSV or PDF.

Finally, the system incorporates a data deletion process, which enables users to delete their input data or entire datasets. This is done via the `/delete-user-dataset` (POST) endpoint. The process ensures that the system remains efficient by allowing users to manage their stored data and remove obsolete files when necessary. This structured workflow ensures a systematic approach to molecular data processing, with clear stages for data input, validation, processing, and output generation.

Chapter 9

Conclusion and Future work

9.1 Main Conclusions

This dissertation focused on predicting the toxicity of chemical compounds by employing ML and DL models, specifically targeting three toxicity endpoints— mutagenicity, carcinogenicity, and genotoxicity. The primary objective was to explore the performance of OmniumAI internal pipelines. Singletask and multitask learning approaches in toxicity prediction were followed to determine the most effective strategy for accurate classification. By leveraging data from the OpenFoodTox dataset, the study aimed to develop robust predictive models that could efficiently assess these endpoints, offering a solution to reduce time-consuming and expensive laboratory toxicity testing, while allowing to find potential toxic food ingredients.

To achieve these objectives, a comprehensive methodology was implemented. The study began with data collection, which involved the extraction of the dataset from EFSA public sources, and its subsequent analysis to understand the key features and endpoints. It was also discussed about what type of *in silico* model would be relevant, giving the available features. Class imbalance was detected, along with the available representations for molecular data. Following, the preprocessing involved transforming the data to a relevant sample that could be utilized following the aims of the project. An important step was to impute the missing SMILES using trusted APIs and to define a strategy for the repeated assays for the same compound.

Relatively to model training, both singletask and multitask learning techniques were employed, with hyperparameter optimization managed through the TPE algorithm. Different models were explored, including graph-based neural networks (D-MPNN) and convolutional neural networks (CNN), which have shown potential in capturing complex molecular patterns.

The obtained results demonstrated the advantage of multitask learning across all labels, where it consistently outperformed the singletask models. The multitask models benefitted from the shared knowledge across related tasks, which improved generalization capabilities and yielded higher predictive accuracy. In

particular, the top trial balanced accuracy reached 0.9232 in the validation set. The graph-based featurizers and models, as predicted, are in the top, confirming that directly capturing the molecular structures, preserving spatial relationships and connectivity patterns, is crucial for understanding chemical properties and interactions, and so assess toxicity.

While the project successfully developed robust models and an integrated software solution, several challenges were encountered. One significant issue was the class imbalance, particularly for the positive toxicity labels, which required careful handling during model training. Despite these challenges, the balanced accuracies achieved, particularly in multitask learning, underscore the potential of machine learning models in toxicity prediction.

9.2 Future Work

Although this study has provided promising results, several areas warrant improvement and further research to enhance both the model's accuracy and practical utility. One promising direction is data augmentation. While augmenting molecular data using SMILES strings or molecular fingerprints could increase the dataset size, caution is required to avoid introducing misleading augmentations that generate noise. Given that the toxicity of a molecule can be highly sensitive to even minor structural changes—such as modifications to functional groups or bonds—a SMILES-based augmentation approach could lead to erroneous predictions. A fingerprint-based approach, which encodes the presence or absence of particular substructures, could be more stable for increasing the number of positive instances in imbalanced datasets, even still presenting challenges in validation aspects.

Integration of data into the OmniumAI database is essential to ensure that a broader range of compounds and toxicity endpoints is available for future tasks. Expanding the database to include the compounds studied here would increase the predictive power of existing models and facilitate further research into toxicity prediction. This integration would also allow for continuous updates to the models, incorporating new data as it becomes publicly available.

Transfer learning techniques offer another promising avenue for improving performance on under-represented toxicity endpoints. Toxicity prediction often suffers from a lack of labeled data for certain endpoints, as was this case. Transfer learning, where models pre-trained on large datasets (even from unrelated domains) are fine-tuned for toxicity prediction, could help overcome these data limitations. This approach would leverage knowledge gained from larger datasets to improve predictions in cases where toxicity data is sparse, thus enabling better generalization across different toxicity endpoints.

Moreover, the dose-response data available in the OpenFoodTox dataset presents an opportunity to develop dose-response models or models based on various determining factors of toxicity, such as exposure conditions or host-specific factors. This would allow for a more nuanced understanding of how different doses impact toxicity outcomes, enhancing the real-world applicability of the models.

Explainable Artificial Intelligence (XAI) should also be incorporated into future developments. While the current models provide accurate predictions, their decision-making process remains a “black box”. By integrating XAI techniques, it would be possible to explain which molecular features most influence toxicity predictions. This would not only foster trust in the model's decisions, particularly in regulatory or clinical contexts, but also provide valuable insights into the underlying chemical properties that contribute to toxicity.

Finally, the software application developed as part of this project could benefit from further refinements. Enhancing the user interface and improving the application's scalability would allow for broader adoption within the scientific community. Improving accessibility and performance, particularly in large-scale predictions or real-time analyses, would ensure that the application becomes a more practical tool for researchers and industry professionals alike.

Bibliography

- [1] Takayuki Shibamoto and Leonard F. Bjeldanes. *Introduction to Food Toxicology*. Academic Press, 2nd edition, 2009.
- [2] Efrén Pérez Santín, Raquel Rodríguez Solana, Mariano González García, María Del Mar García Suárez, Gerardo David Blanco Díaz, María Dolores Cima Cabal, José Manuel Moreno Rojas, and José Ignacio López Sánchez. Toxicity prediction based on artificial intelligence: A multidisciplinary overview. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 11, 9 2021.
- [3] Philip Wexler, Ann de Peyster, Shayne C. Gad, Bruce D. Anderson, Michael A. Kamrin, Betty J. Loecey, Harihara M. Mehendale, Carey N. Pope, Lee R. Shugart, and Pertti J. Hakkinen. *Encyclopedia of Toxicology*, volume 1. Academic Press, 2nd edition, 2006.
- [4] Zhoumeng Lin and Wei-Chun Chou. Machine Learning and Artificial Intelligence in Toxicological Sciences. *Toxicological Sciences*, 189(1):7–19, 07 2022.
- [5] Xinxin Wang, Yamine Bouzembrak, AGJM Oude Lansink, and H. J. van der Fels-Klerx. Application of machine learning to the monitoring and prediction of food safety: A review. *Comprehensive Reviews in Food Science and Food Safety*, 21(1):416–434, 2022.
- [6] Arwa B. Raies and Vladimir B. Bajic. In silico toxicology: computational methods for the prediction of chemical toxicity. *WIREs Computational Molecular Science*, 6(2):147–172, 2016.
- [7] Daniel T. Teitelbaum. Introduction to toxicology: Occupational & environmental. In Bertram G. Katzung, Susan B. Masters, and Anthony J. Trevor, editors, *Basic & Clinical Pharmacology*, pages 1001–1002. McGrawHill Medical, 12th edition, 2012.
- [8] Karl K Rozman and John Doull. Dose and time as variables of toxicity. *Toxicology*, 144:169–178, 2000.
- [9] Valérie Maeder, Beate I. Escher, Martin Scheringer, and Konrad Hungerbühler. Toxic ratio as

- an indicator of the intrinsic toxicity in the assessment of persistent, bioaccumulative, and toxic chemicals. *Environmental Science and Technology*, 38:3659–3666, 7 2004.
- [10] Ilona Silins and Johan Högberg. Combined toxic exposures and human health: Biomarkers of exposure and effect. *International Journal of Environmental Research and Public Health*, 8:629–647, 2011.
- [11] Tea Štefanac, Dijana Grgas, and Tibela L. Dragičević. Xenobiotics—division and methods of detection: A review. *Journal of Xenobiotics*, 11:130–141, 12 2021.
- [12] UC Davis School of Veterinary Medicine. Toxtutor - toxicology msdt, 2024.
- [13] Lauren M. Aleksunes, David L. Eaton, Klaassen, and Curtis D. Casarett *amp*; Doull's *Toxicology: The Basic Science of Poisons, 9th edition*, chapter Principles of Toxicology, pages 5–18. McGraw-Hill Education, New York, NY, 2019.
- [14] Evgenios Agathokleous and Edward J Calabrese. Hormesis: The dose response for the 21st century: The future has arrived. *Toxicology*, 425:152249, 2019.
- [15] Edward J Calabrese. Hormesis: Why it is important to toxicology and toxicologists. *Environmental Toxicology and Chemistry*, 27:1451–1474, 7 2008.
- [16] Andrew D Wallace. Toxic endpoints in the study of human exposure to environmental chemicals. In *Progress in Molecular Biology and Translational Science*, Progress in molecular biology and translational science, pages 89–115. Elsevier, 2012.
- [17] Russell S Thomas, Richard S Paules, Anton Simeonov, Suzanne C Fitzpatrick, Kevin M Crofton, Warren M Casey, and Donna L Mendrick. The US federal tox21 program: A strategic and operational plan for continued leadership. *ALTEX*, 35(2):163–168, March 2018.
- [18] Henry C Pitot. The molecular biology of carcinogenesis. *Cancer*, 72:962–970, 8 1993.
- [19] A.I. Baba and C. Cătoi. *Comparative Oncology*. NCBI bookshelf. Publishing House of the Romanian Academy, 2007.
- [20] Hartmut Jaeschke, Gregory J. Gores, Arthur I. Cederbaum, Jack A. Hinson, Dominique Pessayre, and John J. Lemasters. Mechanisms of Hepatotoxicity. *Toxicological Sciences*, 65(2):166–176, 02 2002.

- [21] R John and A M Herzenberg. Renal toxicity of therapeutic drugs. *Journal of Clinical Pathology*, 62(6):505–515, May 2009.
- [22] P.S. Spencer and P.J. Lein. *Neurotoxicity*, page 489–500. Elsevier, 2014.
- [23] Jason R Richardson, Vanessa Fitsanakis, Remco HS Westerink, and Anumantha G Kanthasamy. Neurotoxicity of pesticides. *Acta neuropathologica*, 138:343–362, 2019.
- [24] Leila Peivasteh-Roudsari, Raziye Barzegar-Bafrouei, Kurush Aghbolagh Sharifi, Shamimeh Azimisalim, Marziyeh Karami, Solmaz Abedinzadeh, Shabnam Asadinezhad, Behrouz Tajdar-Oranj, Vahideh Mahdavi, Adel Mirza Alizadeh, et al. Origin, dietary exposure, and toxicity of endocrine-disrupting food chemical contaminants: A comprehensive review. *Heliyon*, 2023.
- [25] Dayne Filer, Heather B Patisaul, Thaddeus Schug, David Reif, and Kristina Thayer. Test driving toxcast: endocrine profiling for 1858 chemicals included in phase ii. *Current Opinion in Pharmacology*, 19:145–152, 2014. Gastrointestinal • Endocrine and metabolic diseases.
- [26] Antony J Williams, Christopher M Grulke, Jeff Edwards, Andrew D McEachran, Kamel Mansouri, Nancy C Baker, Grace Patlewicz, Imran Shah, John F Wambaugh, Richard S Judson, et al. The comptox chemistry dashboard: a community data resource for environmental chemistry. *Journal of cheminformatics*, 9:1–27, 2017.
- [27] Michael Kuhn, Ivica Letunic, Lars Juhl Jensen, and Peer Bork. The SIDER database of drugs and side effects. *Nucleic Acids Res*, 44(D1):D1075–9, October 2015.
- [28] Pekka Kohonen, Emilio Benfenati, David Bower, Rebecca Ceder, Michael Crump, Kevin Cross, Roland C Grafström, Lyn Healy, Christoph Helma, Nina Jeliaskova, et al. The toxbank data warehouse: Supporting the replacement of in vivo repeated dose systemic toxicity testing. *Molecular Informatics*, 32(1):47–63, 2013.
- [29] Philip Wexler. Toxnet: An evolving web resource for toxicology and environmental health information. *Toxicology*, 157(1):3–10, 2001.
- [30] Alessandra Roncaglioni, Anna Lombardo, and Emilio Benfenati. The vegahub platform: The philosophy and the tools. *Alternatives to Laboratory Animals*, 50(2):121–135, 2022. PMID: 35382564.
- [31] Minjun Chen, Ayako Suzuki, Shraddha Thakkar, Ke Yu, Chuchu Hu, and Weida Tong. DILIrank: the largest reference drug list ranked by the risk for developing drug-induced liver injury in humans. *Drug Discov Today*, 21(4):648–653, March 2016.

- [32] Emilio Benfenati, Alessandra Roncaglioni, Edoardo Carnesecchi, Matilda Mazzucotelli, Marco Marzo, Andrey Toropov, Alla Toropova, Rossella Baldin, Andrea Ciacci, Simona Kovarich, Luca Sartori, Chihae Yang, Tomasz Magdziarz, Bryan Hobocienski, and Aleksandra Mostrag. Maintenance, update and further development of efsa's chemical hazards: Openfoodtox 2.0. *EFSA Supporting Publications*, 18(3):6476E, 2021.
- [33] Thomas Engel. Basic overview of chemoinformatics. *Journal of Chemical Information and Modeling*, 46:2267–2277, 11 2006. doi: 10.1021/ci600234z.
- [34] Frank K. Brown. Chapter 35 – chemoinformatics: What is it and how does it impact drug discovery. *Annual Reports in Medicinal Chemistry*, 33:375–384, 1998.
- [35] Mike Hann and Richard Green. Chemoinformatics — a new name for an old problem? *Current Opinion in Chemical Biology*, 3:379–383, 1999.
- [36] B. Firdaus Begam and J. Satheesh Kumar. A study on cheminformatics and its applications on modern drug discovery. In *INTERNATIONAL CONFERENCE ON MODELLING OPTIMIZATION AND COMPUTING*, volume 38, pages 1264–1275. Elsevier Ltd, 2012.
- [37] Yashwant Pathak, Surovi Saikia, Sarvadaman Pathak, Jayvadankumar Patel, and Bhupendra Gopalbhai Prajapati, editors. *Artificial intelligence in bioinformatics and chemoinformatics*. CRC Press, London, England, October 2024.
- [38] Thomas Engel and Johann Gasteiger, editors. *Chemoinformatics*. Wiley-VCH Verlag, Weinheim, Germany, May 2018.
- [39] Jerzy Leszczynski, Anna Kaczmarek-Kedziera, Tomasz Puzyn, Manthos G. Papadopoulos, Heribert Reis, and Manoj K. Shukla. *Handbook of computational chemistry*, 2017.
- [40] National Research Council (U.S.). *A Framework to Guide Selection of Chemical Alternatives*. National Academies Press, Washington, D.C., DC, October 2014. ISBN-13: 978-0-309-31013-0.
- [41] Andrew Leach and Val Gillet. *An Introduction To Chemoinformatics*. Springer Dordrecht, 1 2007.
- [42] Adrià Fernández-Torras, Arnau Comajuncosa-Creus, Miquel Duran-Frigola, and Patrick Aloy. Connecting chemistry and biology through molecular descriptors. *Current Opinion in Chemical Biology*, 66:102090, 2022.

- [43] Chemistry Jonathan Clayden and Chemistry Nick Greeves. *Organic Chemistry*. Oxford University Press, Cary, NC, 2 edition, 2012.
- [44] David Weininger. Smiles, a chemical language and information system: 1: Introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28:31–36, 2 1988.
- [45] Daylight Chemical Information Systems, Inc. Daylight. Accessed: 2023-12-20, 2024.
- [46] David Weininger, Arthur Weininger, and Joseph L. Weininger. Smiles. 2. algorithm for generation of unique smiles notation. *Journal of Chemical Information and Computer Sciences*, 29:97–101, 1989.
- [47] Tengfei Ma, Jie Chen, and Cao Xiao. Constrained generation of semantically valid graphs via regularizing variational autoencoders, 2018.
- [48] N Oboyle and A Dalke. DeepSMILES: An adaptation of SMILES for use in Machine-Learning of chemical structures. *ChemRxiv*, 2018.
- [49] Gerard Parkin. A convenient approach for the construction of lewis structures by focusing on the valence requirements of outer atoms. *Journal of Chemical Education*, 100:4644–4652, 12 2023. doi: 10.1021/acs.jchemed.3c00702.
- [50] Robert E. Belford and Sunghwan Kim. Line notation (smiles and inchi), 2015.
- [51] Ingo Muegge and Prasenjit Mukherjee. An overview of molecular fingerprint similarity search in virtual screening. *Expert Opin Drug Discov*, 11(2):137–148, December 2015.
- [52] Jifeng Wang, Li Zhang, Jianqiang Sun, Xin Yang, Wei Wu, Wei Chen, and Qi Zhao. Predicting drug-induced liver injury using graph attention mechanism and molecular fingerprints. *Methods*, 221:18–26, 2024.
- [53] Delora Baptista, João Correia, Bruno Pereira, and Miguel Rocha. Evaluating molecular representations in machine learning models for drug response prediction and interpretability. *ournal of integrative bioinformatics*, 19, 2022.
- [54] Tiago Janela, Kosuke Takeuchi, and Jürgen Bajorath. Introducing a chemically intuitive Core-Substituent fingerprint designed to explore structural requirements for effective similarity searching and machine learning. *Molecules*, 27(7), April 2022.

- [55] Sunghwan Kim. Exploring chemical information in PubChem. *Curr Protoc*, 1(8):e217, August 2021.
- [56] Jurgen Bajorath. *Chemoinformatics for Drug Discovery*. John Wiley & Sons, Nashville, TN, November 2013.
- [57] Jean-Loup. Faulon and Andreas Bender. Handbook of chemoinformatics algorithms, 2010.
- [58] Maria Virginia Sabando, Ignacio Ponzoni, Evangelos E Milios, and Axel J Soto. Using molecular embeddings in QSAR modeling: does it make a difference? *Briefings in Bioinformatics*, 23(1):bbab365, 09 2021.
- [59] Empowering innovation & scientific discoveries | cas, 2023. Accessed: 2023-11-11.
- [60] Acs publications, 2023. Accessed: 2023-11-11.
- [61] Stephen R Heller. *The Beilstein Online Database*, volume 436, pages 1–9. American Chemical Society, 8 1990. doi:10.1021/bk-1990-0436.ch001.
- [62] Ida Schomburg, Antje Chang, Christian Ebeling, Marion Gremse, Christian Heldt, Gregor Huhn, and Dietmar Schomburg. BRENDA, the enzyme database: updates and major new developments. *Nucleic Acids Res*, 32(Database issue):D431–3, January 2004.
- [63] Gale database, 2023. Accessed: 2023-12-30.
- [64] Ufordat database, 2023. Accessed: 2023-12-30.
- [65] Bernd Page and Kristina Voigt. Recent history and development of environmental information systems and databases in germany. *Online Information Review*, 27:37–50, 1 2003.
- [66] Spresi database, 2023. Accessed: 2023-12-30.
- [67] Parva Kumar Sharma and Inderjit Singh Yadav. *Chapter 2 - Biological databases and their application*, pages 17–31. Academic Press, Jan 2022.
- [68] European molecular biology laboratory, 2023. Accessed: 2023-12-30.
- [69] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, Leonid Zaslavsky, Jian Zhang, and Evan E Bolton. PubChem 2019 update: improved access to chemical data. *Nucleic Acids Research*, 47(D1):D1102–D1109, 10 2018.

- [70] Anna Gaulton, Louisa J. Bellis, A. Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun McGlinchey, David Michalovich, Bissan Al-Lazikani, and John P. Overington. ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic Acids Research*, 40(D1):D1100–D1107, 09 2011.
- [71] Barbara Zdrazil, Eloy Felix, Fiona Hunter, Emma J Manners, James Blackshaw, Sybilla Corbett, Marleen de Veij, Harris Ioannidis, David Mendez Lopez, Juan F Mosquera, Maria Paula Magarinos, Nicolas Bosc, Ricardo Arcila, Tevfik Kizilören, Anna Gaulton, A Patrícia Bento, Melissa F Adasme, Peter Monecke, Gregory A Landrum, and Andrew R Leach. The ChEMBL Database in 2023: a drug discovery platform spanning multiple bioactivity data types and time periods. *Nucleic Acids Research*, 52(D1):D1180–D1192, 11 2023.
- [72] Meredith Ayers. Chempid: the free chemical database. *Reference Reviews*, 26(7):45–46, 2012.
- [73] John J Irwin, Teague Sterling, Michael M Mysinger, Erin S Bolstad, and Ryan G Coleman. Zinc: A free tool to discover chemistry for biology. *Journal of Chemical Information and Modeling*, 52:1757–1768, 7 2012. doi: 10.1021/ci3001277.
- [74] David S Wishart, Yannick D Feunang, An C Guo, Elvis J Lo, Ana Marcu, Jason R Grant, Tanvir Sajed, Daniel Johnson, Carin Li, Zinat Sayeeda, Nazanin Assempour, Ithayavani Iynkkaran, Yifeng Liu, Adam Maciejewski, Nicola Gale, Alex Wilson, Lucy Chin, Ryan Cummings, Diana Le, Allison Pon, Craig Knox, and Michael Wilson. DrugBank 5.0: a major update to the DrugBank database for 2018. *Nucleic Acids Research*, 46(D1):D1074–D1082, 11 2017.
- [75] Jacob White. Pubmed 2.0. *Medical Reference Services Quarterly*, 39(4):382–387, 2020. PMID: 33085945.
- [76] Greg Landrum et al. Rdkit: A software suite for cheminformatics, computational chemistry, and predictive modeling. *Greg Landrum*, 8:31, 2013.
- [77] Noel M O’Boyle, Michael Banck, Craig A James, Chris Morley, Tim Vandermeersch, and Geoffrey R Hutchison. Open babel: An open chemical toolbox. *Journal of Cheminformatics*, 3:33, 2011.
- [78] Open for innovation | knime, 2024. Accessed: 2024-01-03.
- [79] Stephan Beisken, Thorsten Meinl, Bernd Wiswedel, Luis F de Figueiredo, Michael Berthold, and Christoph Steinbeck. Knime-cdk: Workflow-driven cheminformatics. *BMC Bioinformatics*, 14:257, 2013.

- [80] Oleg Trott and Arthur J Olson. AutoDock vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *J Comput Chem*, 31(2):455–461, January 2010.
- [81] Christoph Steinbeck, Christian Hoppe, Stefan Kuhn, Matteo Floris, Rajarshi Guha, and Egon Willighagen. Recent developments of the chemistry development kit (cdk) - an open-source java library for chemo- and bioinformatics. *Current Pharmaceutical Design*, 12(17):2111–2120, June 2006.
- [82] Egon L Willighagen, John W Mayfield, Jonathan Alvarsson, Arvid Berg, Lars Carlsson, Nina Jelizkova, Stefan Kuhn, Tomáš Pluskal, Miquel Rojas-Chertó, Ola Spjuth, Gilleain Torrance, Chris T Evelo, Rajarshi Guha, and Christoph Steinbeck. The chemistry development kit (cdk) v2.0: atom typing, depiction, molecular formulas, and substructure searching. *Journal of Cheminformatics*, 9:33, 2017.
- [83] Stuart Russell and Peter Norvig. *Artificial intelligence*. Pearson, Upper Saddle River, NJ, 4 edition, November 2020.
- [84] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, New York, NY, 1 edition, August 2006.
- [85] Erik Brynjolfsson and ANDREW McAfee. Artificial intelligence, for real. *Harvard business review*, 1:1–31, 2017.
- [86] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. Adaptive Computation and Machine Learning series. MIT Press, London, England, November 2016.
- [87] Rajat Verma, Vishal Nagar, and Satyasundara Mahapatra. *Introduction to Supervised Learning*, chapter 1, pages 1–34. John Wiley & Sons, Ltd, 2021.
- [88] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dhharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, December 2018.
- [89] Zhi-Hua Zhou. *Machine Learning*. Springer Singapore, 2021.
- [90] Oona Rainio, Jarmo Teuho, and Riku Klén. Evaluation metrics and statistical tests for machine learning. *Scientific Reports*, 14:6086, 2024.

- [91] Xindong Wu, Vipin Kumar, J Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J McLachlan, Angus Ng, Bing Liu, Philip S Yu, Zhi-Hua Zhou, Michael Steinbach, David J Hand, and Dan Steinberg. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14:1–37, 2008.
- [92] S. Raschka. *Python Machine Learning: Unlock Deeper Insights Into Machine Learning with this Vital Guide to Cutting-edge Predictive Analytics*. Community experience distilled. Packt Publishing, 2015.
- [93] Bahzad Charbuty and Adnan Abdulazeez. Classification based on decision tree algorithm for machine learning. *Journal of Applied Science and Technology Trends*, 2(01):20 – 28, Mar. 2021.
- [94] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning: with Applications in R*. Springer Texts in Statistics. Springer New York, 2013.
- [95] João Gama. Combining classification algorithms. *Ai Communications*, 13:135–136, 01 2000.
- [96] Teuvo Kohonen. An introduction to neural computing. *Neural Networks*, 1(1):3–16, 1988.
- [97] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133, 1943.
- [98] Ethem Alpaydin. *Machine Learning, revised and updated edition*. The MIT Press Essential Knowledge series. MIT Press, London, England, August 2021.
- [99] Dr. Aishwarya P. Dr. Gopal Krishna Shyam, Farhana Kausar. Understanding and study of weight initialization in artificial neural networks with back propagation algorithm. *INFORMATION TECHNOLOGY IN INDUSTRY*, 9(1):1443–1450, March 2021.
- [100] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, and Jonathan Taylor. *Unsupervised Learning*, pages 503–556. Springer International Publishing, Cham, 2023.
- [101] Koosha Sharifani and Mahyar Amini. Machine learning and deep learning: A review of methods and applications. *World Information Technology and Engineering Journal*, 10(07):3897–3904, 2023.
- [102] Ajay Shrestha and Ausif Mahmood. Review of deep learning algorithms and architectures. *IEEE Access*, 7:53040–53065, 2019.

- [103] Van Hiep Phung and Eun Joo Rhee. A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets. *Applied Sciences*, 9(21), 2019.
- [104] Kai Heinrich, Patrick Zschech, Christian Janiesch, and Markus Bonin. Process data properties matter: Introducing gated convolutional neural networks (gcnn) and key-value-predict attention networks (kvp) for next event prediction with deep learning. *Decision Support Systems*, 143:113494, 2021.
- [105] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017.
- [106] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.
- [107] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
- [108] Oliver Kramer. *Scikit-Learn*, pages 45–53. Springer International Publishing, Cham, 2016.
- [109] Isaiah Hull. *TensorFlow 2*, pages 1–59. Apress, Berkeley, CA, 2021.
- [110] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.
- [111] A. Gulli and S. Pal. *Deep Learning with Keras*. Packt Publishing, 2017.
- [112] Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander Smola. Autogluon-tabular: Robust and accurate automl for structured data. *arXiv preprint arXiv:2003.06505*, 2020.

- [113] Brent Komer, James Bergstra, and Chris Eliasmith. Hyperopt-sklearn. *Automated Machine Learning: Methods, Systems, Challenges*, pages 97–111, 2019.
- [114] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 2623–2631, New York, NY, USA, 2019. Association for Computing Machinery.
- [115] Luis G. Valerio. In silico toxicology for the pharmaceutical sciences. *Toxicology and Applied Pharmacology*, 241(3):356–370, 2009.
- [116] Arwa B. Raies and Vladimir B. Bajic. In silico toxicology: computational methods for the prediction of chemical toxicity. *WIREs Computational Molecular Science*, 6(2):147–172, 2016.
- [117] Andrey A Toropov, Alla P Toropova, Ivan Raska, Jr, Danuta Leszczynska, and Jerzy Leszczynski. Comprehension of drug toxicity: software and databases. *Comput Biol Med*, 45:20–25, November 2013.
- [118] Omar Deeb and Mohammad Goodarzi. In silico quantitative structure toxicity relationship of chemical compounds: some case studies. *Curr Drug Saf*, 7(4):289–297, September 2012.
- [119] James Devillers. Methods for building QSARs. *Methods Mol Biol*, 930:3–27, 2013.
- [120] Chuipu Cai, Pengfei Guo, Yadi Zhou, Jingwei Zhou, Qi Wang, Fengxue Zhang, Jiansong Fang, and Feixiong Cheng. Deep Learning-Based prediction of Drug-Induced cardiotoxicity. *J Chem Inf Model*, 59(3):1073–1084, February 2019.
- [121] Shuangquan Wang, Huiyong Sun, Hui Liu, Dan Li, Youyong Li, and Tingjun Hou. ADMET evaluation in drug discovery. 16. predicting hERG blockers by combining multiple pharmacophores and machine learning approaches. *Mol Pharm*, 13(8):2855–2866, July 2016.
- [122] Yanli Wang, Stephen H. Bryant, Tiejun Cheng, Jiyao Wang, Asta Gindulyte, Benjamin A. Shoemaker, Paul A. Thiessen, Siqian He, and Jian Zhang. PubChem BioAssay: 2017 update. *Nucleic Acids Research*, 45(D1):D955–D963, 11 2016.
- [123] Teresa Maria Creanza, Pietro Delre, Nicola Ancona, Giovanni Lentini, Michele Saviano, and Giuseppe Felice Mangiatordi. Structure-Based prediction of hERG-Related cardiotoxicity: A benchmark study. *J Chem Inf Model*, 61(9):4758–4770, September 2021.

- [124] Limeng Pu, Misagh Naderi, Tairan Liu, Hsiao-Chun Wu, Supratik Mukhopadhyay, and Michal Brylinski. etoxpred: a machine learning-based approach to estimate the toxicity of drug candidates. *BMC Pharmacology and Toxicology*, 20:2, 2019.
- [125] Xu Wang, Bei Yun, Zihan Zhang, Xiaoxi Wang, Yifan Wu, Yubo Hu, Shiyi Fang, Junjie Lv, Lina Chen, and Wan Li. Darg: An integrated knowledge base for analyzing addictive drug related genes. *Genes & Diseases*, page 101369, 2024.
- [126] David Wishart, David Arndt, Allison Pon, Tanvir Sajed, An Chi Guo, Yannick Djoumbou, Craig Knox, Michael Wilson, Yongjie Liang, Jason Grant, et al. T3db: the toxic exposome database. *Nucleic acids research*, 43(D1):D928–D934, 2015.
- [127] Ruichao Xue, Zhao Fang, Meixia Zhang, Zhenghui Yi, Chengping Wen, and Tielu Shi. Tcmid: traditional chinese medicine integrative database for herb molecular mechanism analysis. *Nucleic acids research*, 41(D1):D1089–D1095, 2012.
- [128] Youjun Xu, Jianfeng Pei, and Luhua Lai. Deep learning based regression and multiclass models for acute oral toxicity prediction with automatic chemical feature extraction. *J Chem Inf Model*, 57(11):2672–2685, October 2017.
- [129] Dejun Jiang, Zhenxing Wu, Chang-Yu Hsieh, Guangyong Chen, Ben Liao, Zhe Wang, Chao Shen, Dongsheng Cao, Jian Wu, and Tingjun Hou. Could graph neural networks learn better molecular representation for drug discovery? a comparison study of descriptor-based and graph-based models. *Journal of Cheminformatics*, 13:12, 2021.
- [130] Xiao Li, Lei Chen, Feixiong Cheng, Zengrui Wu, Hanping Bian, Congying Xu, Weihua Li, Guixia Liu, Xu Shen, and Yun Tang. In silico prediction of chemical acute oral toxicity using multi-classification methods. *J Chem Inf Model*, 54(4):1061–1069, April 2014.
- [131] Kedi Wu and Guo-Wei Wei. Quantitative toxicity prediction using topology based multitask deep neural networks. *J Chem Inf Model*, 58(2):520–531, January 2018.
- [132] Jennifer H. Olker, Colleen M. Elonen, Anne Pilli, Arne Anderson, Brian Kinziger, Stephen Erickson, Michael Skopinski, Anita Pomplun, Carlie A. LaLone, Christine L. Russom, and Dale Hoff. The ecotoxicology knowledgebase: A curated database of ecologically relevant toxicity tests to support environmental research and risk assessment. *Environmental Toxicology and Chemistry*, 41(6):1520–1539, 2022.

- [133] Patricia Tomasulo. Chemidplus-super source for chemical and drug information. *Medical reference services quarterly*, 21(1):53–59, 2002.
- [134] Youjun Xu, Ziwei Dai, Fangjin Chen, Shuaishi Gao, Jianfeng Pei, and Luhua Lai. Deep learning for Drug-Induced liver injury. *J Chem Inf Model*, 55(10):2085–2093, October 2015.
- [135] Minjun Chen, Vikrant Vijay, Qiang Shi, Zhichao Liu, Hong Fang, and Weida Tong. FDA-approved drug labeling for the study of drug-induced liver injury. *Drug Discov Today*, 16(15-16):697–703, May 2011.
- [136] Xiao Li, Yaojie Chen, Xinrui Song, Yuan Zhang, Huanhuan Li, and Yong Zhao. The development and application of in silico models for drug induced liver injury. *RSC Adv.*, 8:8101–8111, 2018.
- [137] Charmaine S M Chu, Jack D Simpson, Paul M O'Neill, and Neil G Berry. Machine learning - predicting ames mutagenicity of small molecules. *J Mol Graph Model*, 109:108011, September 2021.
- [138] Congying Xu, Feixiong Cheng, Lei Chen, Zheng Du, Weihua Li, Guixia Liu, Philip W Lee, and Yun Tang. In silico prediction of chemical ames mutagenicity. *J Chem Inf Model*, 52(11):2840–2847, October 2012.
- [139] Thi Tuyet Van Tran, Hilal Tayara, and Kil To Chong. Ampred-cnn: Ames mutagenicity prediction model based on convolutional neural networks. *Computers in Biology and Medicine*, 176:108560, 2024.
- [140] Katja Hansen, Sebastian Mika, Timon Schroeter, Andreas Sutter, Antonius ter Laak, Thomas Steger-Hartmann, Nikolaus Heinrich, and Klaus-Robert Müller. Benchmark data set for in silico prediction of ames mutagenicity. *Journal of Chemical Information and Modeling*, 49:2077–2081, 9 2009. doi: 10.1021/ci900161g.
- [141] Li Zhang, Haixin Ai, Wen Chen, Zimo Yin, Huan Hu, Junfeng Zhu, Jian Zhao, Qi Zhao, and Hongsheng Liu. CarcinoPred-EL: Novel models for predicting the carcinogenicity of chemicals using molecular fingerprints and ensemble learning methods. *Sci Rep*, 7(1):2118, May 2017.
- [142] Romualdo Benigni, Cecilia Bossa, Ann M Richard, and Chihae Yang. A novel approach: chemical relational databases, and the role of the ISSCAN database on assessing chemical carcinogenicity. *Ann Ist Super Sanita*, 44(1):48–56, 2008.

- [143] Andreas Mayr, Günter Klambauer, Thomas Unterthiner, and Sepp Hochreiter. Deeptox: Toxicity prediction using deep learning. *Frontiers in Environmental Science*, 3, 2016.
- [144] Yuzhong Peng, Ziqiao Zhang, Qizhi Jiang, Jihong Guan, and Shuigeng Zhou. Top: A deep mixture representation learning method for boosting molecular toxicity prediction. *Methods*, 179:55–64, 2020. Interpretable machine learning in bioinformatics.
- [145] Abdul Karim, Avinash Mishra, M A Hakim Newton, and Abdul Sattar. Efficient toxicity prediction via simple features using shallow neural networks and decision trees. *ACS Omega*, 4:1874–1888, 1 2019. doi: 10.1021/acsomega.8b03173.
- [146] Chun Wei Yap. PaDEL-descriptor: an open source software to calculate molecular descriptors and fingerprints. *J Comput Chem*, 32(7):1466–1474, December 2010.
- [147] Anjali Setiya, Vinod Jani, Uddhavesb Sonavane, and Rajendra Joshi. Moltoxpred: small molecule toxicity prediction using machine learning approach. *RSC Adv.*, 14:4201–4220, 2024.
- [148] Jannis Born, Greta Markert, Nikita Janakarajan, Talia B. Kimber, Andrea Volkamer, María Rodríguez Martínez, and Matteo Manica. Chemical representation learning for toxicity prediction. *Digital Discovery*, 2:674–691, 2023.
- [149] Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chem. Sci.*, 9:513–530, 2018.
- [150] Kim Sydow Campbell. Collecting information: Qualitative research methods for solving workplace problems. *Technical communication*, 46(4):532–545, 1999.
- [151] Paul F Dubois, Konrad Hinsin, and James Hugunin. Numerical python. *Computers in Physics*, 10(3):262–267, 1996.
- [152] Brian Lipp. Modern data architectures with python: a practical guide to building and deploying data pipelines, data warehouses, and data lakes with python, 2023.
- [153] Databricks. Medallion Architecture - Data Engineering Wiki — dataengineering.wiki. <https://dataengineering.wiki/Concepts/Medallion+Architecture>. [Accessed 03-07-2024].

- [154] Sunghwan Kim, Paul A Thiessen, Tiejun Cheng, Bo Yu, and Evan E Bolton. An update on PUG-REST: RESTful interface for programmatic access to PubChem. *Nucleic Acids Res*, 46(W1):W563–W570, July 2018.
- [155] National Cancer Institute NCICADD Group. NCI/CADD Chemical Identifier Resolver — cactus.nci.nih.gov. <https://cactus.nci.nih.gov/chemical/structure>. [Accessed 03-07-2024].
- [156] J.L.C.M. Dorne, J. Richardson, A. Livaniou, E. Carnesecchi, L. Ceriani, R. Baldin, S. Kovarich, M. Pavan, E. Saouter, F. Biganzoli, L. Pasinato, M. Zare Jeddi, T.P. Robinson, G.E.N. Kass, A.K.D. Liem, A.A. Toropov, A.P. Toropova, C. Yang, A. Tarkhov, N. Georgiadis, M.R. Di Nicola, A. Mostrag, H. Verhagen, A. Roncaglioni, E. Benfenati, and A. Bassan. Efsa’s openfoodtox: An open source toxicological database on chemicals in food and feed and its future developments. *Environment International*, 146:106293, 2021.
- [157] Azizullah Azizullah and Donat-P. Häder. 10 - ecotox. In Donat-P. Häder and Gilmar S. Erzinger, editors, *Bioassays*, pages 199–219. Elsevier, 2018.
- [158] Anil Kumar and Manabendra Saharia. Python environment and basics. In *Python for Water and Environment*, pages 7–12. Springer, 2024.