Deep Painting: Cheminformatics approaches to connect the biological profiles of compounds with their structural information.

Dissertation
zur Erlangung des Grades
"Doktor der Naturwissenschaften"
im Promotionsfach Chemie
am Fachbereich Chemie, Pharmazie, Geographie und Geowissenschaften
der Johannes Gutenberg-Universität
in Mainz

Aishvarya Tandon
geb. in Lucknow 03. Oktober 1994

Mainz, den 14. Mai 2024

D77

1. Berichterstatter: [redacted]
2. Berichterstatter: [redacted]
Tag der mündlichen Prüfung:

Declaration

Publication, Presentations and Posters

The results presented in this thesis contributed to the following publication:

- A. Tandon, A. Santura, A. Pahl, H. Waldmann, P. Czodrowski, "Identification of Lysosomotropism using Explainable Machine Learning and Morphological Profiling Cell Painting Data", *RSC Medicinal Chemistry* **2024**, Manuscript Accepted.

Additionally, following talks and posters were presented at different conferences during the course of the PhD:

- **Talk** "X-FP: eXplainable FingerPrints" at the *RDKit User Group Meeting* **2023**
- **Poster** "X-FP: Software Presentation and Validation Case Study" at the *First Mainz and Friends Artificial Intelligence Conference* **2023**
- **Poster** "Lysosomotropism meets Machine-Learning: A cheminformatics approach to understand lysosomotropism" at the *RDKit User Group Meeting* **2022**
- **Talk** "The Art of Deep Painting" at the *German Conference on Cheminformatics* **2020** (Virtual)
- **Talk** "Deep Painting – an amalgamation of Cell Painting and Machine Learning" at the *RDKit User Group Meeting* **2020** (Virtual)

Contents

Abstract

In this doctoral thesis, different research projects were carried out, with the underlying theme of connecting the biological profiles of the compounds - determined by the morphological profiling Cell Painting Assay (CPA), with their structural information using different cheminformatics methods.

In the first project, Machine Learning (ML) models were developed to perform Quantitative Structure-Activity Relationship (QSAR) to predict the Cell Painting (CP) profiles of the compounds using different molecular representations. Different combinations of modeling methods and representations were explored, however, none of them had good predictive power. An hypothesis of models' low predictive power is that compounds with similar structures do not show similar bioprofiles.

In the second project, a generative modeling method was developed to generate numerous compounds having similar CP profiles. The prepared generative method did not generate compounds with relevant medicinal chemistry structures.

Lysosomotropism is a phenomenon where small, lipophilic compounds with basic moieties get trapped in the lysosomes. CPA is a reliable surrogate to identify compounds showing lysosomotropism because such compounds have a distinct phenotype. In the third project, it was reported that compounds having suitable lipophilicty and basicity do not qualify them to be lysosomotropic. To study which factors drive lysosomotropism and *vice-versa*, Matched Molecular Pair Analysis (MMPA) and Explainable Machine Learning (XML) was performed on suitable subset of the internal CPA data.

Modern ML methods like Neural Network (NN) and ensemble decision trees generally have high predictive power but suffer from "black box" nature as their predictions can not be explained. In the fourth project, an open-source software called *eXplainable FingerPrints* (X-FP) was developed. X-FP uses ML feature attribution methods like Shapley Additive Explanations (SHAP) in context of Morgan Fingerprints (MF), and generates an intuitive report highlighting important MF bits and the substructures encoded by them.

Induction is the percentage of significantly altered CP features, and is used as a measurement of compound's activity in the CPA. Induction prediction is desirable for chemists as they can prioritize the predicted CPA active compounds. A web-app which used a binary classifier NN was developed on the existing internal platform for this purpose.

Zusammenfassung

Im Rahmen dieser Doktorarbeit wurden verschiedene Forschungsprojekte durchgeführt, deren Ziel es war, die biologischen Profile der Verbindungen, die durch das morphologische Profiling (Cell Painting Assay) bestimmt wurden, mit ihren strukturellen Informationen zu verbinden, indem verschiedene Methoden der Cheminformatik eingesetzt wurden.

Im ersten Projekt wurden Machine Learning-Modelle entwickelt, um Quantitative Structure-Activity Relationship zur Vorhersage der Cell Painting-Profile der Verbindungen unter Verwendung verschiedener molekularer Darstellungen durchzuführen. Es wurden verschiedene Kombinationen von Modellierungsmethoden und Darstellungen erforscht, aber keine von ihnen hatte eine gute Vorhersagekraft. Eine Hypothese für die geringe Vorhersagekraft der Modelle ist, dass Verbindungen mit ähnlichen Strukturen keine ähnlichen Bioprofile aufweisen.

Im zweiten Projekt wurde eine generative Modellierungsmethode entwickelt, um zahlreiche Verbindungen mit ähnlichen Kurzprofilen zu erzeugen. Die endgültige Methode erzeugte keine Verbindungen mit relevanten Strukturen aus der medizinischen Chemie.

Lysosomotropismus ist ein Phänomen, bei dem kleine, lipophile Verbindungen mit basischen Anteilen in den Lysosomen gefangen werden. CPA ist ein zuverlässiges Surrogat zur Identifizierung von Verbindungen, die Lysosomotropismus zeigen, da solche Verbindungen einen ausgeprägten Phänotyp haben. Im dritten Projekt wurde berichtet, dass Verbindungen, die eine geeignete Lipophilie und Basizität aufweisen, nicht als lysosomotrop einzustufen sind. Um zu untersuchen, welche Faktoren den Lysosomotropismus und umgekehrt antreiben, wurden Matched Molecular Pair Analysis und Explainable Machine Learning an einer geeigneten Teilmenge der internen CPA-Daten durchgeführt.

Moderne ML-Methoden wie Neural Network und Ensemble-Entscheidungsbäume haben im Allgemeinen eine hohe Vorhersagekraft, leiden aber unter “black box” Natur, da ihre Vorhersagen nicht erklärt werden können. Im vierten Projekt wurde eine Open-Source-Software namens *eXplainable FingerPrints* (X-FP) entwickelt. X-FP verwendet ML-Methoden zur Merkmalszuweisung wie Shapley Additive Explanations (SHAP) im Kontext von Morgan Fingerprints (MF) und erzeugt einen intuitiven Bericht, der wichtige MF-Bits und die von ihnen kodierten Substrukturen hervorhebt.

Die Induktion ist der Prozentsatz der signifikant veränderten CP-Merkmale und wird als Maß für die Aktivität der Substanz im CPA verwendet. Die Induktionsvorhersage ist für Chemiker wünschenswert, da sie den vorhergesagten CPA aktiven Verbindungen Priorität einräumen können. Zu diesem Zweck wurde auf der bestehenden internen Plattform eine Web-App entwickelt, die einen binären Klassifikator NN verwendet.

1. Introduction

1.1. Morphological Profiling Cell Painting Assay

1.1.1. Background

Investigation of biological activities of novel chemical matter and/or potential drug candidates is of high relevance in chemical biology and drug discovery. Such investigations are usually performed with target or cell-based assays. However, such assays are generally limited to the biological activities they detect and cannot provide a holistic view of the activities induced by the compounds.

Since specific cellular processes or states can be reflected in cell morphology, perturbations of cell morphology by genes, chemical compounds or diseases can be subjected to predictive analysis.[1] Image-based morphological profiling of cells involves the measurement of cellular features, such as size, shape, staining intensities, etc. of cells perturbed by various treatments, usually in a high-throughput setting. Morphological profiling differs from conventional screening assays. In conventional screening assays, a limited number of features are quantified as their association with the biology of interest is established. In contrast, morphological profiling is unbiased. All measurable features are obtained together, providing a holistic view of a cellular state. Another advantage of obtaining information in such a holistic manner is that a single experimental campaign can be reused for multiple purposes.[2]

The Cell Painting Assay (CPA) is an image-based morphological profiling assay. It quantitatively captures changes in cell morphology induced by treatments compared to the control.[3] This assay uses six different dyes - with different excitation and emission wavelengths (except for two dyes with similar wavelengths) - to selectively stain different cell organelles and compartments. High content imaging is then performed in 5 channels corresponding to the dyes, followed by a series of analyses by different software, resulting in morphological fingerprints or Cell Painting (CP) profiles. Each CP profile contains hundreds of morphological features. The entire CPA protocol can be run in medium to high throughput. Experimental details are available in the Section 1.1.2.

One of the major challenges in chemical biology is the detection of bioactivity of newly synthesized compounds, the Mode of Action (MoA) of which is usually unknown. Mor-

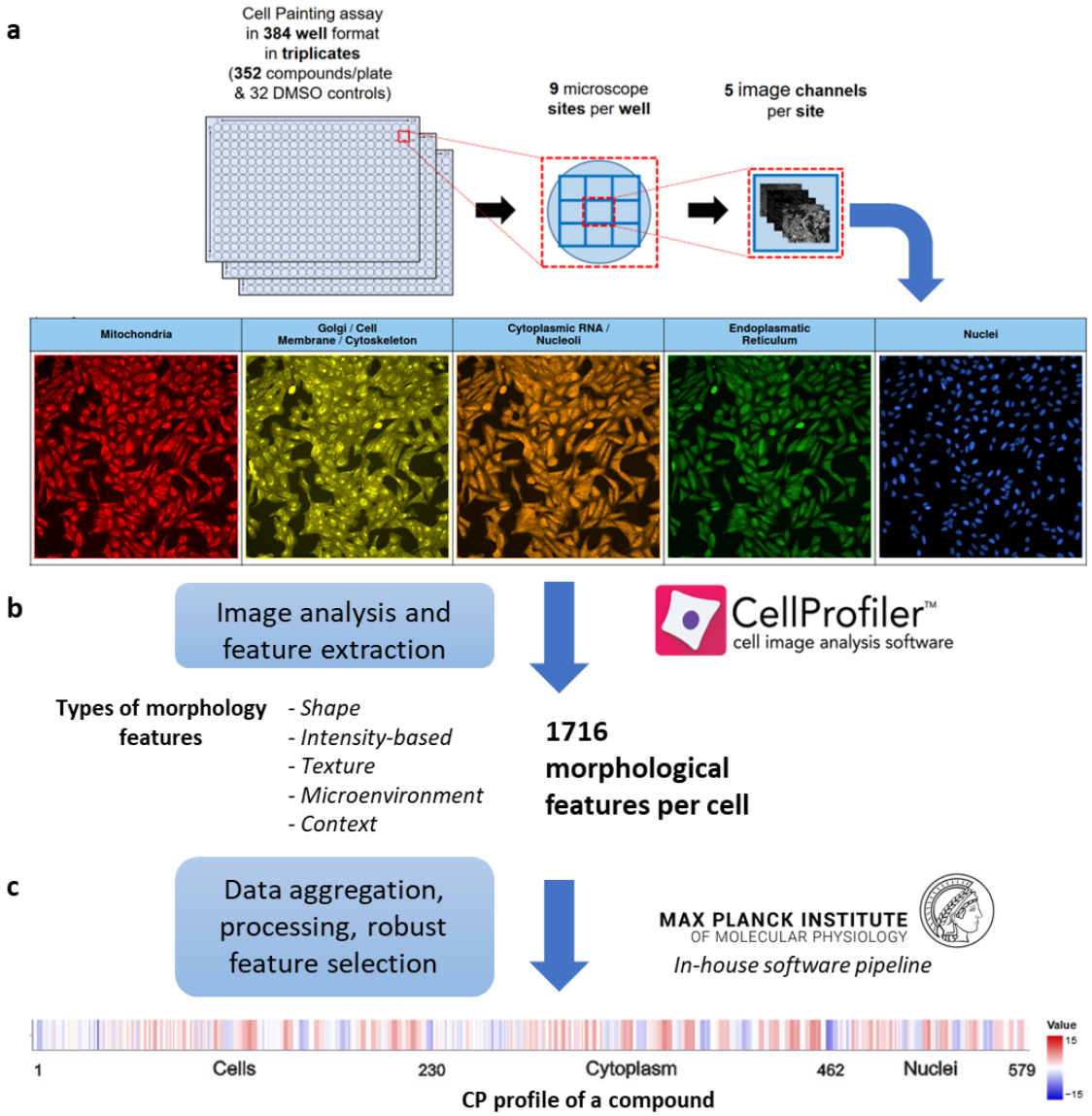
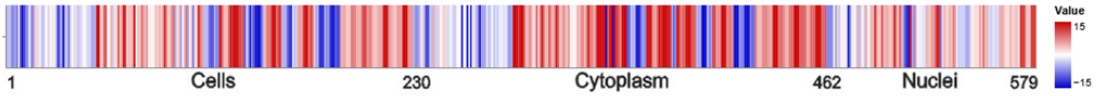


Figure 1.1.: Overview of the Cell Painting Assay (CPA) at MPI Dortmund.
a) Experimental setup - involving assay setup and image acquisition across 5 different channels.
b) Morphological profiles per cell are calculated by CellProfiler, an open-source software developed by the Broad Institute.
c) In-house software pipeline performs data aggregation and data processing. A feature vector comprising of all the *z-scores* is the CP profile of a compound. CP profiles prepared this way can be visualized as a color-coded heatmap, where each CP feature's value ranges from -15 (blue) to +15 (red).



#	CP features
1	Median_Cells_AreaShape_Area
2	Median_Cells_AreaShape_MaximumRadius
3	Median_Cells_AreaShape_MeanRadius
.	.
.	.
229	Median_Cells_Texture_Variance_Syto_5_00

#	CP features
230	Median_Cytoplasm_AreaShape_Area
234	Median_Cytoplasm_AreaShape_MinFeretDiameter
235	Median_Cytoplasm_AreaShape_MinorAxisLength
.	.
.	.
461	Median_Cytoplasm_Texture_Variance_Syto_5_00

#	CP features
462	Median_Nuclei_AreaShape_Area
463	Median_Nuclei_AreaShape_MajorAxisLength
464	Median_Nuclei_AreaShape_MaxFeretDiameter
.	.
.	.
579	Median_Nuclei_Texture_Variance_Ph_golgi_5_00

Figure 1.2.: A CP profile of a compound is made up individual morphological features. In literature, such profiles are also referred as compounds' biological fingerprints.

phological profiling methods to correctly predict MoA and toxicity of compounds with unknown bioactivity are already reported by several studies.[4] CPA quantitatively captures changes in cell morphology induced by test compounds compared to the control. It can therefore identify biological activities and targets of chemical compounds without a prior target hypothesis, in contrast to screening assays that focus on specific targets or pathways. Identification is typically done by comparing the Cell Painting (CP) profiles of test compounds with the CP profiles of well-annotated reference compounds whose modes of action and targets are known.

Apart from the application of detecting bioactivity of compounds in chemical biology, the CPA has also been applied successfully in different domains.[5] It has found application in annotating gene and allele functions,[6] as input to machine learning methods for predicting assay-specific biological activities resulting in better hit rates with high predictive performance,[7–9] predicting toxicities,[10,11] repurposing drugs for cancers,[12] etc.

1.1.2. Cell Painting Assay at MPI Dortmund

The Chemical Biology Group at the Max Planck Institute of Molecular Physiology, Dortmund, is one of the leading academic research groups in the field of CPA. CPA is used extensively to elucidate the mechanisms of action and bioactivities of new chemical matter. Table 1.1 contains a list of selected publications from the group for the years 2019 to 2023. CPA was used in these publications. The Figure 1.1 gives with an overview of the CPA at MPI Dortmund - from the experimental setup in the wet-lab to the generation of

the CP profile of in-house compound library. The Figure 1.2 shows that a CP profile of a compound is a list of different CP features aggregated in form of *z-scores*.

Table 1.1.: Research involving Cell Painting Assay at MPI Dortmund

No.	Research	Publication
1.	Pyrano-furo-pyridone PNPs as mitochondrial complex I inhibitor	Christoforow *et al.* (2019)[13]
2.	Indocinchona alkaloids PNPs as autophagy inhibitors	Foley *et al.* (2020)[14]
3.	Deferoxamine and iron chelators	Schneidewind *et al.* (2020)[15]
4.	Cholesterol homeostasis and lysosomotropism	Schneidewind *et al.* (2021)[16]
5.	Formation of pseudo-sesquiterpenoid alkaloids PNPs with diverse bioactivities	Liu *et al.* (2021)[17]
6.	PNP fragments and their connections affecting bioactivities	Grigalunas *et al.* (2021)[18]
7.	Identification of dihydroorotate dehydrogenase inhibitors	Schölermann *et al.* (2022)[19]
8.	Identification of tubulin-targeting compounds	Akbarzadeh *et al.* (2022)[20]
9.	CP subprofile analysis	Pahl *et al.* (2023)[21]
10.	Detection of mitochondrial stress phenotype	Adariani *et al.* (2023)[22]
11.	Identification of motor protein Eg5 as cellular target of spirooxindoles	Liu *et al.* (2023)[23]
12.	Novel bioactivity of chromaline PNPs	Zinken *et al.* (2023)[24]
13.	Illuminating Dark Chemical Matter (DCM)	Pahl *et al.* (2024)[25]
14.	Bioactivity of monoterpene indole alkaloids PNPs	Xie *et al.* (2023)[26]

Detection of bioactivity using Cell Painting Assay

The in-house MPI compound library of 14,837 compounds (processed dataset, as of June 2023) assayed in the CPA, can be broadly divided into three categories:
1) Reference compounds - compounds with annotated biological activity, mainly from - Prestwick Chemical Library, the Library of Pharmacologically Active Compounds, Selleckchem, MedChemExpress, Structural Genomics Consortium, etc.
2) Internal research compounds primarily consisting of Natural Products (NP)-inspired small molecules and Pseudo-Natural Products (PNP).
3) Compounds from external collaborators.

The analysis of the latest compoud library is present in the Section 1.1.2.

Compounds with similar CP profiles are expected to share the same MoA.[19] Therefore, comparing the CP profiles of a compound of unknown bioactivity with the CP profiles of compounds with annotated bioactivities, which serve as references or landmark here, can be used to generate hypotheses for the compound's target and/or MoA.[4,13]

In the Chemical Biology group, CPA is mainly used to generate hypotheses about novel chemical matter, which is usually followed by target identification methods. Due to its target-agnostic approach, CPA can cover large areas of biological space in a single experimental approach. This makes it an optimal method for evaluating the bioactivities. In addition to novel chemical matter or compounds under investigation, CPA may also detect unreported bioactivities and off-target activities for compounds with known bioactivities.

One of the compound classes of interest to the group is natural products and their derivatives. Natural products (NPs) are a great inspiration for drug design. They are the result of thousands of years of evolution to perform their functions optimally. As a result, they tend to have good bioavailability.[27] While some NPs are already fragment-sized, NPs can generally be simplified into NP-derived fragments. By combining different NP fragments, novel scaffolds are formed, which are referred to as "pseudo-natural products" (PNPs). PNPs may retain the chemical and biological relevance of NPs, but since they are not available from the current biosynthetic pathways in nature due to the unprecedented arrangements of the fragments, they may have novel or different bioactivities and targets compared to the guiding NPs.[28] While it has been reported that NPs and PNPs are commonly found in biologically relevant compounds, the vast number of NP fragments leaves a huge scope for the design of novel PNPs.[29] Figure 1.3 shows the design of PNP by combining NP-derived fragments and NP fragments, followed by analysis in CPA.

Christoforow *et al.* (2019) designed and synthesized a collection of PNPs, termed "pyrano-furo-pyridone" (PFP) PNPs, by synthetically recombining NP-derived pyridone and dihydropyran fragments. When analyzed in the CPA, these PFPs showed biosimilarity to numerous reference compounds. These reference compounds are involved in various signaling pathways involving reactive oxygen species. They confirmed that the PFPs induce the formation of reactive oxygen species and reported the PFPs as structurally novel inhibitors of mitochondrial complex I. This analysis demonstrated that similar physiological responses can be elicited by compounds with different annotated targets if the targets are involved in a common MoA.[13]

Foley *et al.* (2020) synthesized a novel class of PNP compounds called indocinchona alkaloids by fusing the cinchona alkaloid scaffold with the indole ring system. In a series of cell-based screens, one potential compound of this class, termed azaquindole-1, was found to be a potent autophagy inhibitor. When analyzed in the CPA, the CP profile of azaquindole-1 was found to be similar to a lipid kinase VPS34 inhibitor, SAR405. In fact, azaquindole-1 was found to be a potent inhibitor of VPS34 in a confirmatory screen. They found that for the azaquindole class of compounds, inhibition of VPS34 strongly correlated with autophagy inhibitory activity, and therefore they hypothesize the relationship between VPS34 and autophagy inhibition. With this study, the authors highlighted the potential of CPA as a target identification method.[14]

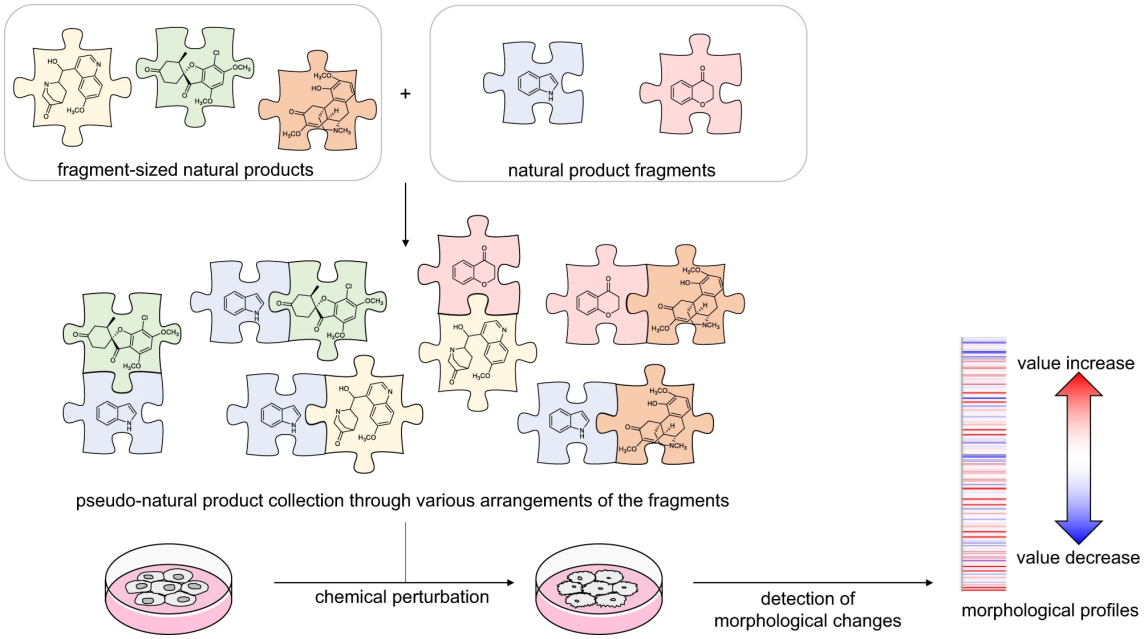


Figure 1.3.: Illustration of design of Pseudo-Natural Products by fragment combination and evaluation in the Cell Painting Assay.
Used with permission of Springer Nature BV, from "Natural product fragment combination to performance-diverse pseudo-natural products", Grigalunas *et al.* 2021, Nat Commun 12, 1883 (2021);[18] permission conveyed through Copyright Clearance Center, Inc.

Schneidewind *et al.* (2020) determined the CP profile of deferoxamine, an iron chelator used to treat iron-overload diseases. They identified a number of metal chelating agents among the reference compounds that are highly biosimilar to deferoxamine. In addition, they found that several deferoxamine-biosimilar reference compounds have different annotated cellular targets. They concluded that high biosimilarity in the CPA is exhibited by the compounds that are iron chelators and those that impair the cell cycle in the S/G2-phase by targeting DNA or S/G2 regulating proteins. This compound cluster based on the CP profiles of these reference compounds can be used to determine the MoA of unannotated compounds.[15]

Schneidewind *et al.* (2021) identified a large biocluster of compounds that alter cholesterol homeostasis. This is primarily due to the physicochemical properties of the compounds - protonation and lipophilicity. This phenomenon, termed lysosomotropism, is well described in compounds with a suitable cross-section of these physicochemical properties and is pharmaceutically relevant. Interestingly, many well-established compounds with different primary targets were also found to be present in this cluster.[16] More details about lysosomotropism are covered in Chapter 5 - Lysosomotropism and Explainable Machine Learning.

Liu *et al.* (2021) designed a collection of PNP compounds using chemical ring distortion of NPs and a complexity-to-diversity strategy. The stereochemical diversity of this collection of pseudo-sesquiterpenic alkaloids was reflected in different biological per-

formances. They found that different stereoisomers resulted in different CP profiles - indicating different bioactivities.[17]

Grigalunas *et al.* (2021) prepared a PNP library of 244 structurally diverse compounds. These compounds were a synthetic combination of the fragment-sized NPs - quinine, quinidine, sinomenine, and griseofulvin - with chromanone or indole-containing fragments. They analyzed the compounds in the CPA and made several observations. First, most of the compounds (84%) were active in the CPA. Second, the CP profiles of the PNPs appeared to be mostly due to the combination of NP fragments rather than the individual NP fragments from which they were derived. In other words, the CP profiles of PNPs are mostly different from the CP profiles of the NPs and PNPs from which they are derived. Third, the connectivity between two NP fragments also affects the bioactivity. Finally, NP fragments can be broadly categorized into 'dominating' and 'non-dominating' fragments. Dominating fragments are those whose combinations can lead to similar bioactivity, whereas non-dominating fragments are those whose combinations can lead to novel bioactivity. Therefore, CPA can be useful to drive the PNP design collection as it can categorize the bioactivities of the NP fragments, which in turn can be used to either explore novel biological space or build upon known bioactivity.[18]

Dihydroorotate Dehydrogenase (DHODH) is a rate-limiting enzyme in pyrimidine biosynthesis. It is a target of interest for several diseases, including autoimmune diseases, multiple sclerosis, rheumatoid arthritis, and viral diseases. Schölermann *et al.* (2022) identified a cluster of compounds biosimilar to brequinar, a DHODH inhibitor, in which inhibition of pyrimidine biosynthesis is a common property. They report that CPA can identify DHODH modulation at the mitochondria, which is its site of action. They also find that inhibition of purine and pyrimidine biosynthesis show similar morphological changes in the DNA but distinct changes in the rest of the cell compartments, thus CPA can be used to differentiate between inhibitors of pyrimidine and purine biosynthesis.[19]

Akbarzadeh *et al.* (2022) used nocodazole as a representative tubulin-interfering compound among the reference compounds. They found that several reference compounds for which tubulin was not reported as a target were highly biosimilar to nocodazole. In addition, they identified 15 nocodazole biosimilar compounds with different chemical scaffolds in the internal research compounds. They validated the tubulin targeting of these compounds in an independent experiment. The nocodazole biosimilarity was used to define a tubulin CP cluster, which may enable the identification of tubulin-targeting small molecules.[20]

Ideally, if the CP profiles of a novel compound and the reference compound(s) are very similar, it can be expected that both will have a similar MoA. However, this is not always the case due to numerous limitations of reference compounds - they may have incomplete annotation, polypharmacology, and unaddressed targets. To address such concerns, Pahl *et al.* (2023b) introduced subprofile analysis. First, cluster subprofiles were defined per different established bioclusters. A cluster subprofile is a subset of

CP features common to the reference compounds of that biocluster. In the next step, a median subprofile - which would define a biocluster - is calculated per biocluster. These "standard" median subprofiles can now be used for comparison with the subprofiles of the test compounds. High biosimilarity of a test compound subprofile to a standard subprofile would indicate bioactivity for that particular biocluster. If there are multiple highly biosimilar clusters, it can be inferred that the compound may exhibit polypharmacology.[21]

Adariani *et al.* (2023) found a mitochondrial stress phenotype induced by several small molecules with different annotated targets. They defined a novel CP cluster called MitoStress. Members of this cluster show such a phenotype despite having different targets because such targets indirectly suppress mitochondrial respiration and increase reactive oxygen species levels, the latter of which can lead to mitochondrial fragmentation.[22]

Liu *et al.* (2023) synthesized a novel PNP called spirooxindoles, which is a combination of oxindoles and iso-oxindoles with a spirocyclic linkage. They analyzed the bioactivity of this novel class of compounds using the CPA. These compounds did not initially show biosimilarity to any of the pre-existing bioclusters, but in a low-dimensional UMAP plot they were next to the tubulin cluster, suggesting that they could affect microtubule dynamics, which could further induce mitotic arrest. When compared to reference compounds for biosimilarity, these compounds were found to be biosimilar to mitotic kinesin Eg5 inhibitors. Upon further investigation, these novel PNPs were found to indeed inhibit Eg5, indicating that CPA can be used to detect Eg5 inhibitors.[23]

Zinken *et al.* (2023) performed a CPA-guided PNP design to further validate the findings of Grigalunas *et al.* (2020). They synthesized 39 compounds belonging to a novel PNP compound class called chromaline, which consists of the NP fragments chromane and 4H-quinoline. Both the chromane fragment and the 4H-quinoline scaffold were assumed to be non-dominating, thus the resulting chromaline PNPs might have a novel biological space. To ensure that the connectivity does not confer bioactivity, the same bridged connectivity pattern was used for all compounds. They found that indeed the chromaline compound class mostly showed unique bioactivity.[18,24]

A collection of compounds known as Dark Chemical Matter (DCM) are so named because they have been reported to be inactive in more than 100 different assays. The interesting aspect of these compounds is that they are often structurally similar to bioactive small molecules and as such do not have a separate structural scaffold class. Pahl *et al.* (2024) performed CPA on a subset of this DCM compound library to detect bioactivity. Using CPA, they were able to identify compounds in this library that are modulators of microtubule dynamics, DNA synthesis, and DHODH. They concluded that the use of unbiased CPA allowed the detection of bioactivity for various compounds in the DCM compared to target-based approaches where such compounds may not have been exposed to the correct assay. They also suggest that such bioactive compounds may exhibit less polypharmacology and are likely to be highly selective for their targets.[25]

Xie *et al.* (2023) synthesized a PNP compound library based on the fragments of an NP class, the monoterpene indole alkaloids. While 67% of the compounds in this synthesized PNP library were active in CPA at 10 μM, all of the compounds were active at 50 μM compound concentration. Most of these compounds were found to be lysosomotropic, perhaps because of their favorable physicochemical properties. However, two compounds showed biosimilarity to the DNA synthesis and tubulin bioclusters, respectively. Subsequent evaluation revealed that they were indeed inhibitors of their respective classes.[26]

Protocol

CPA protocol at MPI Dortmund closely follows the method described by Bray *et al.* (2016).[2,21]

U2OS cells are seeded in a 384-well plate containing 5 μL U2OS medium. The cell concentration is 1600 cells per well. The plate is incubated at 37° C for 4 hours. While the standard concentration is 10 μM, compound treatment of varied concentration, if necessary, is done with the Echo 520 acoustic dispenser. An incubation for 20 hours at 37° C follows. For control, DMSO of same concentration is used instead.

First, mitochondria staining takes place by 0.1 μg/μL Mito Tracker Deep Red for 30 minutes at 37° C in the dark. Cells are then fixed using 3.7% formaldehyde in PBS for 20 minutes at 37° C in the dark. The cells are washed three times with 70 μL of PBS, followed by permeabilization with 25 μL of 0.1% Triton X-100. Each well would be left with 10 μL volume.

25 μL of a staining solution mix containing 1% BSA, 5 μL/mL Phalloidin, 25 μg/mL Concanavalin A, 5 μg/mL Hoechst 33342, 1.5 μg/mL WGA-Alexa594 conjugate, and 1.5 μM SYTO 14 solution is then added to each of the well. Table 1.2 lists the dyes, the cell compartment they stain and their excitation and emission ranges. The excitation and emission ranges are taken from the ThermoFisher Scientific catalog web pages last accessed on November 21, 2023.

The plate is incubated for 30 minutes at room temperature in the dark and then washed three times with 70 μL PBS. PBS is not aspirated after the last wash. The plates are sealed and centrifuged at 500 rpm for 1 minute.

To reduce the plate effect, plates are prepared in triplicates with shifted layout. Imaging of plates is performed by Micro XL High-Content Screening System (Molecular Devices) in 5 channels (DAPI: Ex350-400/Em410-480; FITC: Ex470-500/Em510-540; Spectrum Gold: Ex520-545/Em560-585; TxRed: Ex535-585/Em600-650; Cy5: Ex605-650/Em670-715) with 9 sites per well and 20× magnification (binning 2).

1716 morphological cell features per microscope site are extracted from images using the open-source *CellProfiler* package (version 3.0.0). Data are aggregated as medians per

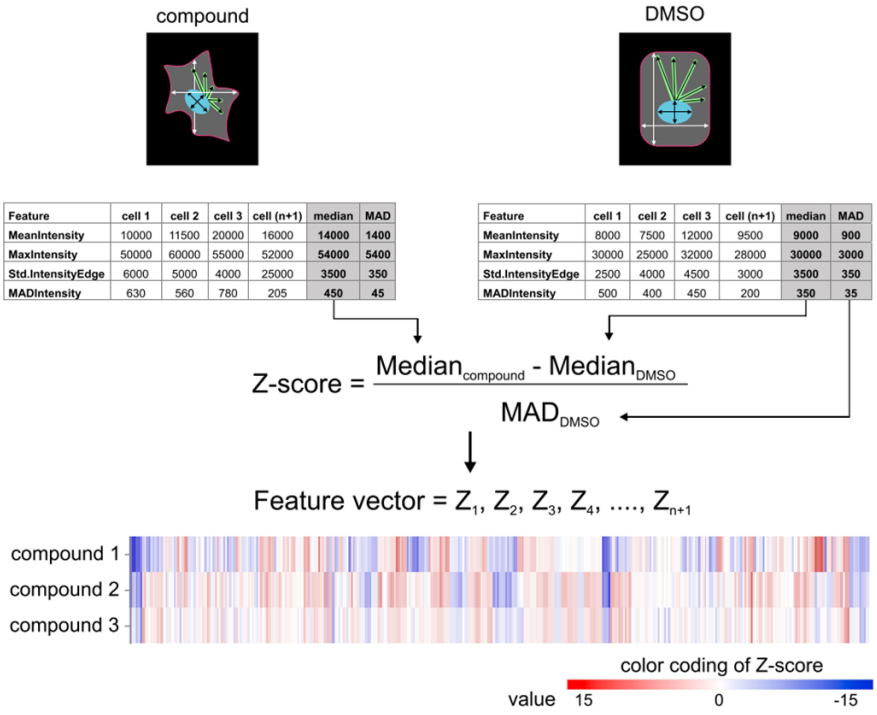


Figure 1.4.: A schematic on *z-score* calculation of each CP feature.
Reprinted from Cell Chemical Biology, Volume 28, Issue 3, Ziegler *et al.* (2021),[1] "Morphological profiling of small molecules", Pages 300-319, Copyright (2021), with permission from Elsevier.

well (9 sites $->$ 1 well), and then over the triplicates. Further analysis was performed using a custom in-house software pipeline written in Python, utilizing several data processing libraries and packages. Out of 1716 cell features, a set of robust and reproducible 579 features were determined as described in Woehrmann *et al.* (2013) and Pahl *et al.* (2023).[21,30]

The CP feature of a compound is a list of the *z-scores* of these 579 cellular features, where *z-score* is a measure of how far a data point is from a median value (Figure 1.4). *z-scores* are calculated relative to the median of the DMSO controls using the following formula:

$$\text{z-score} = \frac{\text{value}_{\text{measurement}} - \text{median}_{\text{controls}}}{\text{MAD}_{\text{controls}}}$$

where, MAD stands for Median Absolute Deviations.

Induction, a measurement of bioactivity is a percentage of significantly altered features. It is calculated using the following formula:

$$\text{Induction } [\%] = \frac{\text{number of features with absolute values} > 3}{\text{total number of features}}$$

Similarities between two CP profiles, termed biosimilarity, are calculated from the correlation distances (CD) between two profiles using the following formula:

$$CD = 1 - \frac{(u - \bar{u}) \cdot (v - \bar{v})}{\|(u - \bar{u})\|_2 \|(v - \bar{v})\|_2}$$

where $\bar{u}$ is the mean of the elements in u, $\cdot$ denotes the dot product, and $\|x\|_2$ is the Euclidean norm of x:

$$\|x\|_2 = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

This calculation is done using the Python library, *scipy* (https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.correlation.html).

The biosimilarity is then defined as:

$$\text{Biosimilarity} = 1 - CD$$

Biosimilarity values smaller than 0 are set to 0. It is expressed in percent.

Table 1.2.: List of dyes and the corresponding cell organelles

No.	Dye	Subcellular component	Wavelengths (Excitation / Emission)
1.	Hoechst 33342	Nuclei	361 nm / 497 nm
2.	Concanavalin A	Endoplasmic Reticulum	495 nm / 519 nm
3.	SYTO 14	Nucleoli	517 nm / 521 nm
4.	Phalloidin	Actin	581 nm / 609 nm
5.	Wheat germ agglutinin	Golgi	590 nm / 617 nm
6.	MitoTracker Deep Red	Mitochondria	644 nm / 665 nm

Data

In the following section, the latest available in-house CP dataset (June 26, 2023) is analyzed. The dataset was imported as a Pandas DataFrame. There were 53,052 entries. A preprocessing step was performed. This preprocessing included keeping only compounds tested at the 10 μM concentration (this was chosen as the default concentration), removing toxic compounds, removing compounds with a missing structure, removing compounds with an impurity flag, and removing compounds from the DCM campaign. If multiple measurements of the same compounds were present, the one with the highest induction was selected. The final processed dataset contained 14,837 compounds. Since this dataset consists of compounds over years, it can be visualized as shown in Figure 1.5a.

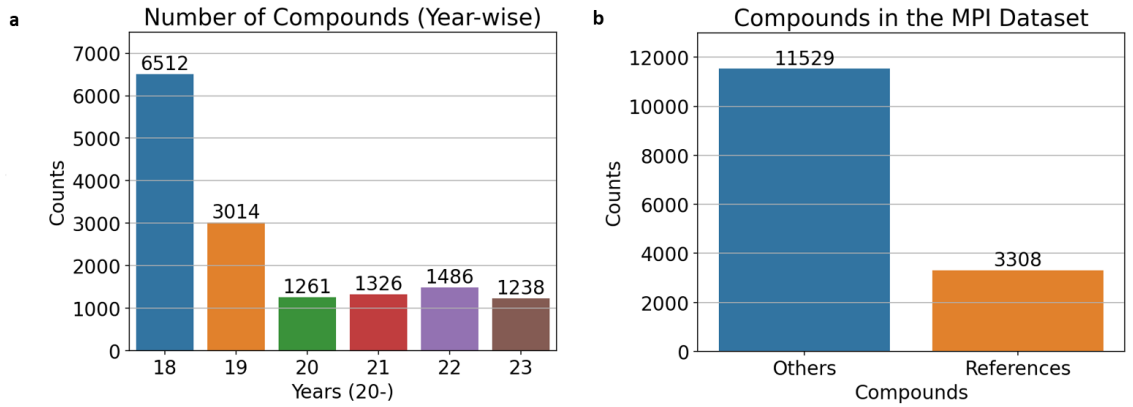


Figure 1.5.: Data distribution in the processed CP dataset.
a) Number of compounds over year.
b) Distribution of references and other compounds.

Multiple measurements of same compound happen when it is provided by different collaborators for their respective projects to be assayed in the CPA. It is also possible for the compound to be different forms, for instance, with salts.

Toxicity in the CPA is determined as follows. During the incubation stage of the CPA, cells normally divide once. When the cell count, relative to the DMSO control, is at 50%, it means that while the cells have not died yet, they are in full arrest and have stopped dividing. However, when the relative cell count goes below 50%, the compounds are marked as toxic.

The Figure 1.5b shows the number of reference compounds and the rest of the compounds in the dataset.

Compounds with an induction of 5 or more are considered active in the CPA. There are 4778 CP active and 10059 CP inactive compounds in the dataset (Figure 1.6). It should be noted that the terms "CP active" and "CP inactive" do not indicate the activities of these compounds in different assays. In other words, a compound may have a known bioactivity but be CP inactive if it does not induce sufficient morphological perturbations on cell morphology in the CPA, and *vice versa*.

Figure 1.7 shows different bioclusters and the number of compounds present per biocluster. While a compound may have different bioactivities and fall into several bioclusters, in this case a compound is assigned to a biocluster for which it shows highest proximity. This calculation is based on the subprofile analysis shown by Pahl *et al.* (2023).[21] Example reference compounds from each biocluster are shown in supplementary figures S1 - S13.

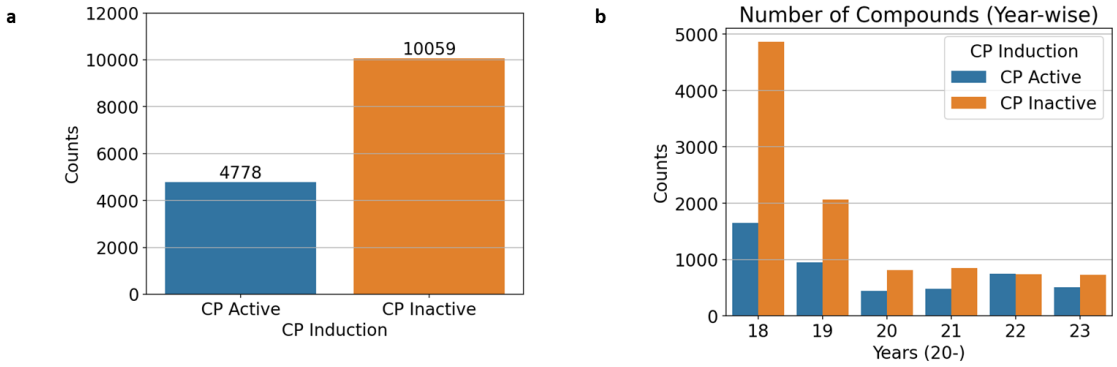


Figure 1.6.: CP Induction.
a) Compound distribution based on being CP active or inactive.
b) Yearly compound distribution binned by induction.

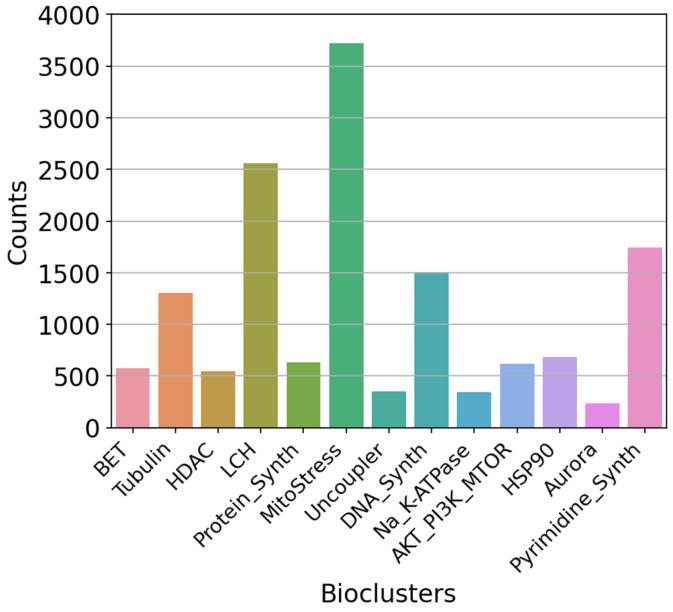


Figure 1.7.: Number of compounds per bioclusters

1.2. Cheminformatics

1.2.1. Background

Cheminformatics, as a scientific disciple, can be (simply) defined as the use of computational methods in solving chemistry problems.[31] Nowadays, it is an integrated part of the drug discovery process. While cheminformatics, as a general term, may encompass broad areas of research domains, like molecular modeling, Quantitative Structure-Activity Relationship (QSAR), quantum mechanics, etc., in this thesis it will be exclusively cover Quantitative Structure-Activity Relationship/Machine Learning methods, similarity search, Matched Molecular Pair Analysis, and calculation of physical chemical properties of compounds.

1.2.2. Machine Learning Methods

Artificial intelligence is a broad field of computer science that focuses on developing algorithms and models that can simulate human-like intelligence and perform tasks in a similar way.

Machine Learning (ML) is a subset of artificial intelligence. According to Andrew Ng, *"Machine learning can be defined as the process of inducing intelligence into a system or machine without explicit programming."* It is a multidisciplinary field of computer science, mathematics, statistics, and application domains. ML models identify the patterns and relationships in the data and make predictions based on this knowledge.

One of the earliest applications of a statistical method - a multiple linear regression - to explain a bioactivity (here, the plant-growth inhibitory activity of phenoxyacetic acids) using chemical descriptors was more than fifty years ago.[32] The goal of Quantitative Structure-Activity Relationship (QSAR) modeling is to predict the bioactivities of compounds from their chemical properties usually expressed as structural information or descriptors. QSAR is based on the principle that similar chemical structures should have similar (bio)activities. QSAR models take the information about the compound in the form of chemical descriptors or its substructures as input and predict the activity as output. Such models are typically developed using statistics (for instance, Matched Molecular Pair Analysis) or machine learning methods.

The use of ML models to perform QSAR has become a common practice in drug discovery in the last few decades, both in academia and pharmaceutical industries. In the pharmaceutical industry, the goal of ML models is to save resources and time by finding ideal drug candidates or deprioritizing unpromising ones. While in academia, successful models help to extend their applications to novel problems.[33]

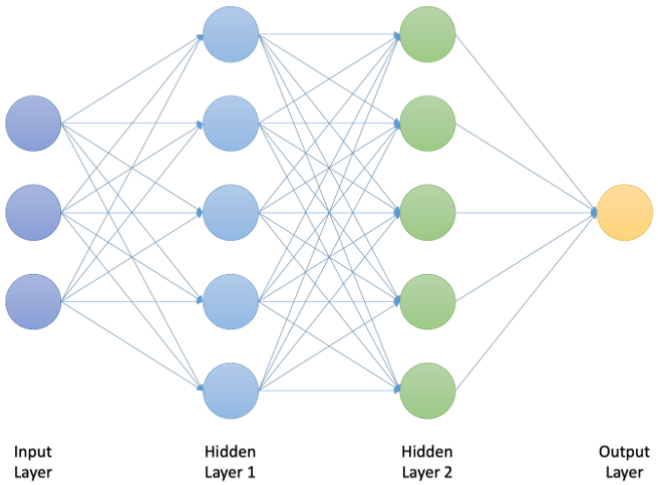


Figure 1.8.: Diagram of a Deep Neural Network architecture

Deep Neural Networks

A simplified representation of the structure of a neural network is shown in Figure 1.8. It is a hierarchical organization of neurons connected to other neurons. The term layer is used to define a group of neurons that are logically separated in this hierarchical structure. The input data is received by the first layer of neurons, called the input layer, whose output then provides input to the first hidden layer, which in turn provides input to the next layer, and so on. A hidden layer is a layer of neurons that are not connected to the input or output of the network and are responsible for learning the complex representations of the data.

Each neuron as shown in Figure 1.9, has many components. Weights (W) are scalar values assigned to each neuron. They determine the influence of each input on the output of a neuron. Biases are scalar values added to the weighted sum of the inputs before the activation function is applied. They allow neurons to shift their activation threshold up or down in response to different patterns in the data.

Activation function (ϕ) is a mathematical function that is applied to the output of a neuron to introduce non-linearity. A few of the common activation functions are:

- Sigmoid: Maps real numbers to a value between 0 and 1.

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

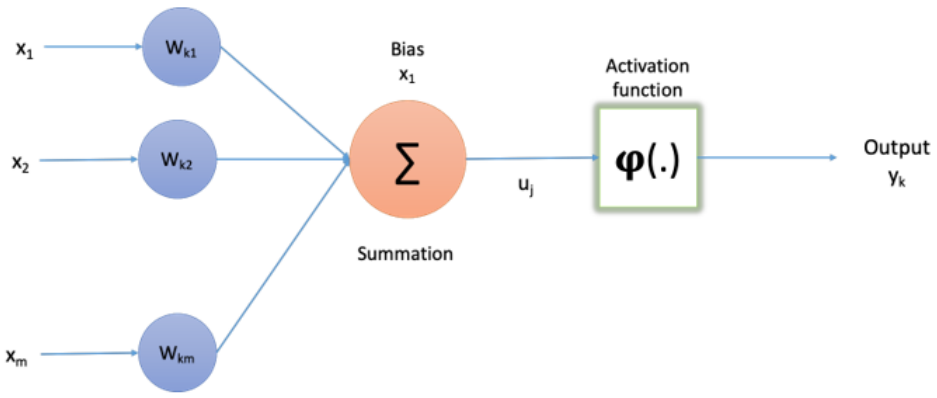


Figure 1.9.: Diagram of an individual neuron in a Neural Network architecture

- Tanh: Maps real numbers to a value between -1 and 1.

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

- Rectified Linear Unit: Maps real numbers to zero or themselves.

$$\text{ReLU}(x) = \max(0, x)$$

- Leaky ReLU: Modified version to ReLU to prevent neurons to become inactive due to ReLU function outputting zero for negative inputs.

$$\text{LeakyReLU}(x) = \max(\alpha x, x), \quad \alpha > 0$$

- Softmax: Outputs probability distribution over multiple classes.

$$\text{softmax}(x) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$

The simplified model training process is described. A feed-forward neural network takes an input (x) and produces an output ($\hat{y}$). The x provides the initial information that is then propagated through different neurons and layers resulting in $\hat{y}$. This process is called forward propagation.[34] The error between the predicted output and the ground truth is calculated based on the loss function selected by the user. This error is then used to update the weights and biases of the entire network in the opposite direction ("backpropagation"). The parameters are then updated to minimize the error.

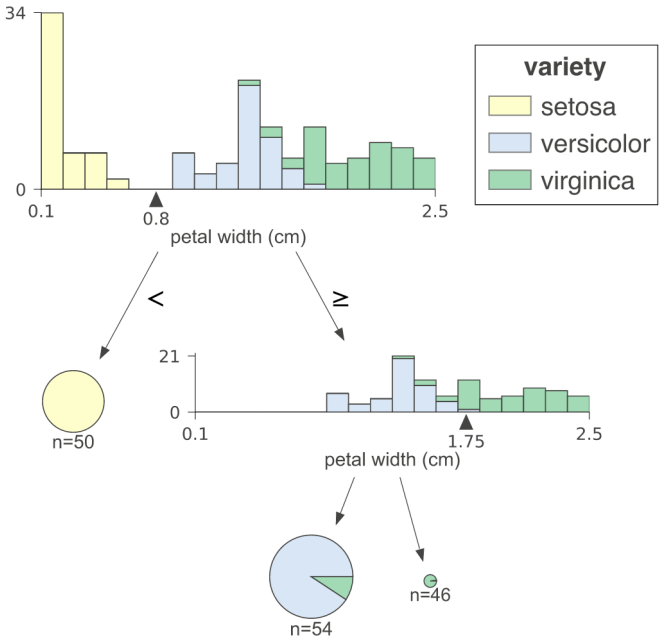


Figure 1.10.: A simple decision tree classifying species of iris plants based on samples' petal width.
Image credit: "dtreeviz" Python software package.

Tree-based Machine Learning Methods

Tree-based models use decision trees as their core. Decision trees are non-parametric, supervised ML methods that have a hierarchical tree-like structure. They consist of:

1. A root node, which is the starting point of the tree.

2. Internal nodes, which represent the decision rules based on the values of the input it receives. Each internal node has one incoming edge and several outgoing edges.

3. Branches, representing the decision rules - leading from one node to another.

4. Leaf nodes, representing the final results of the decision process.

Decision trees have the advantage of being non-parametric, i.e., they make no assumptions about the underlying data, and are very convenient to interpret and visualize. However, they are prone to overfitting, which leads to poor generalization of the data. Figure 1.10 shows an exemplary decision tree classifying iris plants.

Ensemble tree methods are a popular choice among data scientists because they reduce overfitting and generally have better predictive performance than decision trees. Two such ensemble methods are random forests and gradient boosting machines. However, due to their ensemble nature, the convenience of interpretation and visualization is lost.

1.2.3. Model inputs

In this thesis, these three types of molecular representations were explored as input to the models.

String-based Molecular Representation

Simplified Molecular Input Line Entry System (SMILES) are well-established string-based graph representations of compounds. Introduced by Weininger in 1988, SMILES are the *de facto* standard molecular representation in cheminformatics.[35]

SMILES represent structures with a minimal intuitive grammar and are human readable. While they can describe even complex chemical structures with simplicity, there are two major problems in their application in ML applications. First, a compound can be represented by several SMILES. This problem is usually solved by having a canonicalized version created by a cheminformatics software. The second problem is situational - their application in generative modeling. The intention behind the development of SMILES was to have an efficient system for storing chemical information. With the advent of generative modeling, it was observed that a high fraction of generated SMILES by such models were invalid, primarily because they were syntactically invalid - they did not correspond to a valid molecular graph, or simply violated basic rules of chemistry such as the valence rules.[36]

To address the second problem, Krenn *et al.* (2020) developed SELFIES. SELFIES are a novel string representation of molecular graphs, where each string corresponds to a valid molecule. Such molecules are valid even if the characters in a string are scrambled. This nature of SELFIES makes them an ideal choice for use in ML modeling, especially in generative molecular modeling.[37] SELFIES is present on GitHub as an open-source Python software.

Since such strings cannot be used directly as input to the ML models, in this thesis - SELFIES were converted to one/two dimension(s) one-hot-encoded format. This results in a sparse binary array. This conversion of SELFIES to one-hot encoded array requires a dictionary of SELFIES alphabets and a max SELFIES length (i.e., the alphabet count of the molecule with most SELFIES alphabet in the dataset). This one-hot encoded sparse binary array is then the product of the max SELFIES length in the dataset and the number of SELFIES alphabets in the dictionary. This conversion is bi-directional - SELFIES can be converted to one-hot encoded arrays and vice versa. Figure S14 provides with a visualization of the two-dimensional one-hot encoded SELFIES.

While this one-hot encoded array can be used as input for models, they also pose issues. First, they are sparse. For example, if the maximum SELFIES length is 100 characters, and there are 200 unique alphabets in the SELFIES dictionary; the one-hot encoded SELFIES length will be 20,000, with a maximum of 100 "1"s (meaning turned on bits). Such high-

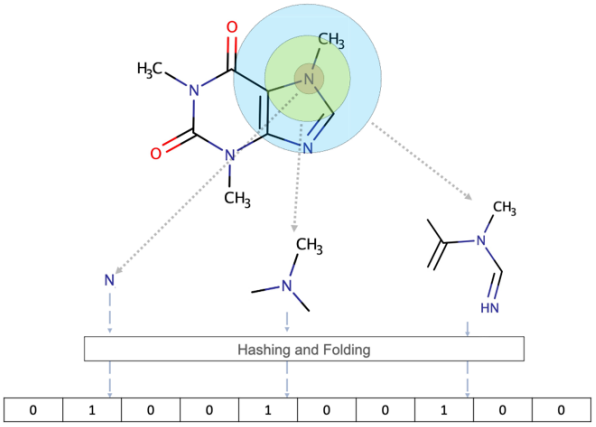


Figure 1.11.: A simplified representation of the Morgan Fingerprint as bit vector
MF are circular in nature where unique substructure environments are hashed and then folded to a fixed bit vector length. The radius is the bond length from an initial atom. Here we see 3 different substructures based on different radii (0, 1, 2) being encoded.

dimensional data is a problem when its value is equal to or greater than the total number of samples in the dataset, and such a phenomenon is referred to as the "curse of dimensionality". It can also lead to problems such as model over-complication, over-fitting, and increased computation. Second, this encoding will not work on molecules with SELFIES alphabets outside the declared array is introduced. Finally, similar to SMILES, multiple SELFIES can represent a single molecule, so canonicalization is still required.

The initial versions of SELFIES - version 1.0.x are used (https://github.com/aspuru-guzik-group/selfies/releases) for string based molecular representation as model input in the first two projects.

Morgan Fingerprints

Molecular fingerprints, such as extended-connectivity fingerprints (ECFPs), are topological fingerprints. When calculated as bit vectors, ECFPs encode the substructures into a binary format where "1" means presence of a substructure and "0" means absence. They are circular in nature and are not predefined - allowing them to encode virtually any substructure.[38]

In this thesis, the RDKit implementation of ECFP, called Morgan Fingerprints (MF) (as bits), is the most used molecular fingerprint. A simplified representation of MF is shown in Figure 1.11. Morgan Fingerprints are further discussed in details in the Chapter X-FP.

Molecular Descriptors

Molecular descriptors are useful numerical values highlighting different properties of compounds and are calculated using different theories from graph theory, physical chemistry, quantum chemistry, organic chemistry, algebraic topology, and so on.[39] In this thesis, all molecular descriptors are calculated using RDKit, unless specified.

1.2.4. Classification and Regression

Predictive machine learning models are approximations of a target function (f) from the input variables (x) to the output variable (y). Such models are trained on historical data and used for prediction on novel data whose characteristics are unknown. Depending on the nature of the target variable, predictive models can be divided into classification and regression models. When the output variable is in the form of discrete classes, such as Class A and Class B, classification models are used. On the other hand, if the output variable is a continuous range, such as all floats between 0 and 10, regression models are used.

There are cases where a regression problem can be transformed into a classification problem. Taking compound activity as an example in cheminformatics, regression modeling is done to predict the activity of a compound within a numerical range. However, with a suitable cut-off, the compounds can be divided into two labels - "active" or "inactive", thus allowing the use of classification modeling.

Depending on the number of labels to be predicted, classification modeling can be further divided into

1. **Binary classification**: Classification where a sample has one of two labels. For example, *soluble* or *insoluble*.

2. **Multi-class classification**: Classification where a sample can belong to any of several labels. For example, *soluble*, *slightly soluble*, or *insoluble*.

3. **Multi-label classification**: Classification where, in addition to multi-class classification, a sample can belong to multiple labels. For example, a sample with labels - *soluble* and *toxic*.

1.2.5. Metrics

Model metrics quantify a model's predictive power. Few of the commonly used metrics are discussed below.

Regression metrics:

- Mean Absolute Error (MAE): It is the average of all the differences between the predicted values and the actual values.

$$\text{MAE}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} |y_i - \hat{y}_i|$$

where, $\hat{y}$ is the predicted value of the ith sample, and y_i is the corresponding true value.

- Mean Squared Error (MSE): It is the average of all the squared differences between the predicted values and the actual values.

$$\text{MSE}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} (y_i - \hat{y}_i)^2$$

where, $\hat{y}$ is the predicted value of the ith sample, and y_i is the corresponding true value.

- Root Mean Squared Error (RMSE): It is the square root of the MSE.

$$\text{RMSE} = \sqrt{\text{MSE}}$$

- Coefficient of Determination (r^2): It is a measure of the *goodness of fit* of a model by representing the coefficient of how well the predicted values fit compared to the original values.

$$r^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

where, $\hat{y}$ is the predicted value of the ith sample, y_i is the corresponding true value, and $\bar{y}$ is the mean of the true values.

Classification metrics:

- Balanced Accuracy: For binary classification, it is the arithmetic mean of the true positive rate and the true negative rate.[40] It avoids inflated model performance on imbalanced data.

$$\text{Balanced Accuracy} = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right)$$

- Cohen's kappa: Originally used for measuring inter- and intra- rater reliability, by taking agreement by chance into account. It can also be expanded to be used in classification problems to understand how well a trained model performs compared to a random model.

$$\kappa = \frac{p_o - p_e}{1 - p_e}$$

where, p_o is the observed agreement between the raters. In terms of classification, it is the model accuracy. p_e is the expected agreement by chance. In terms of classification, it is the baseline accuracy. Cohen's kappa ranges from -1 (completely disagreement) to 1 (completely agreement), with 0 indicating agreement no better than chance.

2. Cell Painting Assay meets QSAR

2.1. Introduction

There are three major advantages to rapidly predicting the CP profiles of compounds using *in-silico* models. First, it bypasses the challenges of performing CPA in an academic setting; second, it generates rich structure-activity hypotheses that can help researchers; and lastly, it is time-saving.

Performing high throughput screening with the CPA, at least in an academic setting, is challenging. On a technical level, it requires a variety of expertise and human resources - cell line preparation, automation of high-throughput equipment, management and storage of chemical data, extraction of morphological features using the CellProfiler software, IT to manage high-content images and feed these images into the software pipeline to generate human-readable output.[2] Financially, the cost of operation, equipment - such as the multiplex imaging microscope, image data storage, and computing resources to perform the analyses is also not trivial. Accurate prediction of a compound's CP profile using *in-silico* models can be beneficial. It can address most of the challenges mentioned above, as the researcher would only need to prepare a virtual compound library and enter it to the model to get the prediction. Later on, to perform a confirmatory analysis, short-listed compounds from the virtual library can be sent to existing facilities performing the CPA.

CPA can be used as a preliminary assay to evaluate the bioactivities of novel synthesized compounds since the morphological changes induced by these compounds can highlight their Mode of Action (MoA). Using *in-silico* models with good predictive capabilities to predict CP profiles of potential new compounds can allow researchers to synthesize targeted compound libraries. Furthermore, the application of explainability tools to such models can allow researchers to develop hypotheses to correlate (sub)structures with bioactivities determined in the CPA.

Depending on the size of the compound library and the experimental setup, performing CP assay can also be time-consuming. *In-silico* models can allow researchers to perform virtual screening of large virtual vendor compound libraries and even of potential (and novel) compounds. This can save them time by increasing the hit-rates. Based on a model's confidence in a prediction, compounds with higher probability of target bioactivity can be prioritized.

In this project, different Machine Learning (ML) approaches to develop successful Quantitative Structure-Activity Relationship (QSAR) models of the CP profiles are investigated. The underlying assumption is that each CP feature is a predictable bioactivity that can be predicted using the structure information of compounds. In other words, each CP feature of a compounds is a dependent variable (or response variable) and the structure information of a compound is an independent variable (or predictor variable) and that there is an unknown underlying function which can map the independent to the dependent variables. Three different methods to encode the structure information - string-based molecular representations (as SELFIES), molecular fingerprints (as Morgan Fingerprints), and molecular descriptors - are explored in this project. These approaches involved combination of different ML methods, input types encoding different aspects of molecular information, and prediction methods - regression and classification.

2.2. Results

2.2.1. Modeling Results

A combination of the model inputs, model types, and predictions mentioned in the previous section were explored. These are listed in Table 2.1.

Table 2.1.: Modeling approaches Project 1

No.	Models	Abbreviation
1.	Regression - DNN - SELFIES	Reg-DNN-SELFIES
2.	Regression - DNN - MF	Reg-DNN-MF
3.	Regression - DNN - Descriptors	Reg-DNN-Desc
4.	Regression - DNN - MF + Descriptors	Reg-DNN-MF+Desc
5.	Regression - Tree - MF	Reg-Tree-MF
6.	Regression - Tree - Descriptors	Reg-Tree-Desc
7.	Regression - Tree - MF + Descriptors	Reg-Tree-MF+Desc
8.	Classification - DNN - SELFIES	Class-DNN-SELFIES
9.	Classification - DNN - MF	Class-DNN-MF
10.	Classification - DNN - MF + Descriptors	Class-DNN-MF+Desc
11.	Classification - Tree - SELFIES	Class-Tree-SELFIES
12.	Classification - Tree - MF	Class-Tree-MF

Apart from the mentioned approaches in Table 2.1, these additional regression modeling approaches were also explored (Table 2.2).

Since ML models benefit from more data available for training, the models in this project used the most recent version of the CPA dataset available at the time of their development. All versions of the dataset underwent the same processing, which included selecting compounds tested at concentrations of either 10 μM or 2 μM, excluding toxic compounds, and removing duplicate values by retaining the most active measurement.

Table 2.2.: Additional regression models

No.	Model Type	Predicting
1.	Ensemble of single feature regression models	Complete CP profile
2.	Optuna-tuned single feature regression	Single CP feature
3.	Optuna-tuned all features regression	Complete CP profile

In this project, three version of CPA dataset were used as shown in Table 2.3. After processing, the "Early 2020" version consisted of approximately 12,700 compounds, the "Late 2020" version consisted of approximately 14,000 compounds, and the "Late 2021" version consisted of 15,500 compounds.

Table 2.3.: Models and the version of CPA data used.

No.	Models	CPA Data Version
1.	Reg-DNN-SELFIES	Early 2020
2.	Reg-DNN-MF	Early 2020
3.	Reg-DNN-Desc	Late 2020
4.	Reg-DNN-MF+Desc	Late 2020
5.	Reg-Tree-MF	Early 2020
6.	Reg-Tree-Desc	Late 2020
7.	Reg-Tree-MF+Desc	Late 2020
8.	Class-DNN-SELFIES	Early 2020
9.	Class-DNN-MF	Early 2020
10.	Class-DNN-MF+Desc	Late 2020
11.	Class-Tree-SELFIES	Early 2020
12.	Class-Tree-MF	Early 2020
13.	Ensemble of single feature regression models	Early 2020
14.	Optuna-tuned single feature regression	Late 2021
15.	Optuna-tuned all features regression	Late 2021

The random train-test split method was used to validate model performances. For all models, the models were trained on a randomly split train set and validation was performed on a randomly split test set. For most models, a small randomly split validation set was also available to track model training progress over different epochs.

Table 2.4 and Table 2.5 show the model performance of all regression and classification models, respectfully.

In general, none of the models described in Table 2.1 showed satisfactory model performances. In most cases, especially in the DNN modeling strategies, the hyperparameters were selected manually. To eliminate any bias introduced in this way, an automated hyperparameter selection strategy using the Optuna framework was performed for a regression model with MF as input. However, the performance of this model on the randomly divided test set did not show any improvement, and was also poor: MAE of 2.20, RMSE of 3.67, and the r^2 of 0.20.

Table 2.4.: Regression models' performances

No.	Regression Models	MAE	RMSE	r^2
1.	Reg-DNN-SELFIES	2.18	3.75	0.1
2.	Reg-DNN-MF	2.09	3.53	0.24
3.	Reg-DNN-Desc	2.09	3.79	0.2
4.	Reg-DNN-MF+Desc	2.01	3.49	0.29
5.	Reg-Tree-MF	2.06	3.57	0.20
6.	Reg-Tree-Desc	2.24	3.91	0.24
7.	Reg-Tree-MF+Desc	2.02	3.53	0.24

Table 2.5.: Classification models' performances

No.	Classification Models	Number of Classes	Balanced Accuracy
1.	Class-DNN-SELFIES	3	0.51
2.	Class-DNN-MF	3	0.59
3.	Class-DNN-MF+Desc	3	0.60
4.	Class-Tree-SELFIES	4	0.33
5.	Class-Tree-MF	5	0.37

Another example of hyperparameter selection was done for a multi-output regression. Neural networks support multi-output prediction. This makes it possible to predict a complete CP profile. Since only a single model is used, it facilitates model training, validation and maintenance. In this case, the MFs of the compounds were provided as model input and the model was trained to predict their CP profiles. The predicted CP profiles of the test set were then compared to the ground truth for biosimilarity. The biosimilarities were then plotted as a density plot (Figure S23) that could be manually inspected. Again, poor model performance was observed as most of the predicted CP profiles showed very low biosimilarities to the ground truth.

In an ensemble regression modeling approach, XGBoost models were trained on default hyperparameters per CP feature. Predictions of every model were combined to form a predicted CP feature. Here as well, the predicted CP features were compared against the ground truth, and very low biosimilarities was observed (Figure S22).

2.2.2. Post-hoc Analysis

Given consistent poor model performances, even for the models with optimized hyperparameters, another question is raised - are CP profiles of structurally similar compounds similar across the dataset? To investigate this, a post-hoc analysis was carried out by examining the activity landscape of the CPA data.

In the words of Wassermann *et al.* (2010), "In general terms, we can define an activity landscape as any representation that integrates the analyses of the structural similarity

of and potency differences between compounds sharing the same biological activity."[41] One of the ways to represent activity landscape is via Structure-Activity Similarity (SAS) map, proposed by Shanmugasundaram and Maggiora in 2001. By systematic pairwise compound comparison of all the compounds in a dataset, SAS map compare structural and activity similarities. Here, on a 2D plane, structural similarity is present on the x-axis (low to high), and activity similarity is present on the y-axis (also, low to high). Next, each pair of compounds in the dataset is plotted given their structural and activity similarities.

Each of the four quadrants of this map represent different aspects of the activity landscape. Upper left quadrant (low structural similarity, high activity similarity) represents the scaffold hopping area, where diverse structures show similarly high or low activity. Upper right quadrant (high structural similarity, high activity similarity), is the "smooth" regions of the Structure Activity Relationship (SAR), i.e., ideal for modeling. Lower left quadrant (low structural similarity, low activity similarity) is non-descriptive and according to Wassermann *et al.*, compounds in this area are "less-interesting". Lower right quadrant (high structural similarity, low activity similarity) is the area of "activity cliffs", term first used by Maggoira. These compounds which are structurally similar however exhibit large difference in activity are reported to cause ML models to perform detrimentally.[41–43]

While in the SAS map, the activity similarity is calculated in terms of potency, in this work, the CP profile similarity of the compound pairs is used instead. Thus, this modified SAS map represents compounds' biosimilarity (CP profiles similarity) and structural similarity activity landscape. The structural similarity is calculated using RDKit's (bulk) Tanimoto similarity function using compounds represented as MF of radius 2 and 2048 bits. This map is shown in Figure 2.1

On this SAS map, most compound pairs lie on the lower left quadrant. It can be observed that most compounds are not structurally similar with each other. While bioactivity (i.e., CP profile similarity) is more spread showing that even compounds having somewhat similar bioactivities (faint light blue bins over 0.6 'Bio Similarity') are structurally diverse.

2.3. Methods

Because of the exploratory nature of this project, many models with different hyperparameters and data settings were developed and tested. In the subsection below, exemplary models from each class are discussed. Except for the ensemble models, where all CP features were predicted, in these exemplary models, the first CP feature *Median_Cells_AreaShape_Area* (as the 'Y' variable) is predicted.

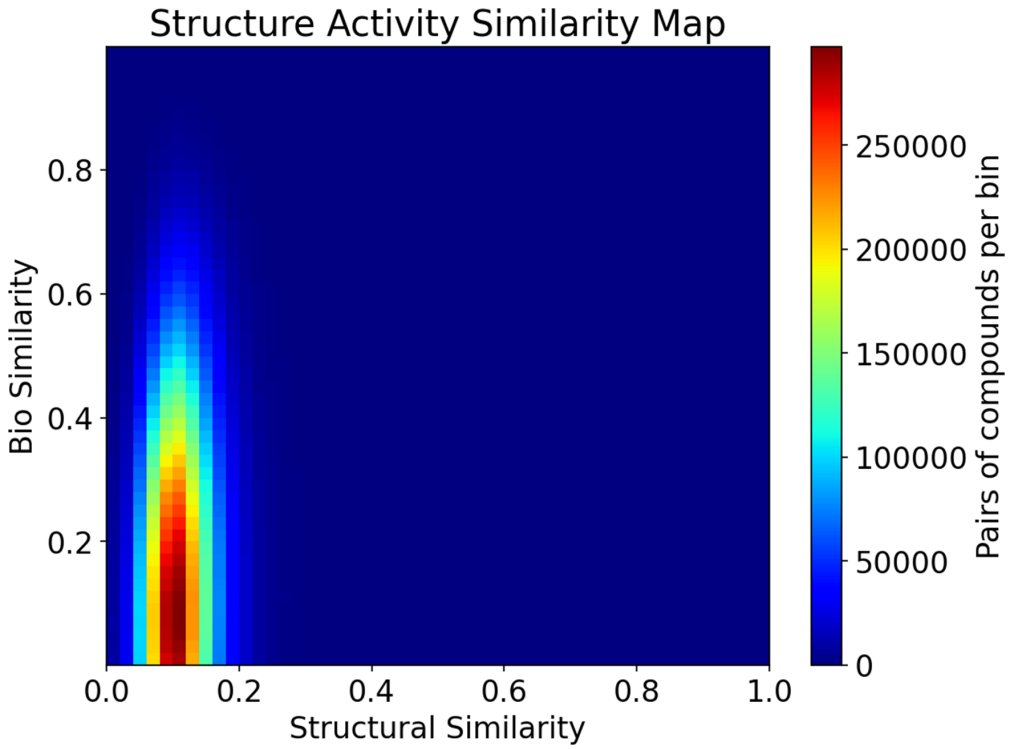


Figure 2.1.: SAS map of compound pairs in the "Early 2020" CPA dataset. There are 50 bins present on each axis.

2.3.1. Regression Models

Regression - DNN - SELFIES

After the initial filtering of the CPA dataset from Early 2020, the compounds with the maximum SELFIES length till 100 were short-listed. The total number of unique SELFIES alphabet in the dataset was 120. Therefore, the 1D-OHE resulted in the shape (12575, 12000), where 12575 is the number of compounds in the processed dataset and 12000 is the 1D-OHE length.

The dataset was first split into 70%-30% random train-test split. The training set was further split into 90%-10% train-validation split. The validation set was used to monitor the model training and to perform early stopping to prevent overfitting. As a result, there were 7921 compounds in the training set, 3773 in the test set, and 881 in the validation set.

Figure S15 shows the architecture of the developed model. Rectified Linear Unit (ReLU) is used as the activation function for each hidden layer, the last layer has a linear activation, ADAM is used as the optimizer, and MSE is used as the loss function.

The batch size was set to 256. The model was trained for 6 epochs (out of 20) until an early stop was triggered.

Regression - DNN - MF

Using Morgan Fingerprints (MF) as input, very similar models to the Reg-DNN-SELFIES were developed. An example model is discussed below.

The differences in the model architecture were that the input layer is of size (12745, 4096) followed by a dense layer of the same dimension. Here, the processed dataset from Early 2020 had 12745 compounds because it did not undergo the final filtering of removal of compounds with SELFIES length greater than 100. Next, MF of radius 3 with 4096 bits were generated. The model architecture is shown in Figure S16.

The dataset was first split into 70%-30% random train-test split. The training set was further split into 90%-10% train-validation split. This resulted in 8028 compounds in the training set, 3824 in the test set, and 893 in the validation set.

The batch size was set to 256. Here too, the model was trained for 6 out of 20 epochs until an early stop was triggered.

Regression - DNN - Descriptors

An example of a model which takes molecular descriptors from RDKit as input is discussed.

As shown in Baltruschat and Czodrowski (2020), 196 of the 200 descriptors were calculated by RDKit.[44] This was done because the unused descriptors ('MaxPartialCharge', 'MinPartialCharge', 'MaxAbsPartialCharge', and 'MinAbsPartialCharge') computed 'NaN' (*not a number*) for many compounds in the CP dataset. Once these descriptors were computed for all the compounds, the values were 'MinMax' scaled using the *scikit-learn* library. These scaled values became the model input.

The processed CP dataset from Late 2020 had 14235 compounds. The dataset was first split into 70%-30% random train-test split. The training set was further split into 90%-10% train-validation split. This resulted in the train set of size 8967, test set of 4271, and validation set of 997.

Figure S17 shows the architecture of the developed model. ReLU is used as the activation function for each hidden layer except for the last layer, which has a linear activation. ADAM is used as the optimizer and MSE is used as the loss function.

The batch size was set to 128. The model trained for 35 out of 50 epochs when an early stop was triggered.

Regression - DNN - MF + Descriptors

A similar model to the previously described Reg-DNN-Desc model was developed using both molecular fingerprints and the descriptors as input on the Late 2020 dataset.

Apart from the inputs - which were concatenated, resulting in an input dimension of 2244 (MF radius 3, 2048 bits) - the only other difference was in the shapes of the first three hidden layers (2244, 2000, and 500). The model architecture is shown in Figure S18.

This model trained for 23 out of 50 epochs when an early stop was triggered.

Regression - Tree - MF

An XGBoost Regression model with default hyperparameters was prepared on the processed CP dataset with 12745 compounds (Early 2020). The 80%-20% random train-test split resulted in 10196 compounds in the train set while 2549 compounds in the test set, and MF were used as model input (radius 3, 2048 bits).

Regression - Tree - Descriptors

196 RDKit descriptors were computed for the processed CP dataset of 14235 compounds (Late 2020).The descriptors were not scaled. This dataset was subjected to 70%-30% random train-test split resulting in 9964 compounds in the train set while 4271 compounds in the test set.

An XGBoost regression model with default hyperparameters was trained and then tested.

Regression - Tree - MF+Descriptors

The processed CP dataset, MF preparation (radius 3, 2048 bits), RDKit descriptors computation including 'MinMax' scaling, is identical to the Reg-DNN-MF+Desc model. The only differences are: the processed Late 2020 CP dataset is not further split into validation set, and the XGBoost regressor is used for modeling. The size of the train set was 9964 compounds and the test set was 4271 compounds.

2.3.2. Classification Models

As highlighted in Section 1.2.4, by assigning cut-offs in the numeric range, it is possible to convert a regression problem to a classification problem. Different cut-offs and modeling approaches were investigated.

Classification - DNN - SELFIES

Here is an example of a model that takes SELFIES as input to predict a CP feature *Median_Cells_AreaShape_Area* that has been transformed into 3 classes.

Processed Early 2020 CP dataset with 12731 compounds was used. 1D-OHE were prepared as model inputs. The CP feature *Median_Cells_AreaShape_Area* was selected - the values between -3 and 3 were labeled as 'Unchanged', the values below -3 were labeled as 'Low', and the values above 3 were labeled as 'High'. Compounds per class: 'Unchanged' - 10642, 'Low' - 1694, 'High' - 395.

The dataset was first split into 70%-30% stratified random train-test split. This stratification was done to ensure a similar distribution of class labels. The train set was further split into 90%-10% train-validation split. As a result, the train set had 8019, the test set had 3820, and the validation set had 892 compounds.

Class weights were calculated for each class so that they could be provided to the model as an argument while calling *model.fit()*. By providing class weights to the model while training it, the model factored the under-represented classes by weighing the loss function.

The model architecture is shown in Figure S19. In addition to a dropout layer, batch normalization is introduced after each hidden layer to prevent overfitting. Rectified Linear Unit (ReLU) is used as the activation function for each hidden layer, the last layer has a softmax activation, ADAM is used as the optimizer, and 'categorical crossentropy' is used as the loss function.

The batch size was set to 64. The model trained for 5 out of 15 epochs when an early stop was triggered.

Classification - DNN - MF

An identical model was trained with MF (radius 3, 2048 bits) as input with similar hyperparameters, steps and data as described in Class-DNN-SELFIES. The model architecture is shown in Figure S20. This model trained for 4 epochs when an early stop was triggered.

Classification - DNN - MF+Descriptors

This exemplary model took a combination of MF and molecular descriptors as input. For the CP feature *Median_Cells_AreaShape_Area* - the values between -3 and 3 were labeled as 'Unchanged', the values below -3 were labeled as 'Low', and the values above 3 were labeled as 'High'. Late 2020 CP dataset was used. Compounds per class: 'Unchanged' - 11709, 'Low' - 2082, 'High' - 444.

The dataset was first split into 60%-40% stratified random train-test split. This stratification was done to ensure a similar distribution of class labels. The train set was further split into 90%-10% train-validation split. As a result, the train set had 7686, the test set had 5694, and the validation set had 855 compounds.

The model architecture is shown in Figure S21. The class weights were provided during model training. The batch size was set to 64. The model trained for 5 out of 50 epochs when an early stop was triggered.

Classification - Tree - SELFIES

In this example model of this class, a processed CP dataset with 12731 compounds (Early 2020) was used. 1D-OHE SELFIES were created as model input. The CP feature *Median_Cells_AreaShape_Area* was categorized in 4 classes. Unique to this example, values between 0.5 and -0.5 were labeled 'Unchanged', values between -5 and up to -0.5 were labeled 'Low', values -5 and below were labeled 'VeryLow', and values 5 or above were labeled 'High'. There were 1022 compounds in 'VeryLow', 5084 compounds in 'Low', 3365 compounds in 'Unchanged', and 3260 compounds in 'High'.

This dataset was subjected to 80%-20% stratified random train-test split resulting in 10184 compounds in the train set while 2547 compounds in the test set with equal class distributions.

Random Forest Classifier with default hyperparameters except "class_weight = balanced_subsample" argument to account for class imbalance was trained on the train set.

Classification - Tree - MF

For this category, models using MF as input were developed. In this example, a processed CP dataset with 12745 compounds (Early 2020) was used with MF (radius 4, 2048 bits) as input. The CP feature *Median_Cells_AreaShape_Area* was selected and categorized in 5 classes. Values between -0.5 and 0.5 were labeled 'Unchanged', between -3 and -0.5 or less as 'Low', -3 or less as 'VeryLow', between 0.5 and 3 as 'High', and 3 or more as 'VeryHigh'.

There were 1694 compounds in 'VeryLow', 4412 compounds in 'Low', 3368 compounds in 'Unchanged', 2875 compounds in 'High', and 396 compounds in 'VeryHigh'. This dataset was subjected to 80%-20% stratified random train-test split resulting in 10196 compounds in the train set while 2549 compounds in the test set with equal class distributions.

A random forest classifier with default hyperparameters was trained on the train set. To account for unbalanced class weights, the 'class_weight' argument was passed with the computed class weights.

2.3.3. Other Models

Optuna-tuned Single Feature Regression Models

In this example, an automated hyperparameter exploration was performed with Optuna. The list of hyperparameters explored are shown in Table 2.6.

Table 2.6.: Parameters provided for hyperparameter tuning for Optuna for single feature regression

Parameters	Range - Lower Limit	Range - Upper Limit	Range Type
MF radii	1	4	Integer
Hidden layers	1	5	Integer
Epochs	1	10	Integer
Units per layer	10	2000	Integer (log)
Learning rate	1e-5	1e-1	Float (log)
Batch size	32	256	Integer

The processed CP dataset from Late 2021 had 15518 compounds. It was first split into 70%-30% random train-test split. The test set was further split into 80%-20% test-validation sets split. This resulted in train set with 10862, test set with 3724, and validation set with 932 compounds.

Optuna optimization was carried out for 500 trials. Models were trained on train set, and validated and optimized on validation set. The best trial had the parameters shown in Table 2.7.

Table 2.7.: Best hyperparameters found by Optuna for single feature regression

Parameters	Best Found
MF Radius	1
Hidden layers	5
Epochs	10
Units (Layer 1)	210
Units (Layer 2)	15
Units (Layer 3)	20
Units (Layer 4)	1314
Units (Layer 5)	24
Learning rate	0.0016
Batch size	97

Multi-Output Regression

In an example shown here, the entire CP features of compounds were subjected to multi-output regression.

The processed CP dataset (Late 2021) had 13131 compounds. An additional step in processing was done which involved removing compounds with purity flags. This additional data processing step while novel and unique in this project, was implemented in future projects.

The data was first split into 70%-30% random train-test split. The test set was further split into 80%-20% test-validation sets split. This resulted in train set with 9191, test set with 3152, and validation set with 788 compounds.

Table 2.8.: Parameters provided for hyperparameter tuning for Optuna for multi-output regression

Parameters	Range - Lower Limit	Range - Upper Limit	Range Type
MF radii	1	4	Integer
Hidden layers	1	5	Integer
Epochs	5	20	Integer
Units per layer	100	2000	Integer (log)
Learning rate	1e-5	1e-1	Float (log)
Batch size	32	256	Integer

The loss function of the model was mean squared error. The Optuna study of 1000 trials was performed. The best found hyperparameters are shown in Table 2.9

Table 2.9.: Best hyperparameters found by Optuna for multi-output regression

Parameters	Best Found
MF Radius	4
Hidden layers	2
Epochs	6
Units (Layer 1)	362
Units (Layer 2)	236
Learning rate	0.004
Batch size	156

Ensemble of XGBoost Regression Models

Early 2020 CPA dataset was used in this example of ensemble approach. 1000 compounds were randomly sampled from the processed CP dataset. These samples were used as the test set. A XGBoost regression model, with default hyperparameters, per CP feature were trained and saved on the memory.

2.4. Limitations

This project was exploratory in nature, exploring combinations of different modeling techniques (NN and tree-based), methods (classification or regression), and inputs. Since

all models were developed and tested independently, and given the stochastic nature of ML models, the randomness of data splitting, and different Y variables, it is not possible to systematically compare similar methods (for instance, which combination works best to perform regression).

Here are a few suggestions for improvement if similar analyses were to be performed again:

First, instead of performing numerous analyses per method independently, perform one analysis for all methods together. This would make the project systematic and the software versions and data used consistent. More importantly, it would allow both inter- and intra-comparison of models with different settings and data sets. For example, performing a single study of different model combinations using the same dataset to predict one thing. Second, to improve reproducibility, in addition to the previous suggestion, would be to save each model and figure. Assigning a random state while performing data splitting also improves data reproducibility. Third, use a variety of metrics to evaluate model performance. Finally, use advanced model validation methods such as time-split, whenever possible.

2.5. Conclusion

In this project, various QSAR modeling methods were performed for the prediction of CP profiles of compounds from their chemical structure information. The common conclusion however was that none of the methods showed reasonable performances. It can be reasoned that the underlying principle of the QSAR that "similar chemical structures should have similar bioactivities" is not followed by the CP profiles.

Similar behavior is observed in several studies. Grigalunas *et al.* (2021) and Zinken *et al.* (2023) discuss the concept of "dominating" and "non-dominating" NP fragments, where non-dominating fragments do not show similar bioactivities in their derived PNPs.[18,24] Similarly, differences in CP profiles due to stereoisomers are also reported.[1] Ziegler *et al.* (2021) also mention that structurally diverse data sets may not lead to diversity in bioactivity, as morphological profiles (in this case the CP profiles) do not correlate highly with chemical similarity.[1]

In the post-hoc analysis via SAS map, the structure-(bio)activity landscape of the CPA dataset (Early 2020) was created. Most compounds were found in the “less-interesting” region. The reason of compounds falling in this region can be hypothesized from the dataset composition point of view. In a pharmaceutical setting (at least project-wise), structural analogues are designed as potential drug targets by chemists. This is not the case with the CPA dataset. The reference compounds in the CPA dataset, are present as the representatives of their bio class, rather than structure analogues. Similarly, the internal research compounds, comprising primarily of PNP and NP inspired small com-

pounds, show varied bioactivities. It is unclear on how CPA inactive compounds affect the SAS map analysis since induction is not a parameter to measure biosimilarity between compounds. Nevertheless, this preliminary analysis using activity landscape invites further investigation to study the Structure Activity Relationship (SAR) of the internal CPA dataset.

Coming back to in-silico methods to predict CPA profiles, novel methods such as contrastive learning methods (e.g., CLOOME), where cell images from CPA can be encoded and then mapped together with chemical structure information, may provide an alternative to predicting CP profiles using chemical structure information.[45]

3. Cell Painting Assay and Generative Modeling

3.1. Aim and Motivation

In conventional drug discovery, the usual approach for identifying biologically relevant molecules is to perform a series of top-down screening methods on libraries containing millions of molecules. Such methods are virtual, chemical, and biological in nature. In contrast to this approach is the *de novo* approach. In this approach, new molecules with the desired biological or chemical properties are generated. Some of the benchmarked approaches for *de novo* molecular design are Graph Genetic Algorithms, Variational Autoencoder (VAE), Generative Adversarial Networks (GAN), Recurrent Neural Network (RNN), etc.[46]

In a 2020 preprint, Méndez-Lucio *et al.* (2020) used a generative modeling approach with the combination of GAN and the CP profiles to generate molecules that induce specific phenotypes.[47] Using this approach, they generated a virtual compound library with a large number of molecules that could induce the phenotypes of overexpression of certain genes in the U2OS cell line. About half of these generated molecules were valid. Resulting half of these valid molecules corresponded to unique molecules, and further half of these valid and unique molecules (12% of the generated molecules) had a reasonable synthetic accessibility score.

To validate whether the generated molecules had the same phenotype, they performed chemical structure similarity of the generated molecules with the known agonists of these gene targets and found that the generated molecules shared similar chemical groups with the known agonist molecules. These agonists were not present in the training set.

This preprint was one of the first works to combine CPA with the de novo molecule generation. As CPA is used extensively at the MPI Dortmund, the generation of molecules with the desired phenotype, for example of a biological class under investigation, would have proven to be an academic asset. However, as the code used in the preprint was not open source and the software frameworks they used were old, it was necessary to develop the mentioned code from scratch. Hence, in this project, an implementation of this modeling approach, with updated software packages, was developed for the internal

MPI data.

3.2. Background

3.2.1. Autoencoders (AE) and Variational Autoencoders (VAE)

According to Goodfellow *et al.* (2016), *"An autoencoder is a neural network that is trained to attempt to copy its input to its output."*[34] The Autoencoder (AE) consists of three main parts - an encoder - which takes a high dimensional input and reduces it to a lower dimensional representation, a decoder - which takes this lower dimensional representation and reconstructs the original input, and a latent space - which is the bottleneck layer, the minimum dimension to which data can be compressed. When training an AE, the goal is to reconstruct the input. During the model training process, the encoder learns how to successfully compress information to the limited size bottleneck, and the decoder learns how to pick up this limited information and reconstruct the original input. An illustration of an AE is shown in Figure 3.1.

The encoder function $g()$ is parameterized by ϕ, and the decoder function f is parameterized by θ. The low-dimensional representation for the input x learned in the bottleneck/latent layer z is (eq. 3.1):

$$z = g_\phi(x) \tag{3.1}$$

Similarly, the reconstructed input x' is (eq. 3.2):

$$x' = f_\theta(g_\phi(x)) \tag{3.2}$$

During the model training, the parameters (θ, ϕ) are learned together to output reconstructed output. The reconstruction loss, measured by Mean Squared Error (MSE) and/or cross-entropy, is typically used as a model loss metric to measure how similar or close the reconstructed output is to the input.

Since autoencoders are primarily dimensional reduction neural networks,[49] they have limited applications in generative modeling. Variational Autoencoder (VAE), first described by Kingma and Welling, are similar to AE but differ significantly both mathematically and functionally.[50] The encoder and decoder in the VAE are probabilistic in nature. Unlike AE, where the encoder maps inputs into fixed vectors on the latent space, in the VAE the probabilistic encoder maps inputs into distributions on the latent space. In the absence of a fixed representation, the probabilistic decoder samples from the distributions and can generate multiple similar reconstructed inputs. This construction of

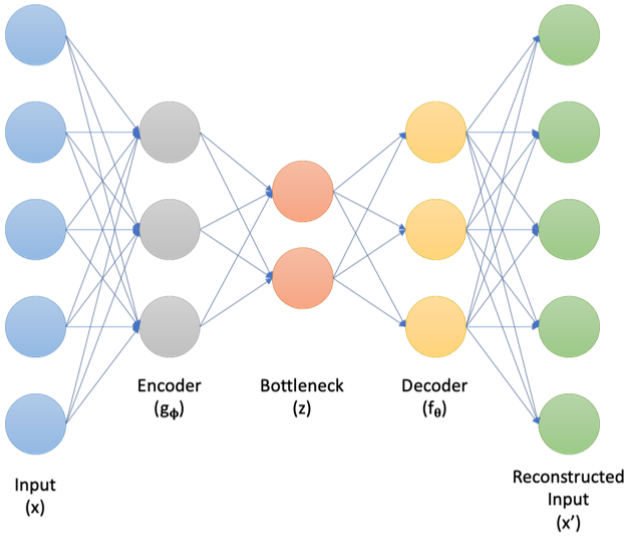


Figure 3.1.: Diagram of an Autoencoder architecture. In the case of a string based chemical representation, x can be an input SMILES/SELFIES of a molecule, while x' should ideally be the same as it is its reconstruction. Diagram inspired by Lilian Weng.[48]

novel outputs is the reason why VAE are generative models.

Equations can be modified for the VAE. The probabilistic encoder function $q()$ is parameterized by ϕ and can be written as $q_\phi(z|x)$, and the probabilistic decoder function $p()$ is parameterized by θ and can be written as $p_\theta(x|z)$.

However, this generative feature poses a problem for backpropagation. Sampling is a stochastic process, as the samples are generated from $z \sim q_\phi(z|x)$. Therefore, backpropagating would not compute an estimate of the derivative, and the model cannot be trained. This problem is solved by a technique called as the reparameterization trick. It is shown in Figure 3.2.

Here, the random latent variable z, can be instead expressed as a deterministic variable. This is done by introducing a new variable ϵ, which is a sample from a standard normal distribution (mean = 0, standard deviation = 1).

The regular definition of z:

$$z \sim \mathcal{N}(\mu, \sigma)$$

In reparameterization trick, z can be defined as (eqs. 3.3, 3.4):

$$z = \mu + \sigma \cdot \epsilon \tag{3.3}$$

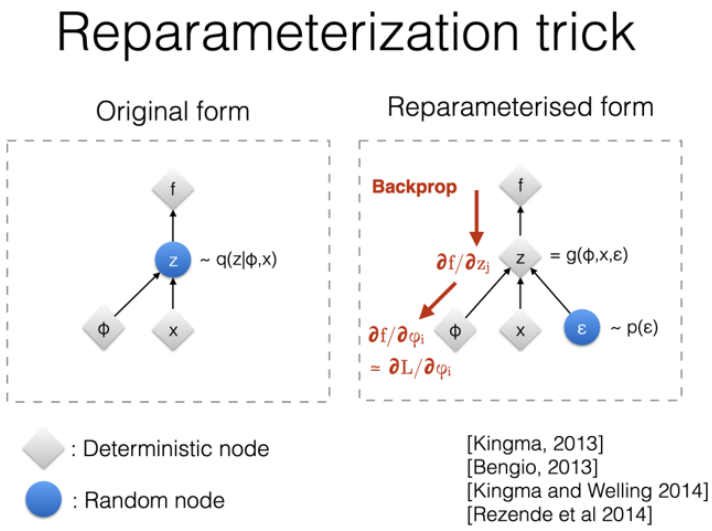


Figure 3.2.: Reparameterization Trick. Image source: Kingma's NIPS 2015 workshop talk (slide 12)

Therefore,

$$z = g(\mu, \sigma, \epsilon) \tag{3.4}$$

where,

$$\epsilon \sim \mathcal{N}(0, 1)$$

In VAE, the loss function, termed as Evidence Lower Bound (ELBO), is a sum of two parts - the reconstruction loss, which measures how well the decoder is able to reconstruct the original input data from the latent representation, and the Kullback-Leibler (KL) divergence loss, which measures how closely the distribution of the latent variables matches a standard normal distribution - where the mean is 0 and the standard deviation is 1. The KL divergence loss encourages the latent variables to follow a standard normal distribution. This is useful for generative purposes since optimizing this loss encourages the latent variables with similar output to be grouped together, and all points in the latent space to correspond to a plausible output.

3.2.2. Generative Adversarial Networks (GAN) and Conditional Generative Adversarial Networks (cGAN)

Generative Adversarial Networks (GAN) are a class of generative machine learning frameworks. GANs consist of two models that operate in an adversarial fashion. Here, a generative model G competes with a discriminative model D. The generative model (hereafter referred to as the "generator") is trained to capture the real data distribution and then synthesize samples from that distribution, while the discriminative model (hereafter referred to as the "discriminator") is trained to determine whether a sample is from the generator or from the data distribution.[51,52]

The training of GAN is a zero-sum game where both models - the generator and the discriminator - train to compete with each other. The discriminator D is trained to maximize the probability of assigning the correct labels to the real samples, i.e. the training data x, and to the samples from the generator. A prior input noise variable $p_{\mathrm{z}}(z)$ is defined, over which the generator G maps the data space as $G(\mathrm{z};\theta_g)$, where θ_g are the trainable parameters of the generator. The generator G is trained to minimize $\log(1 - D(G(z)))$.

This min-max game with value function $V(G, D)$ is shown by (eq. 3.5):

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_{\mathrm{z}}(z)}[\log(1 - D(G(z)))] \tag{3.5}$$

In GAN, there is no control over the modes of the generated data. Mirza and Osindero (2014) showed that it is possible to control the data generation process by conditioning the GAN. Such conditional models, called Conditional Generative Adversarial Networks (cGAN), are created by conditioning both the generator and the discriminator on some auxiliary information, such as class labels, y.[53]

The equation of GAN can therefore be modified accordingly (eq. 3.6):

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x|y)] + \mathbb{E}_{z\sim p_{\mathrm{z}}(z)}[\log(1 - D(G(z|y)))] \tag{3.6}$$

The difference between the training and inference mode of GAN and cGAN is shown in Figure 3.3.

A simplified example to differentiate the functionality of GAN and cGAN is as follows. If a GAN is trained on an image dataset of different types of fruits, it would generate a fruit image of random type every time it is inferred. On the other hand, if a cGAN is trained on the same dataset, but with the labels of the fruits, every time it is inferred along with the query class (i.e., the type of fruit), it would generate the fruit image of the type queried.

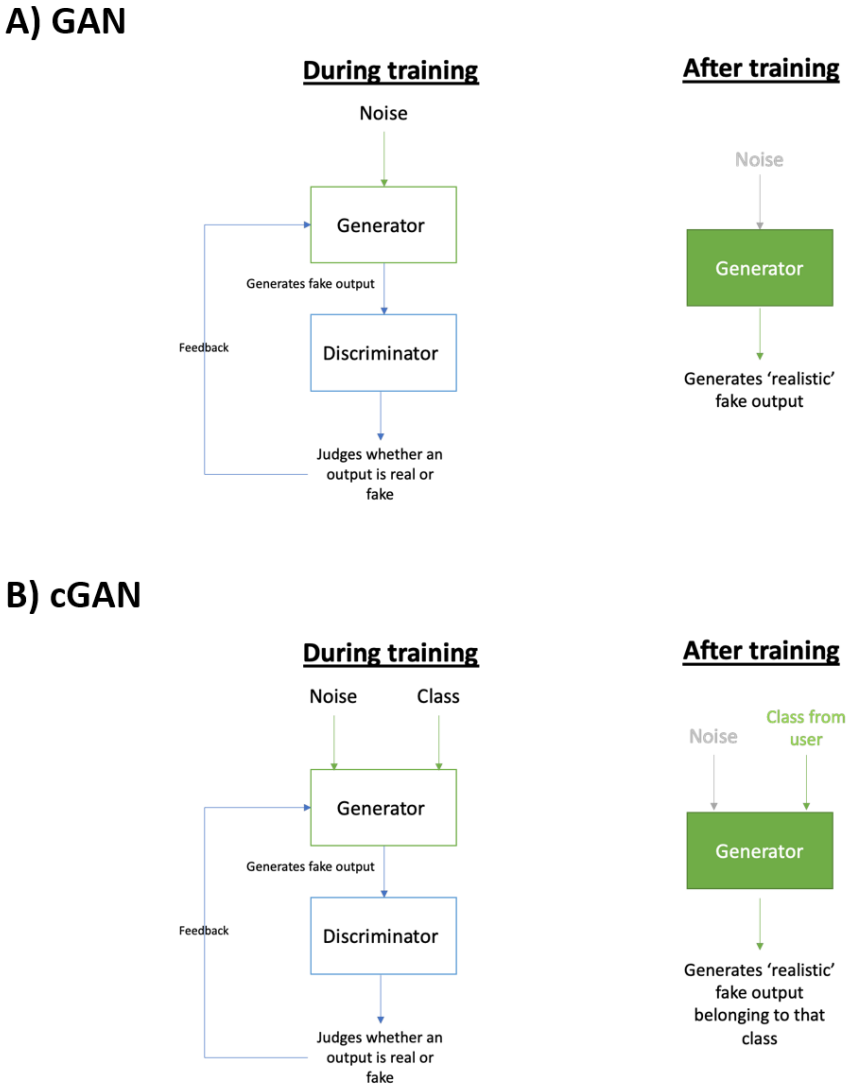


Figure 3.3.: Simplified diagram showing training and inference of A) GAN, and B) cGAN.

3.2.3. Generative Modeling by Méndez-Lucio *et al.* (2020)

The generative model of Méndez-Lucio *et al.* (2020) was a combination of a VAE trained on approximately 1.25 million molecules extracted from CHEMBL 22, and a cGAN trained on 126,779 CP profiles corresponding to 30,616 molecules (4 replicates per molecule).

One-hot encoding is a binary vector representation of categorical data. Here, each category is assigned a unique index and a binary vector of length equal to the number of categories. All elements are zero except for the index corresponding to the specific category, which is set to 1. One-hot encoded SELFIES were used for molecular representation, where each SELFIES alphabet would correspond to 1 when present in a string.

The VAE had two goals. First, to encode the molecular representations into a continuous latent space, and second, to have a decoder that could convert any point on the latent space into a valid molecular representation. This VAE was a similar implementation to the one proposed by Winter *et al.* (2019).[54]

Their encoder consisted of three stacked Gated Recurrent Unit (GRU) cells and two fully connected layers, one of which predicted the mean and the other the standard deviation of a distribution. The cell states of the GRU cells were concatenated and fed into the two hidden layers. The latent vectors sampled from these hidden layers had values between -1 and 1 and had 256 dimensions.

The decoder consisted of a dense layer, three stacked GRU cells, a dropout layer and an output layer. The input was a 256-dimensional latent vector, connected to a hidden layer of dimension 768 (three times the latent dimension), again split into three vectors of 256 dimensions, each of which would be used as the initial state of three stacked GRU cells. The output of these GRU cells follows a dropout of 0.2 and then a connection with an output dense layer with the *softmax* activation function to generate the probability distribution of the one-hot encoded SELFIES.

The goal of the cGAN was to generate molecules that could induce specific CP profiles. The cGAN consisted of three neural networks. First, the generator, whose task was to suggest points in the latent space of the previously trained VAE that would correspond to the molecules inducing a query CP profile. Second, the discriminator, which would train to compete with the generator. Finally, the conditional network, would compute the probability of the molecule suggested by the generator to induce the query CP profile.

The generator would receive two inputs - the CP profiles of 1449 features and a 1000 dimensional noise vector from a standard normal distribution. These CP profiles did not undergo any feature selection. Three hidden layers of 1034, 512, and 256 nodes, respectively, would follow the CP profiles input node, while two hidden layers of 512 and 256 nodes would follow the noise vector input node. The last hidden layers would be concatenated, followed by another hidden layer of 256 nodes, followed by an output layer

of 256 nodes. For all layers except the output layer, where 'tanh' was used, LeakyRelu was used as the activation function.

The discriminator consisted of a 4-layer NN, where all hidden layers consisted of 256 nodes. The output layer had a single node. All hidden layers had LeakyRelu activation function. Two dropout layers with a rate of 0.4 were present between the second and third layers, and between the third and fourth layers.

The conditional network was an independent NN model as it was not trained in tandem with the generator and the discriminator. It received two inputs - the CP profile and the latent space coordinates representing the encoded molecular string. The CP profile input layer was connected to three hidden layers of 1024, 512, 256 nodes each, followed by a dropout layer with a rate of 0.4. The latent space coordinates input layer was connected to two hidden layers of 256 nodes each, followed by a dropout layer. The networks of both input types were concatenated, followed by a connection to a hidden layer of 256 nodes with LeakyRelu as the activation function. The single node output layer had a sigmoid activation function. This conditional network would estimate the probability of a molecule producing a particular CP profile.

The conditional network was trained separately and its weights were frozen after training. The cGAN was trained for 500 epochs with a batch size of 256. An epoch had 496 steps. The weights of the discriminator were updated after every step and those of the generator were updated after every 10 steps. RMSprop with a learning rate of 0.0001 was used to train the generator and the discriminator.

Fréchet distance, which measures the similarities between the two distributions, here the real and the generated distributions, was used to evaluate the models. It is given by:

$$d^2((\mu_r, C_r), (\mu_g, C_g)) = ||\mu_g - \mu_r||^2 + \mathrm{Tr}(C_g + C_r - 2(C_g C_r)^{1/2})$$

The suffixes r and g stand for real and generated, respectively. μ_r and μ_g are the means of two distributions, C_r and C_g are the covariance matrices of the two distributions, $||.||$ denotes the Euclidean distance, Tr denotes the trace of a matrix (the sum of the elements on the main diagonal). The distance $d^2((\mu_r, C_r), (\mu_g, C_g))$ is a measure of the difference between the two distributions.

3.3. Results

Since molecules generated by generative models can also be chemically unrealistic, it is important to filter out such unwanted outputs. *FilterCatalog* by RDKit is a collection of filters, where each filter flags molecules for unwanted properties. Such filters are prominent in the cheminformatics community. For example, Pan Assay Interference (PAINS)

filters out molecules, with specific substructures, which show up as false positives in numerous biological assays. Filters suggested by ZINC, aim to removes molecules with unwanted functional groups. Filters suggested by Brenk *et al.* (2008) flags molecules for unfavorable pharmacokinetics.[55]

In this project, *FilterCatalog* was used for filtering out invalid and unrealistic generated molecules, while retaining the reasonable generated molecules for further analysis. These filters were chosen - PAINS (A, B, and C) filters, filters suggested by Brenk *et al.*, filters suggested by National Institute for Health, and filters suggested by ZINC.

3.3.1. MolVAE Developed

A VAE model was developed and validated. This VAE model, termed MolVAE, takes an input molecule and returns numerous similar molecules. The model input and output are in form of one-hot encoded SELFIES strings.

While for this project, MolVAE was developed to work in-tandem with a cGAN to generate molecules for queried CP profiles, MolVAE in itself is a standalone generative model. In this section, its evaluation and two possible applications are described.

An analysis based on its generative capacity was performed. MolVAE was setup in inference mode. As the generative nature of the VAEs is due to the stochastic nature of the encoder (in turn, due to the reparameterization trick (Eq. 3.4)), the model setup was straight-forward. The encoder was received the input and its output served as the input for the decoder. A set of 1,000 randomly selected molecules, which were not present in the training set, were provided as queries. These 1,000 molecules came from a subset of compounds added after CHEMBL 29 (which was used to train the MolVAE), and are present in the current CHEMBL 34.

For each query molecule, following steps were performed.

- 50 molecules per query were generated.

- Using *FilterCatalog*, unwanted/unrealistic molecules were filtered out.

- Only first instance of each generated molecule was stored. Rest were removed.

- Since query molecule can itself be generated as an output, such duplicates were counted.

- Structure similarity was calculated between remaining valid molecules and the query molecule. Tanimoto similarity of MF of radius 4 and 2048 bits was used for such similarity calculations. The median and the standard deviation of similarities of all this structure similarity were calculated.

Once the above steps were performed for all the query molecules, box-plots were created (Figure 3.4).

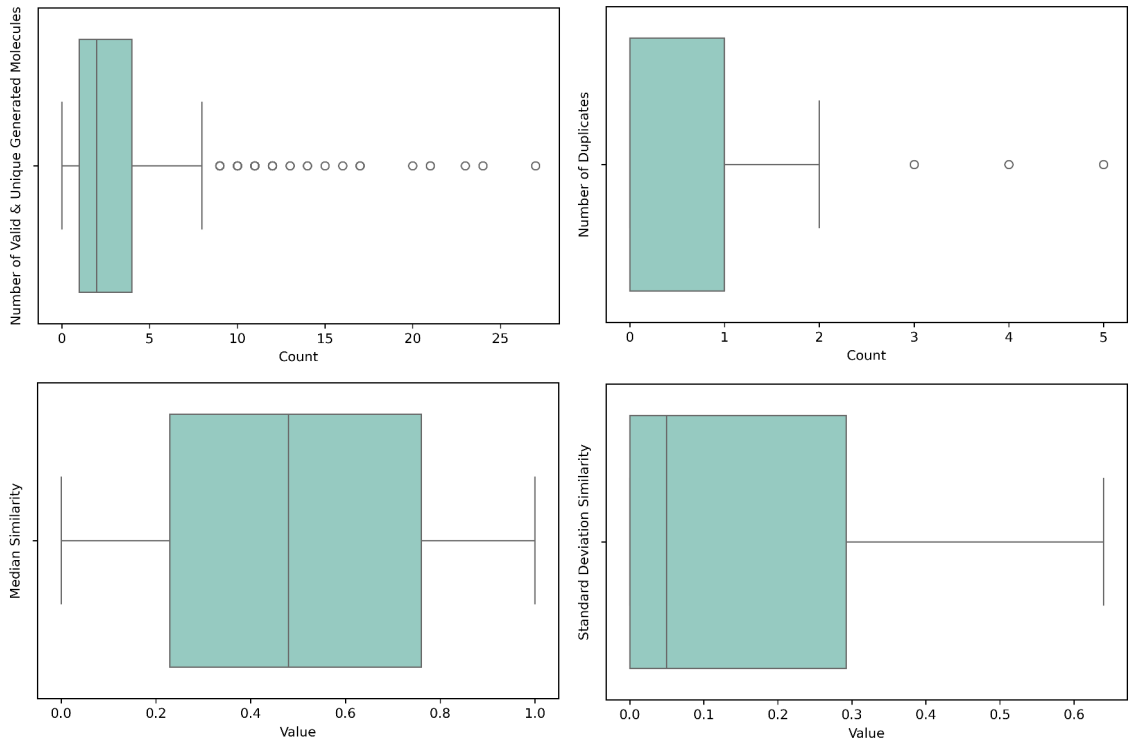


Figure 3.4.: Box-plots of MolVAE's stand-alone generative performance analysis

Many points can be inferred from these box-plots. First, in general, less than 10% novel, valid and unique compounds are generated per iteration, even with a high query number (here, 50). Second, the original query molecule is usually regenerated. Since this data was not present in the training set, this shows that the model can successfully embed a (novel) molecule into low-dimension (its latent layer) and then generate it back. Third, generated molecules show a wide-range of structural similarities with query molecules.

Molecules generated by MolVAE in this analysis were generally valid, however, they were generated multiple times. It is to be noted that due to the stochastic nature of the models, these numbers can vary in next iteration of the same analysis.

Standalone MolVAE Application: Variation Generation

Given the generative nature of the VAE, it is possible to generate similar outputs by repeatedly sampling from the same latent space and passing these latent vectors through the decoder. This property can be exploited to generate molecules similar to the input.

A standalone variation generation script was developed using MolVAE. This script would ask the user for an input molecule and the number of variations to generate, and would ideally return the requested number of similar molecules generated.

Figure 3.5 shows an example query molecule with 8 variations. Each generated molecule's

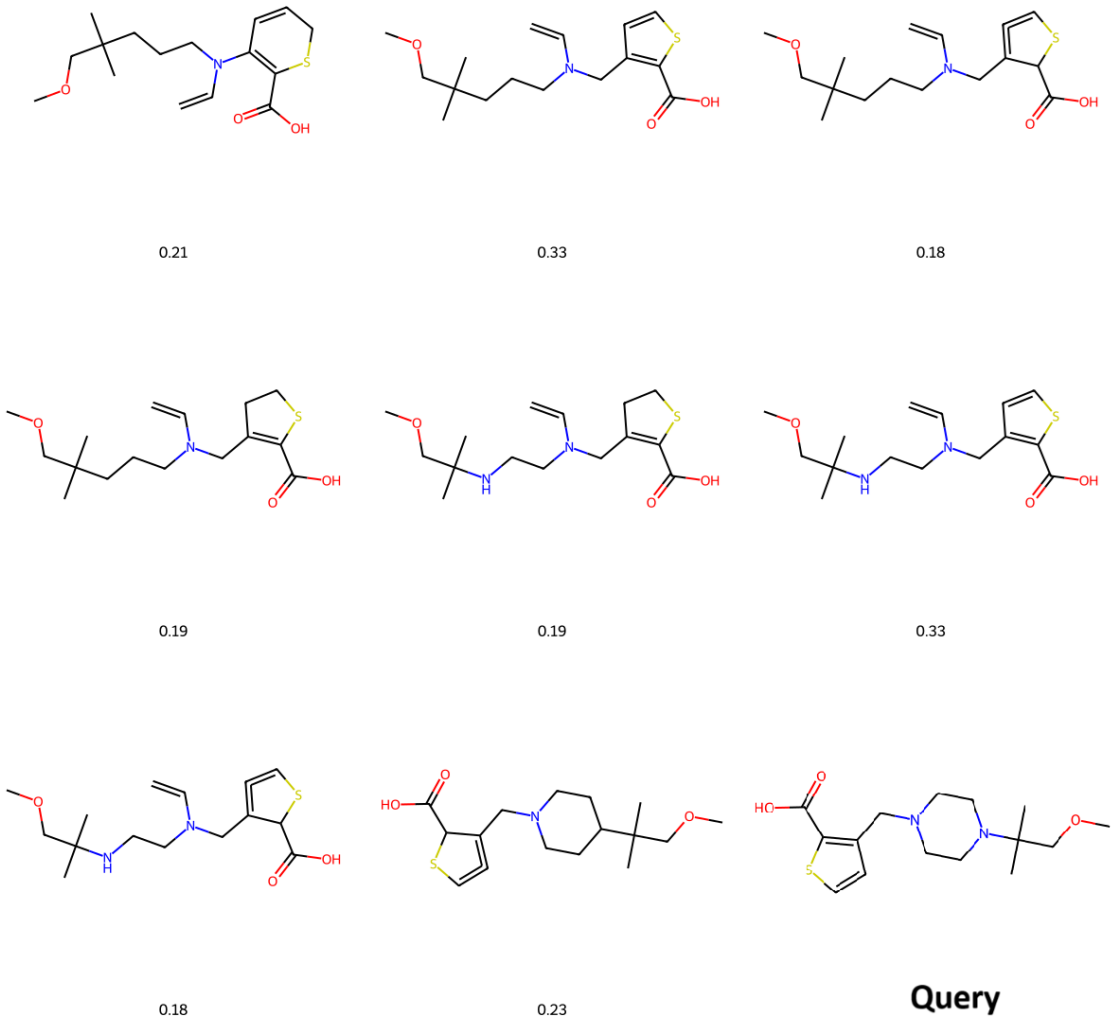


Figure 3.5.: Generated variations of an example query molecule. Values below the molecules are their similarity scores based on MF similarity to the Query.

similarity to the query molecule is also shown.

Standalone MolVAE Application: Interpolation Generation

The latent space has a continuous nature, i.e., any point on the latent space, once decoded, should result in a valid molecule. This makes it possible to sample and interpolate between two points on this high-dimensional space. Interpolation on such a multivariate Gaussian distribution is performed using a mathematical function called the spherical linear interpolation function or *Slerp*.

A standalone interpolation generation script was developed using MolVAE. This script would ask the user for two input molecules and the number of interpolations to generate

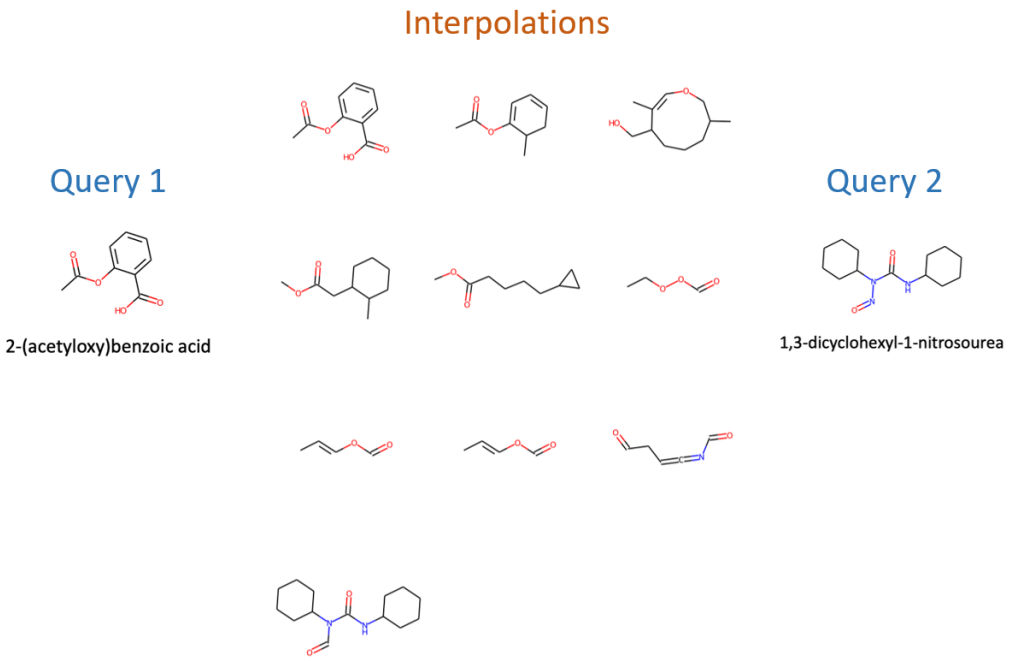


Figure 3.6.: Interpolation generated between latent space mean vectors of these two query molecules.

on the latent space between the means vectors of these two inputs to generate.

Figure 3.6 shows an example of interpolations generated between two query molecules.

3.3.2. cGAN Developed

A cGAN which take CP profile as an input and can generate molecules was developed. A similar analysis was performed like that of MolVAE to assess its performance. Its generative power was found to be non-existent, making it unusable for the generative modeling.

The model setup and analysis is described as follows. The cGAN comprises of three parts - a generator, a discriminator and a conditional network. Once the cGAN is trained, only generator is required to run the model in the inference mode. This generator receives a combination of CP profiles and Gaussian noise as input, and it outputs low-dimension information about the molecule. This low dimension information (latent vector of the MolVAE) is read by the decoder of the MolVAE, which converts this low dimension information into high dimension one-hot encodings.

A portion of the MPI dataset not used for tandem training of the generator and the discriminator (random split test set) was used for performance analysis. This data comprised of 1,385 compounds. The CP profiles of these 1,385 compounds were the model input (queries). For each query, following steps were performed.

- The query was repeated 50 times and along with the noise, provided to the generator.

- Generator model predicts the latent vector, which is then converted to high dimension SELFIES one-hot encoding by the MolVAE's decoder.

- These SELFIES are converted to SMILES, which in turn are converted to molecules.

- These molecules are checked for validity with the *FilterCatalog*.

- MF is generated of the compound whose CP profile is provided as input. All generated molecules' MF are calculated, and are then compared for structural similarity against the 'real' molecule, using Tanimoto similarity.

- QED score, indicating drug-likeness, is calculated for all generated molecules.

- The total number of valid molecules generated, the median and standard deviation of the similarities and the QED is returned.

No valid molecules were generated in this analysis. All generated molecules were unrealistic which showed unnatural chemical properties, like, long aliphatic chains with double or triple bonds or other elements in unusual places; unusual macrocycles or other cyclic entities; valence violations, etc. Due to this, no further analyses were carried out.

3.4. Methods

3.4.1. MolVAE

Model Input and Processing

CHEMBL 29, containing approximately 2.08 million molecules, was selected as the compound dataset. SELFIES (*v 1.0.2*) of all molecules were generated using their canonical SMILES. Molecules with a SELFIES length greater than 120 were filtered out. The frequency of the SELFIES alphabets was calculated and only the top 50 alphabets were selected. These SELFIES alphabets, along with two extra flag/padding alphabets were saved separately in a file. Molecules with SELFIES alphabets other than the selected alphabets were filtered out. It was decided to keep only organic molecules in the dataset. Molecules that did not have one of the SELFIES alphabets - [C], [Ring1] or [=C] - were filtered out. The processed dataset contained 2,016,604 molecules.

The dataset was first split into 90%-10% random train-test split. The test set was further split into 90%-10% test-validation split. The validation set was used to monitor the model training and to perform early stopping to prevent overfitting. As a result, there were 1,814,943 molecules in the training set, 181,494 in the test set, and 20,167 in the validation set.

Encoder

An input layer of shape (x, 120, 52) was initiated. Here, x is the sample size, 120 is the maximum SELFIES length, and 52 is the number of SELFIES alphabets (top 50 plus two flag alphabets). A GRU layer of dimensions 256 was initiated and connected to the input layer. The "return_state" parameter was set to true to return the internal state. The states were connected to two independent dense layers. During the model training, one of these layers learns to predict the mean, while the other learns to predict the logarithm of the variance of the latent distribution.

Sampling Layer

A custom layer, labelled 'Sampling,' was declared to perform the reparameterization trick as described in Section 3.2.1. Reparameterization allows the VAE to backpropagate gradients. As input, this layer takes the mean and the logarithm of the variance of the latent distribution in the form of two dense layers. It then takes a random sample from a standard normal distribution called 'epsilon' (ϵ). This ϵ is then scaled by the standard deviation of the learned distribution, followed by a shift to the mean of the learned distribution, and returned as the output of this layer. It is shown in the equation 3.3.

Decoder

An input layer of 256 dimensions was initiated. This was connected to a dense layer. This dense layer is the latent layer. Its activation function was set to 'tanh', and its dimensions were set to 256.

A repeat function was added to convert the tensor from 2D to 3D. This repeat function was added to a bidirectional GRU layer. Here too, the 'return_state' parameter was set to true to return the internal state. The output of this layer was connected to a dense layer with the same dimensions as original input. This output layer had 'softmax' as the activation function.

Model Initialization and Training

ADAM was chosen as the compiler with its default learning rate. The sum of the categorical cross entropy and the KL divergence (ELBO) was used as the model training loss function. A new hyperparameter β was introduced to constrain the KL divergence. The value of β used was 0.01.

The model was trained for 150 epochs with a batch size of 512 samples. Model checkpoints were created for the best epoch based on the validation score during model training.

The model architecture of MolVAE is shown in Figure 3.7.

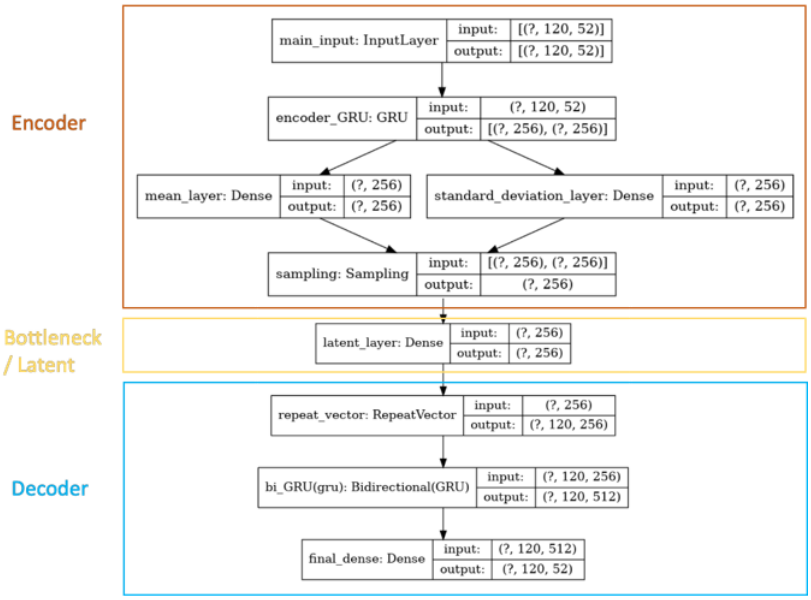


Figure 3.7.: Model architecture of MolVAE

Model Fine-tuning

After first round of training, model weights were loaded for the checkpoint with the lowest validation loss. A sample of molecules from the test set was converted to 2D OHE SELFIES, passed through the encoder model, and then through the decoder model. The output binary array was converted to SELFIES, then to SMILES, then to RDKit molecular objects. This allowed for the generation of molecules that would ideally either vary or be similar to the corresponding input molecules.

The generated molecules were manually inspected. While all of the molecules had a valid molecular string, resulting in the generation of a molecule, many of them were in fact unreasonable molecules. Such unreasonable molecules would exhibit abnormal chemistry.

The model was then retrained for another 100 epochs, but at a much lower learning rate of 0.0001. The above process of loading model weights based on the best checkpoint followed by manual evaluation was performed. The VAE was saved as a whole, while the encoder and the decoder were also saved separately.

3.4.2. Conditional GAN

cGAN comprises of 3 networks - the generator, the discriminator and the conditional network. The model setup and training is complex. The generator and the discriminator train in tandem in a min-max setting. The conditional network is pre-trained and incorporated into the training of the generator and the discriminator.

Cell Painting Assay Dataset and its Processing

The CPA dataset from December 10, 2020, containing 27,608 molecules, was used for this project. It underwent preprocessing - selection of molecules tested at concentrations of either 10 μM or 2 μM, exclusion of toxic molecules, and removal of entries with missing structural information.

The next processing step was to generate SELFIES of all molecules. Molecules with a SELFIES length greater than 120 were filtered out. The SELFIES alphabets saved during the creation of the MolVAE were imported. Molecules with additional SELFIES alphabets were filtered out.

The saved encoder model from MolVAE was loaded. From this saved model, a newer model was initiated where the 'Sampling' and log_var layers were excluded. In other words, this derived model takes 2D OHE SELFIES as input through the encoder's input layer, passes them to the encoder's GRU layer, and finally through the encoder's mean layer. As a result, the mean latent layer coordinates of the molecules could be obtained. This model was then set to not trainable. Using this mean encoder model, the mean latent coordinates of all the molecules in the dataset were calculated and added to an additional column of this dataset.

In the final processing steps, the dataset was ordered by the induction value in the descending order, the duplicate molecules were removed and the dataset was split into 90%-10% training and test sets. The training set consisted of 12,458 molecules. The test set consisted of 1385 molecules and was used as the test set for the entire cGAN in later steps.

Conditional Network Pretraining

Note that the conditional network was pretrained. It simply means that it is trained separately in advance. The term pretrained is used because the conditional network's weight are frozen, and it does not take part in training when the generator and the discriminator train.

The conditional network here is a binary classifier that outputs a probability. If the mean latent coordinates of a molecule, which are (approximately) its valid compressed string representation, match the CP profiles that this molecule induces, it outputs a value around 1, and if there is a mismatch, it outputs a value around 0. This required some custom dataset preparation.

The data for standalone training and evaluation of the conditional network was prepared as follows.

The metadata (such as SMILES, Well ID, etc.) was separated from the CP features (each of the 579 morphological changes). From this metadata - SMILES, Compound ID and

the mean latent coordinates were isolated and stored in a separate DataFrame. A copy of this DataFrame was made in reverse order. Both of the copies were merged with the unmodified CP features data. As a result, the DataFrame with the original SMILES, Compound ID, and the mean latent coordinates was merged with the correct CP features, whereas the reversed DataFrame was merged with the incorrect CP features.

A column named 'Y' was added to the DataFrame with the correct values, and all entries under this column were 1. This marked the entry as correct. Conversely, in the DataFrame with the incorrect values, all entries under 'Y' were 0, indicating that the entry was incorrect.

These two DataFrames were concatenated. The result was a new dataset with 24,916 entries. A train-test-validation split was performed on this new dataset. The dataset was first split into a 90%-10% random train-test split. The training set was further split into a 90%-10% train-validation split. The validation set was used to monitor model training and perform early stopping to prevent overfitting. As a result, there were 20,181 entries in the training set, 2492 in the test set, and 2243 in the validation set.

The model architecture of the conditional network is shown in Figure 3.8. ADAM was used as the optimizer with an initial learning rate of 0.0001, and binary cross-entropy was used as the loss function. The batch size was set to 128. The model was trained for 108 epochs (out of 500) until an early stop was triggered.

To validate the model performance on the test set, the model predictions, which were probabilities, were converted to absolute values. This was done by converting probabilities above or equal to 0.5 to 1, while those below 0.5 were converted to 0. When compared to the true flags, the balanced accuracy of the model was 0.7 and Cohen's kappa was 0.4.

cGAN Model Architecture

The discriminator had an input layer with the size of the latent space - 256. It had 4 hidden layers with 200 nodes each. Each hidden layer had LeakyReLU as activation function with parameter 'alpha' set to 0.2. Between hidden layers 1 and 2, and 3 and 4, there were dropout layers with a rate of 0.4. The output layer had a single node and a sigmoid activation function. It was compiled using the ADAM optimizer with a learning rate of 0.0001. Binary cross entropy was selected as the loss function.

The generator had two input layers. The input layer, which took the noise as input, had a size of 1000. This input layer was followed by two hidden layers of 500 and 200 nodes each. The input layer taking CP profiles had a size of 579. This input layer was followed by two hidden layers of 200 and 100 nodes, respectively. The last hidden layers of both inputs were concatenated. This was followed by another hidden layer of 200 nodes, and this was followed by the output layer of 256 nodes. Each hidden layer had LeakyReLU

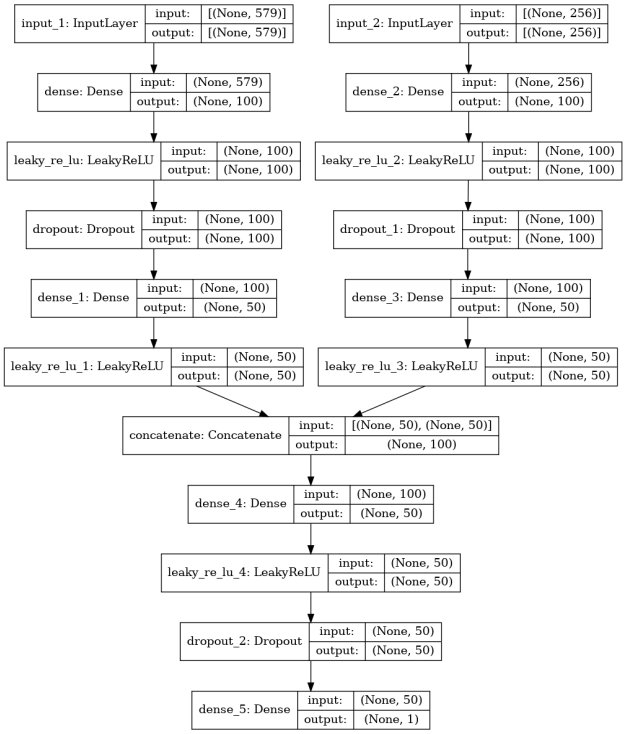


Figure 3.8.: Neural Network (NN) architecture plot of the conditional network. Note that the activation function LeakyReLU is added as a separate layer. All the dropout layers are set to a rate of 0.4.

as the activation function with the 'alpha' parameter set to 0.2. Batch optimization was performed after each hidden layer. Note that the generator is not compiled.

To put all the models together, the cGAN architecture was defined and compiled. For input, the two input layers of the generator were defined as its input layers. For output, the outputs of the discriminator and the conditional net were defined as its outputs. The discriminator and the conditional network were set to non-trainable. It was compiled using the ADAM optimizer with a learning rate of 0.0001. Binary cross entropy was selected as the loss function.

cGAN Training

The model training process described below was inspired by a blog by Jason Brownlee.[56]

Three functions were defined to support cGAN training. The first function would prepare the inputs for the generator for custom batch sizes. The function to generate "real samples" would take the mean latent coordinates of the compounds and their corresponding CP profiles, and a custom batch size as arguments. Given this custom batch size, it would randomly select those many instances of latent coordinates and their corresponding CP profiles. Finally, it would return the selected latent coordinates with their corresponding CP profiles encapsulated in a list, and a flag of 1 indicating that they are

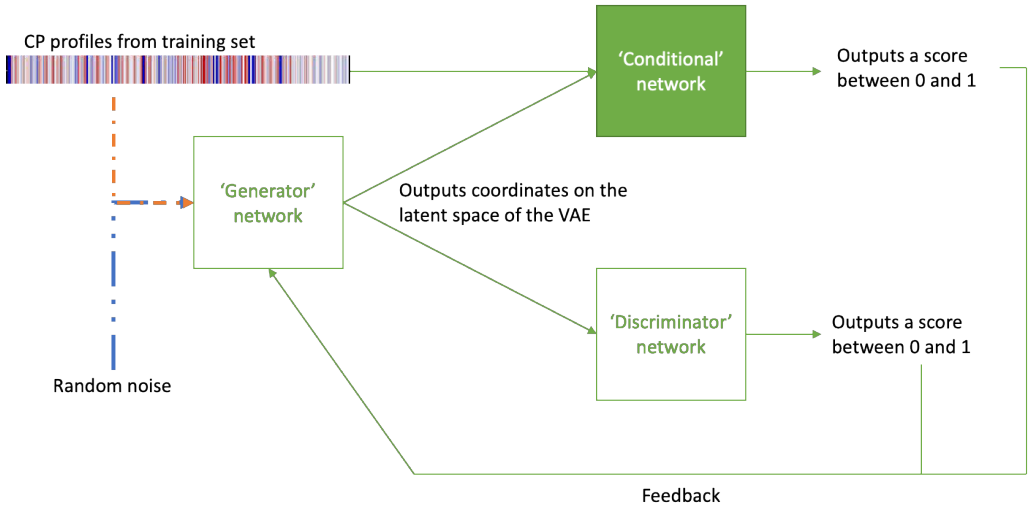


Figure 3.9.: cGAN training diagram.

real samples. The function to generate "fake samples" would return the outputs from the generator of the custom batch size with a flag of 0 indicating that the samples are coming from the generator.

Due to the complex training of the generator and the discriminator, a custom training function was defined. This function implemented the standard training for a cGAN, where the discriminator and the generator are trained alternately. This function took the generator, the discriminator, the combined GAN model, the training set, the noise dimension which was set to 1000, the number of epochs - set to 100, and the batch size - set to 64 as arguments.

It first calculates the batches per epoch. It then enters a loop for the number of epochs specified. Within each epoch, it loops over each batch in the training set. Within each batch, it generates real samples from the training dataset and trains the discriminator on these real samples. It then generates the fake samples using the generator and trains the discriminator on these fake samples. It then trains the generator using the combined cGAN model, where the generator tries to fool the discriminator into thinking that the samples generated by the generator are real. After each epoch, the loss for the discriminator on the real samples, the loss for the discriminator on the fake samples, and the loss for the generator are printed.

Figure 3.9 shows the cGAN training schema.

3.5. Outlook and Conclusion

Two short-comings of this project are clearly noticeable.

First, the underlying assumption of this generative modeling approach was that the la-

tent space of the VAE is a continuous high-dimensional space of valid encoded string molecular representations. SELFIES solve the problem of encoding molecules in a continuous latent space. However, at least for the initial versions of SELFIES, molecules generated by such a method result in a high number of unreasonable structures. Similar behavior has been reported for other de novo molecule generation methods.[46] The fact that such unreasonable molecules were not present in the training data shows that the trained latent space, although continuous, is imperfect.

Second, the ability of MolVAE to generate valid molecules most of the times independently highlights that the cGAN (the generator, in particular) is unable to determine the 'good spots' on the latent space which correspond to valid molecules. The limited training data size of cGAN can be a speculated to be one of the reasons of poor performance as Méndez-Lucio *et al.* trained their cGAN with almost ten-times more data.

This project closely followed the methods described in the preprint by Méndez-Lucio *et al.* (2020).[47] However, an updated version of the manuscript with open source code was published in 2023.[57] The methods and codes mentioned in the updated version were not tested here, since this project was stopped at the end of 2021.

4. Lysosomotropism and Explainable Machine Learning

The following chapter is accepted as a publication in the journal - *RSC Medicinal Chemistry*, under the title, "Identification of Lysosomotropism using Explainable Machine Learning and Morphological Profiling Cell Painting Data" by Aishvarya Tandon, Anna Santura, Dr. Axel Pahl, Prof. Dr. Dr. h.c. Herbert Waldmann, and Prof. Dr. Paul Czodrowski. The manuscript underwent peer-review and is presented here unchanged.

4.1. Abstract

Lysosomotropism is a phenomenon of diverse pharmaceutical interests because it is a property of compounds with diverse chemical structures and primary targets. While it is primarily reported to be caused by compounds having suitable lipophilicity and basicity values, not all compounds that fulfill such criteria are in fact lysosomotropic. Here, we use morphological profiling by means of the Cell Painting Assay (CPA) as a reliable surrogate to identify lysosomotropism. We noticed that only 35% of the compound subset with matching physicochemical properties show the lysosomotropic phenotype. Based on a Matched Molecular Pair Analysis (MMPA), no key substructures driving lysosomotropism could be identified. However, using Explainable Machine Learning (XML), we were able to highlight that higher lipophilicity, basicity, molecular weight, and lower topological polar surface area are among the important properties that induce lysosomotropism in the compounds of this subset.

4.2. Introduction

The lysosome plays a crucial role in the cellular degradation of biopolymers and in processes, such as apoptosis, autophagy, and cell signaling. Lipophilic small molecules and drugs that carry a basic moiety can accumulate in the lysosome, by passing the lysosomal membrane in their neutral form, getting trapped in the compartment due to protonation at the lower pH (Fig. 4.1). The pathophysiological consequences of this phenomenon termed lysosomotropism are not yet fully understood, but impairment

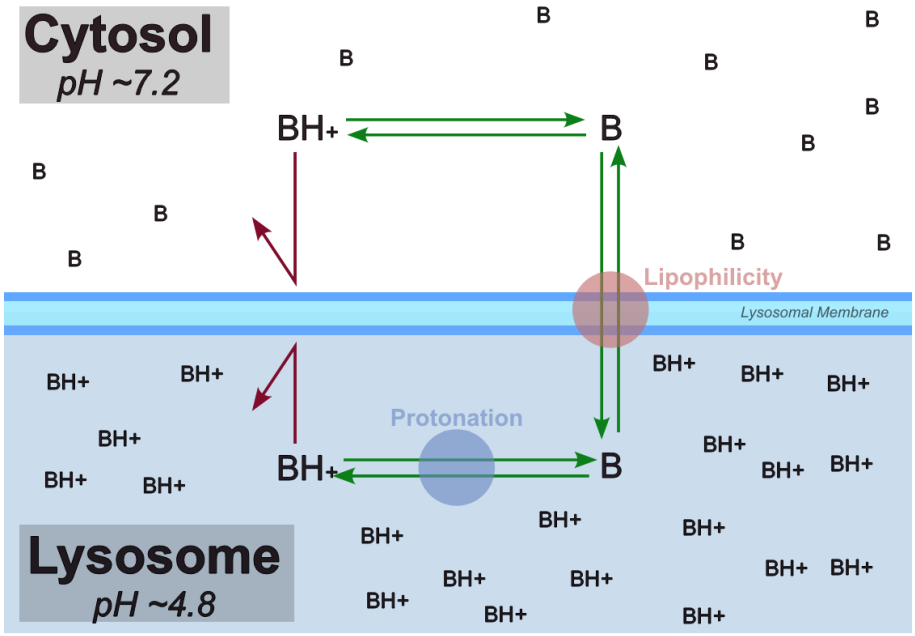


Figure 4.1.: Lysosomotropism is primarily believed to be driven by lipophilicity and protonation state of the compound. In this diagram, "B" is a lysosomotropic compound and "BH^+" is its protonated state. Illustration inspired by Kuzu *et al.* (2017).[60]

of lysosomal functionality can be linked to phospholipidosis and disturbed cholesterol homeostasis.[16,58,59]

Recently lysosomotropism has especially drawn attention in multiple drug repurposing studies targeting severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2). This stems from the crucial involvement of cathepsin L, a lysosomal protease, in cleaving the SARS-CoV-2 spike protein and facilitating virus entry into host cells. By the accumulation of lysosomotropic drugs in the lysosome, the compartment's pH rises, rendering proteolytic enzymes inactive and impeding viral replication.[61–63]

Consequently, well-established lysosomotropic drugs, e. g. chloroquine and hydroxychloroquine, were initially promising drug repurposing candidates against SARS-CoV-2. However, these drugs failed to demonstrate significant clinical benefits.[62,64–69]

In a more general sense, lysosomotropism is also a phenomenon of various drugs. It occurs for structurally different compound classes, modes of actions and targets, and is independent of species and cell-type.[16,70] For instance, lysosomotropic properties have been observed in anticancer compounds (tamoxifen, doxorubicin, daunorubicin, mitoxantrone), tyrosine kinase inhibitors (imatinib, dasatinib, sunitinib, sorefenib), β-blockers (propranolol), antihistamines (promethazine, astemizole, dimebon, desloratadine), and selective serotonin reuptake inhibitors (sertraline, paroxetine, fluoxetine, fluvoxamine).[60,71–73]

Given such broad range of interests, several groups have contributed towards the measurement, quantification and prediction of lysosomotropism, see e.g. Nadanaciva *et al.*

(2011), Ufuk *et al.* (2017), Schmitt *et al.* (2018) and Norinder *et al.* (2019).[71,74–76] Highlighting the importance of identifying lysosomotropism early in the development process, Hu *et al.* (2023) recently developed and published models on phospholipidosis, a process related to lysosomotropism, using compound literature data. They also validated their results using a live-cell imaging assay.[77]

Compounds with a calculated *logP* (clogP) value greater than 2 and a basic *pK*a (bpKa) value between 6.5 and 11, representing lipophilicity and basicity, respectively, are likely to be lysosomotropic.[71] Here, we will refer to this cross-section of properties as the physicochemical window ("PhysChem window"). While lysosomotropism at first seems to be primarily driven by lipophilicity and protonation state of compounds, it has been established that not all molecules which have the suitable physicochemical properties are in fact lysosomotropic.[71,78]

The Cell Painting Assay (CPA) is an unbiased, image-based phenotypic assay where morphological profiles consisting of hundreds of features are generated from the images of compound-treated and control cells.[2] One typical use case of the CPA is the formation of target hypotheses for test compounds with unknown biological activity. Here profiles of test compounds are compared to those of reference compounds with annotated targets or pathways. However, many compounds with lysosomotropic properties induce a distinct phenotype in the CPA which is independent of their target activity.[16] We have observed that comparing morphological profiles of test compounds to that of a known lysosomotropic agent - Smoothened agonist (SAG), is a reliable surrogate for determining lysosomotropism.

Machine Learning (ML) methods have become an integral part of drug discovery. Some prominent methods are QSAR, prediction of chemical reactions and retrosynthesis, and the generation of novel chemical structures.[79,80] Programming packages such as LIME and SHAP offer "explainability" of a model, enabling the interpretation of its predictions.[81,82] The transparency about a model's predictions inspires confidence in researchers to trust them, which is why these packages are gaining popularity and application in drug discovery.[83–85] Similarly, input features found important by a bioactivity prediction model can be determined using such packages, and this information can be used to develop a hypothesis of the underlying mechanism of the bioactivity.

Herein, we investigate the lysosomotropism observed by the CPA using Matched Molecular Pair Analysis (MMPA) and Explainable Machine Learning (XML) to understand which physicochemical descriptor and/or chemical substructures affect lysosomotropism in compounds with feasible basicity and lipophilic values.

With the MMPA, we aim to identify key substructures which are responsible for transformation of a lysosomotropic compound to a non-lysosomotropic compound, and *vice-versa*. Similarly, by interpreting tree-based Machine Learning (ML) models with molecular fingerprints we addressed the identification of important substructures whose pres-

ence affects lysosomotropism. Finally, by interpreting ML models with molecular descriptors as input, the determination of physicochemical parameters which affect lysosomotropism is attempted.

4.3. Results

4.3.1. Determination of Lysosomotropism

The Cell Painting Assay (CPA) is a morphological profiling assay where six dyes are used to selectively stain different cell organelles and compartments, followed by high-content imaging and analysis, generating morphological fingerprints with hundreds of features.[2,3] CPA is an unbiased assay, that can identify biological activity without requiring a prior target hypothesis, and is therefore particularly well-suited for the screening of new chemical entities of unknown activity. Comparison of the CP profile of a hit molecule with the profiles of reference compounds whose modes of actions and targets are known can then provide target hypotheses, potentially enabling target identification.[13–15,19,29,86]

In our implementation of post-imaging analysis following the feature calculation by the open-source software *CellProfiler*, 579 *Z-scores* of morphological features are deduced per compound. The *Z-score* of a morphological feature represents the difference between a morphological feature and its relative DMSO control. A compound's morphological profile (or simply its CP profile) is thereby a list of its *Z-scores*.[13]

Our processed CP data represents a total of 13450 compounds. In this data, 3114 are reference compounds, whose biological activities are annotated, and 10336 are internal research compounds. The internal research compounds primarily consist of natural products-inspired compounds and pseudo-natural products.[18,87] 2065 compounds are present in the PhysChem window, and thereby are relevant to this study.

Induction, a measure of bioactivity, is the percentage of significantly altered features. Compounds with the induction value greater than or equal to 5 are considered bioactive.[13] In the PhysChem window, 1196 compounds are bioactive, while 869 are not.

The CPA is a routine in-house screening assay at the Compound Management and Screening Center, Dortmund, and is used in identifying numerous biological clusters and pathways, among them lysosomotropism. The similarities between CP profiles can be measured by Pearson's similarity. The CP profiles are considered similar if their Pearson's similarity values are greater than 75%. Schneidewind *et al.* (2021) identified a biocluster in the CPA data whose mode of action is likely due to disturbed cholesterol homeostasis caused by lysosomotropism.[16]

Smoothened agonist (SAG), a well-established lysosomotropic compound, is present as a reference compound in the dataset and can be found in the reported biocluster. Because

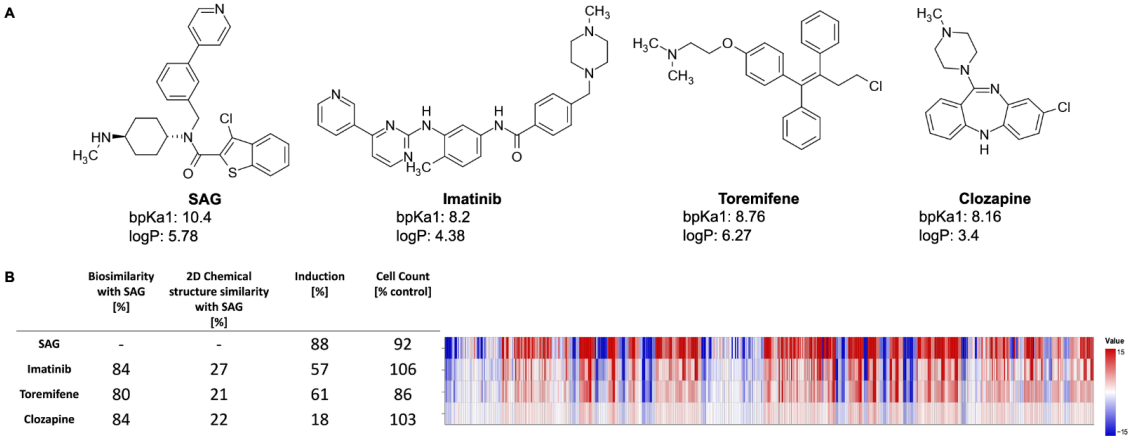


Figure 4.2.:
A) Different chemical structures of SAG, Imatinib, Toremifene, and Clozapine with their calculated bpKa and logP values.
B) The CP profiles (579 features) of SAG, Imatinib, Toremifene, and Clozapine shown as heatmaps. The values of each feature are normalized to the DMSO control. The blue color means that the value is decreased whereas the red color means that the value has increased.
Despite different chemical structures and known pharmaceutical properties, the investigated compounds have similar morphological profiles.

of its pronounced profile in the CPA, SAG was used as the reference compound for defining the lysosomotropic phenotype. This similarity score, termed as the *lyso score*, ranges from 0 (indicating no biosimilarity) to 100 (indicating full biosimilarity). Compounds with a *lyso score* above or equal to 75 were annotated as lysosomotropic given the high biosimilarity of their profile to the profile determined for SAG, the rest were labelled as non-lysosomotropic. Out of the 2065 compounds present in the PhysChem window, 1327 were labelled as non-lysosomotropic and the remaining 738 as lysosomotropic.

Fig. 4.2 exemplarily shows three lysosomotropic reference compounds – Imatinib, Toremifene, and Clozapine – and their CP profiles in comparison to SAG. The morphological profiles of these three compounds and that of SAG are very similar, although these compounds have different primary targets and chemical structures. Imatinib, a tyrosine kinase inhibitor, is primarily used to treat chronic myeloid leukemia,[88] whereas Toremifene, a selective non-steroidal estrogen receptor modulator, is administered to treat breast cancer.[89] Clozapine is an anti-psychotic drug used in the treatment of severely ill patients with schizophrenia. While Clozapine's mode of action is unknown, it is proposed to be an antagonist of dopamine and serotonin receptors.[90]

4.3.2. Matched Molecular Pair Analysis (MMPA)

Concept and Nomenclature

A MMP is formed by two compounds that differ from each other by a defined change at one or more specified positions.[91–93] The point where the change takes place is referred

to as the attachment point; the local environment of an attachment point is denoted as the context, or simply the environment. The part of the molecules that is identical is called constant, and the part that changes is named the variable. Consequently, the compounds belonging to a MMP can be converted to one another by the molecular transformation of variable A to variable B, i. e. $A \rightarrow B$. Transformation types include, for instance, additions ($H \rightarrow X$) or functional group replacements (e. g., $Cl \rightarrow OMe$) or linker/scaffold exchange. Every transformation is associated with a relative change in a property value in general (ΔP),[92] here a change in the *lyso score*. Subsequently, the same transformations are grouped together and the statistics of the property changes are calculated (frequency of occurrence, ΔP distribution, average ΔP, etc.), which collectively yields the rules.[94,95] Taken together, examination of analogous compounds can determine the contribution of each substituent or structural element to the overall property of the compound (assumption of additivity).[96] Furthermore, it is anticipated that the effect of a substituent on the respective physicochemical/biological property can be generalized, i. e. that its contribution is transferable across compound series.[96]

Given that the method captures the implicit knowledge contained in the chemical dataset in a systematic and automated manner, the emergence of the rules is fully explainable (as it is easy to trace back to the underlying compound pairs), resembles the intuitive way of a chemist's thinking, and lacks the "black box" character, which is frequently raised as a point of critique concerning machine learning or other *in silico* methods utilized for (Q)SAR analysis.[93,94]

The MMP concept has been widely employed.[91–99] However, to the best of our knowledge, the application of this approach to a CP data set and with a view to lysosomotropism has not been reported to date.

MMPA results

In total, 6220 MMPs were identified within the dataset of 2065 compounds described above. As a result of the significantly higher number of non-lysosomotropic compared to lysosomotropic compounds in the data set, many more MMPs are found in which both compounds are non-lysosomotropic, compared to the other two cases. For 956 MMPs the lysosomotropic property is altered upon the respective transformation, i. e. the *lyso score* threshold of 75 is passed.

Altogether, 4441 unique transformations have been found regardless of the lysosomotropism classification. However, $> 99\%$ of them occur less than 7 times (Fig. S1) (Supplementary Information). Similar numbers, as well as the Zipfian-shaped distribution of counts (number of occurrence), have already been described by Hussain and Rea (2010),[99] among others.

In summary, only 27 transformations occur ten times or more – of which the top 10

transformations with the highest numbers are shown in Fig. 4.3. Unsurprisingly, most of the "high-count" transformations found either resemble "simple" terminal group substitutions, where only a single atom is replaced, or functional group substitutions.

The Δ value is calculated by subtracting the *lyso score* of the "from" compound from that of "to" compound. In other words, if the Δ *lyso score* distribution is shifted to the right, the transformation is accompanied by an increase in lysosomotropism. However, in none of the "high-count" conversions shown in Fig. 4.3, the change in *lyso score* incidental to the transformations is in one direction only. The MMPA performed in this work intimates that there does not appear to be a dominant structural feature in the compound library under investigation that determines lysosomotropism.

4.3.3. Explainable Machine Learning

Molecular Fingerprints

We chose 3 major categories of molecular fingerprints, Morgan fingerprints (the term is used interchangeably with ECFP here), MACCS keys, and Avalon fingerprints. All fingerprints mentioned here are binary in nature, and are generated by RDKit. Morgan fingerprints use a hashing function to map substructures of different radius to an index of a vector, and that vector can be folded to different sizes.[38,100] We provided 3 different radii of 2, 3 and 4, thereby generated 3 different sets of these fingerprints for our data. MACCS keys encode absence or presence of 166 pre-defined structural features. Avalon fingerprints enumerate certain paths and feature classes of a molecular graph.[101]

Molecular Descriptors

A molecular descriptor, as defined by R. Todeschini and V. Consonni, is "*the final result of a logic and mathematical procedure which transforms chemical information encoded within a symbolic representation of a molecule into an useful number.*"[39]

We used RDKit to generate one- and two-dimensional molecular descriptors of the compounds.[102] Around 200 molecular descriptors can be generated by the RDKit. However, since we were aiming for explainability, we manually selected the descriptors which are intuitive, such as *NumHAcceptors*, *TPSA*, *FractionCSP3*, *fr_nitro*, etc. Some examples of unintuitive descriptors, which were removed, are *PEOE_VSA7*, *VSA_EState8*, *SMR_VSA6*, etc.

In total, 107 intuitive molecular descriptors were selected. These select descriptors are listed in the Table S1. logP and bpKa calculated by ChemAxon *cxcalc*, were also used additionally for one of the models.

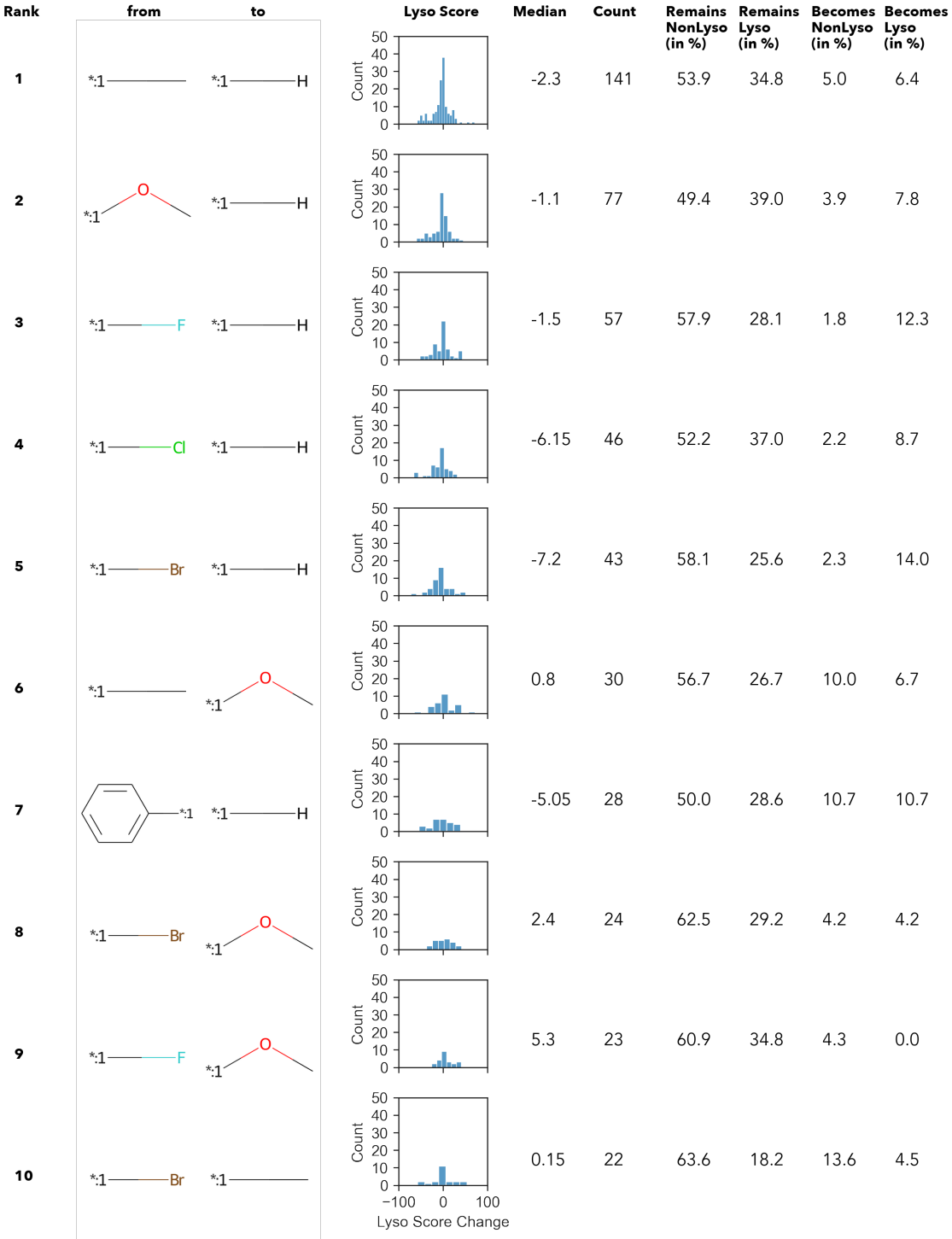


Figure 4.3.: Top 10 transformations with respect to the count. Histograms display the distribution of the change in the *lyso score* value upon the respective transformation; additionally, the median value is given.

Modelling with XGBoost

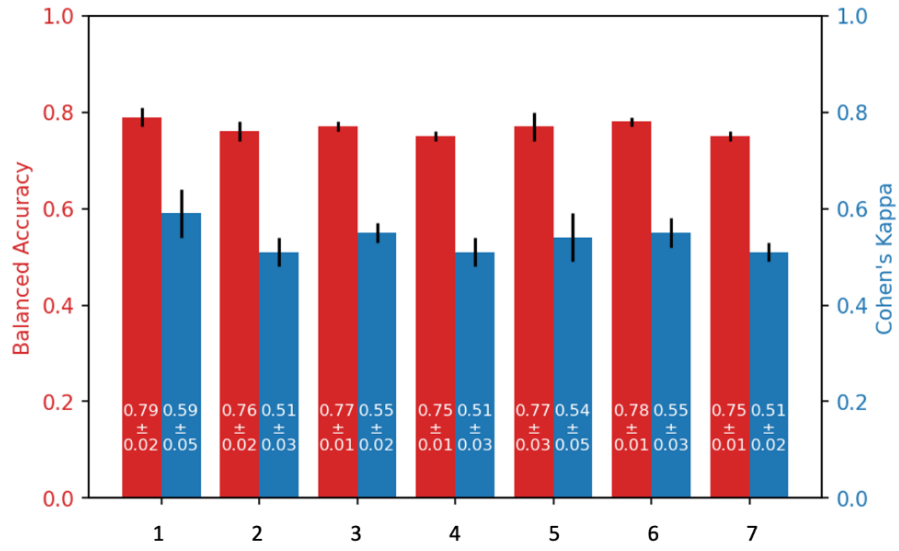
Decision tree models are basic tree-structured ML models. They can be easily understood and interpreted since their logical chain of arriving at a decision can be visualized. However, they suffer from drawbacks, such as overfitting and high computation cost.[103] Gradient boosting is a technique which ensembles numerous weak models (here, decision trees) to make a prediction. This ensemble of trees usually improves both prediction performance and lowers computation cost, however the comprehensibility, which a single decision tree offers, is sacrificed. Extreme Gradient Boosting (XGBoost) is a ML algorithm (and the software library of the same name) which uses optimized gradient boosting, and is engineered to be highly efficient across different platforms and high dimension data.[104]

We prepared XGBoost binary classifiers as described below. Except for the *scale_pos_weight* hyperparameter, which was used to provide the weights of "Non-Lysosomotropic" and "Lysosomotropic" classes to control the class imbalance, default hyperparameters were used for training. The performances of the models trained with the default hyperparameters and with the optimized hyperparameters by the package Optuna[105] were found similar, and thereby the default hyperparameters were used. All the models were trained on the internal data. Stratified 5-fold cross-validation with the Balanced Accuracy and the Cohen's kappa score as model performance metrics, was used to validate the models. The libraries present in the *scikit learn* package were used for these calculations.[40]

Use of SHAP for Feature Importance

Shapley Additive Explanations (SHAP) is a model-agnostic, game-theory based approach to explain ML models. SHAP focuses on local explanations, i. e., the impact of every input feature on the output of a single sample. This impact, a quantitative contribution of the feature, is called Shapley value (term used interchangeably with SHAP value) and is measured in log-odds.[82,106] SHAP is increasingly used for model explanation in cheminformatics.[83,107] We used the *TreeExplainer* from the SHAP Python package, since our models are tree-based. *TreeExplainer* has an important advantage over the default *kernel SHAP*. It computes exact Shapley values by taking advantage of the internal structure of the tree-based models with nominal computation power, whereas the default *kernel SHAP* uses an approximation of Shapley values to save the computation power it would need otherwise to calculate the exact values. Computing exact Shapley values allows global interpretation of the model by combining the local explanations.[108]

Due to the numerical nature of the descriptors, *TreeExplainer* could be used directly on the models trained on the molecular descriptors and various SHAP plots can be em-



Legend:

	Models
1.	Select_RDKit_desc_with_logP_bpKa1_unscaled_model
2.	Morgan_FP_radius4_model
3.	Avalon_fp_model
4.	MACCS_model
5.	Morgan_FP_radius2_model
6.	Select_RDKit_desc_unscaled_model
7.	Morgan_FP_radius3_model

Figure 4.4.: The 5-fold cross validation results of the all models. The black line on top of the bars indicates the standard deviation.

ployed to study the descriptors and their importance on a data set. However, molecular fingerprints are binary in nature and especially in the case of Morgan fingerprints, multiple substructures can be encoded in the same bit. Thus, highlighting bits as important is unintuitive unless the substructures they encode are known. We used *X-FP*, a Python library, to compute the substructures of the bits which Morgan fingerprints encodes. We then used *X-FP*'s functionality to calculate feature importance by SHAP *TreeExplainer* and visualized these important bits and the substructures they encoded.[109]

Model Training Results

The 5-fold cross validation results of all the models are shown in Fig. 4.4. Post cross-validation, models were trained on the entire data, and validated on two different additional datasets.

The molecular descriptor model, "Select_RDKit_desc_with_logP_bpKa1_unscaled_model", and the molecular fingerprint model, "Morgan_FP_radius2_model" were selected as the representatives of

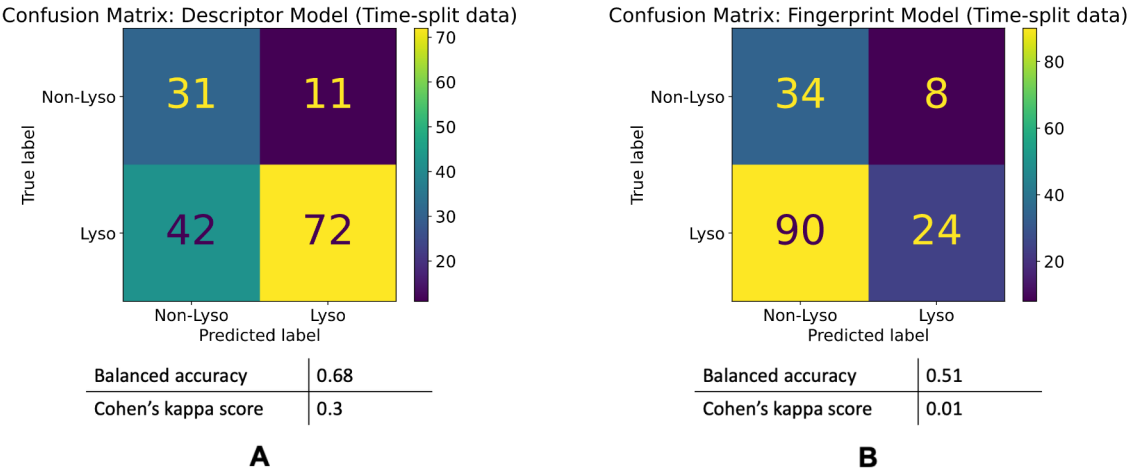


Figure 4.5.: Model performances on the time-split dataset.
Confusion matrices for the Descriptor (A) and Fingerprint (B) models' performances on the time-split dataset.

their respective model types. For convenience, these models are referred to as the "Descriptor model" and the "Fingerprint model", respectively.

The Descriptor model has the average balanced accuracy of 0.79 with the standard deviation of 0.02, and the average Cohen's kappa score of 0.59 with the standard deviation of 0.05. Similarly, the Fingerprint model has the average balanced accuracy of 0.77 with the standard deviation of 0.03, and the average Cohen's kappa score of 0.54 with the standard deviation of 0.05.

Time-split Validation

After the model was developed and validated, new CPA measurements were performed. This allowed us to perform a time-split validation. This dataset consists a total of 156 compounds relevant to this study (located within the PhysChem window), where 114 are labelled as lysosomotropic and the remaining 42 as non-lysosomotropic.

The Descriptor model's balanced accuracy is 0.68 while its Cohen's kappa score is 0.3. The Fingerprint model's balanced accuracy is 0.51 and its Cohen's kappa is 0.01. The confusion matrices of these models' performances are shown in the Figure 4.5.

External Validation

In addition, we performed an external validation of the models with purchased compounds. The selection process of choosing the compounds from the vendor is described in the Materials and Methods section. Out of the 127 compounds in this dataset, in the

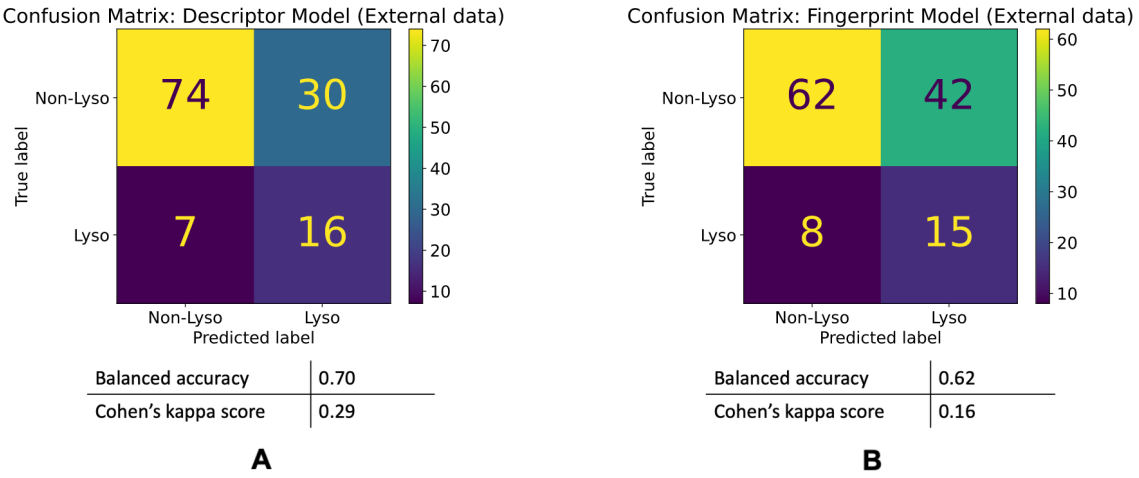


Figure 4.6.: Model performances on the external dataset.
Confusion matrices for the Descriptor (A) and Fingerprint (B) models' performances on the external dataset.

CPA - 104 are non-lysosomotropic while the remaining 23 are lysosomotropic. The Descriptor model's balanced accuracy is 0.70 while its Cohen's kappa score is 0.29. The Fingerprint model's balanced accuracy is 0.62 and its Cohen's kappa is 0.16. The confusion matrices of these models' performances are shown in the Figure 4.6.

SHAP Analysis Results

The SHAP Summary Plots are useful in visualizing the importance of individual input features. In the Summary Plots, Shapley values of each input feature across all the samples in the dataset are plotted together. Since these plots are bee swarm plots by default, the dots having the same Shapley value pile over each other. The input features are ranked higher based on their impact on the overall data. This ordering is based on the mean of the absolute Shapley values for each feature.

The color gradient from blue to red indicates the value of a feature from lower to higher. In our case, positive Shapley value correspond to the lysosomotropic class, and negative Shapley values to the non-lysosomotropic class.

Fig. 4.7 and Fig. 4.8 are the SHAP Summary Plots for the Descriptor model on the training dataset and the validation datasets, respectively. These plots show the descriptors which are found important in each of the SHAP analysis.

SHAP Dependence Plots are scatter plots between a feature and their Shapley values. The Dependence Plots of top 10 features of the Descriptor Model for all the three datasets are present in the Supplementary Information. (Figure S2, S3, and S4)

In the SHAP analysis of the training dataset, logP and bpKa1 are found as the most

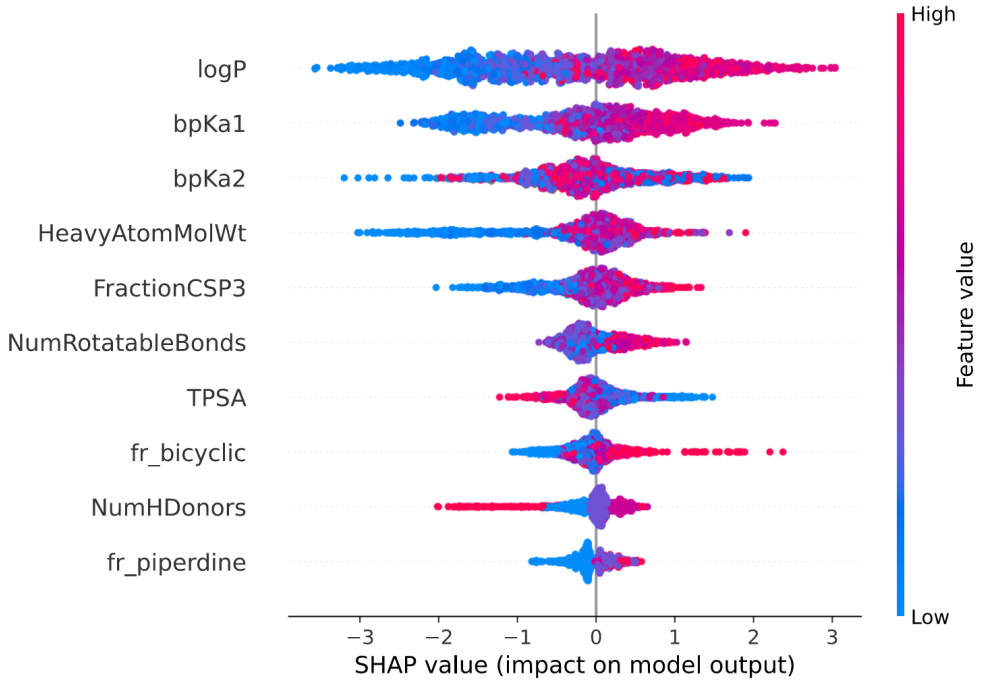


Figure 4.7.: SHAP Summary Plot of the Descriptor model's training dataset.

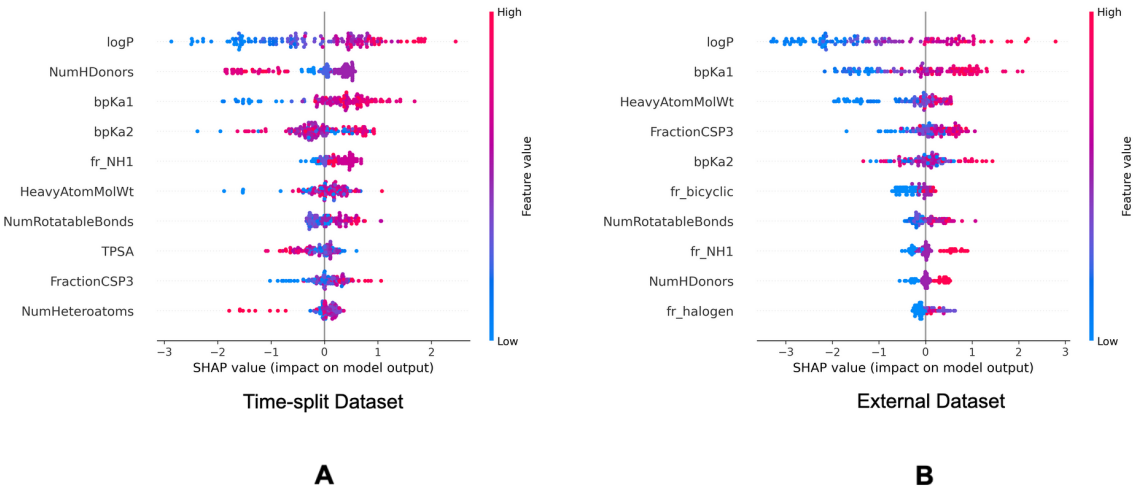


Figure 4.8.: SHAP Summary Plots of the Descriptor model on the validation datasets.
Top 10 molecular descriptors found important in the Descriptor model for the time-split dataset (A) and the external dataset (B).

important descriptors. It can be observed in Fig. 4.7 that the higher logP and bpKa1 values contribute the model output towards the lysosomotropic class, and *vice-versa*. Similar observations are noted in both of the SHAP analysis of the validations sets, the only exception being that the bpKa1 is the third most important descriptor in the time-split dataset.

Descriptor *fr_NH1* which describes the number of secondary amines, is found important in the SHAP analysis of the validations sets. Here, it is noted that the higher number of secondary amines have positive Shapley values indicating that they contribute model outputs to the lysosomotropic class.

Higher Topological Polar Surface Area (TPSA) is associated with poor cell membrane permeability. The inverse relationship between the TPSA values and their corresponding Shapley value indicates that the model outputs are driven towards non-lysosomotropic class when the TPSA of the compounds is higher. This can be justified if it is hypothesized that the non-lysosomotropic compounds have poor lysosome and/or cell membrane permeability.

Descriptor *HeavyAtomMolWt* calculates the average molecular weight of compounds while ignoring the hydrogen atoms. Across all three datasets, especially in the training and the external datasets, it can be observed that lower heavy atom molecular weights have negative Shapley values. Interestingly, the magnitude of negative Shapley values is higher than the positive Shapley values, indicating that the model finds lower molecular weights more important in classifying compounds as non-lysosomotropic.

The *X-FP* reports of the SHAP analysis of the Fingerprint model of different datasets are present in the SI. Bit 2049, primarily encoding the *sp3*-hybridized carbon atom, is found important across all three datasets. When this bit is switched on - indicating the presence of the substructure encoded, the Shapley values are positive. This means that the presence of this substructure contributes the model prediction to the lysosomotropic class. Similarly, Bit 3959, mainly encoding a secondary carbon across all the datasets, is found important in the training set and the external set. Positive Shapley value when this bit is switched on shows that presence of this substructure affects model predictions towards the lysosomotropic class.

Another bit encoding *sp3* hybridized carbon is Bit 1028 which encodes a carbon atom in the aliphatic ring. This bit is found important in the training dataset and the external dataset. Here too, presence of such substructure favors model predictions towards the lysosomotropic class. Interestingly, *FractionCSP3* is a descriptor which describes the fractions of *sp3* hybridized carbons present in a compound, and this descriptor is found important in the SHAP analysis of the Descriptor model of all the datasets. Thus, both of the models find the *sp3* hybridized carbon substructures important and the therefore suggests predictions towards the lysosomotropic class in the presence of such substructures.

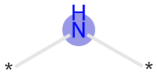
Bits	Substructures	Found important in:	Shapley values
2049		Training, Time-split, External	Positive
3959		Training, External	Positive
1028		Training, External	Positive
2715		Training, Time-split, External	Positive
3200		Training, Time-split, External	Positive

Figure 4.9.: Exemplary top bits and the key substructures they encode found important across different datasets by the Fingerprint model. Presence of these substructures almost always influences model results towards lysosomotropism.

Bit 2715, depending on its neighboring groups, might be encoding a secondary amine. This bit is important across all the three datasets. Bit 3200 encoding an aliphatic nitrogen atom is important across all three datasets. For both of these cases, presence of these substructures would impact the model output towards the lysosomotropic class. This is in line with the finding that basic pKa is consistently found as relevant descriptor in the SHAP analysis which is mostly driven by amine moieties.

The exemplar top bits and their substructures are shown in Table 4.9.

Chemical Structures and Descriptor Space Similarities

We performed chemical space similarity calculation, described in the Materials and Methods section, to ensure the chemical structure diversity between the training dataset and both of the validation sets. The corresponding ECDF plots are shown in Figure 4.10.

80% of the time-split dataset show a maximum Tanimoto similarity of less than 0.4 to the training set, whereas for the external dataset it is 0.3. This shows that the chemical spaces of both of the validation sets are diverse in comparison to the training set.

We also performed Principal Component Analysis (PCA) of the input descriptors of the combined datasets. However, only 19% explained variance ratio was observed in the first three principal components.

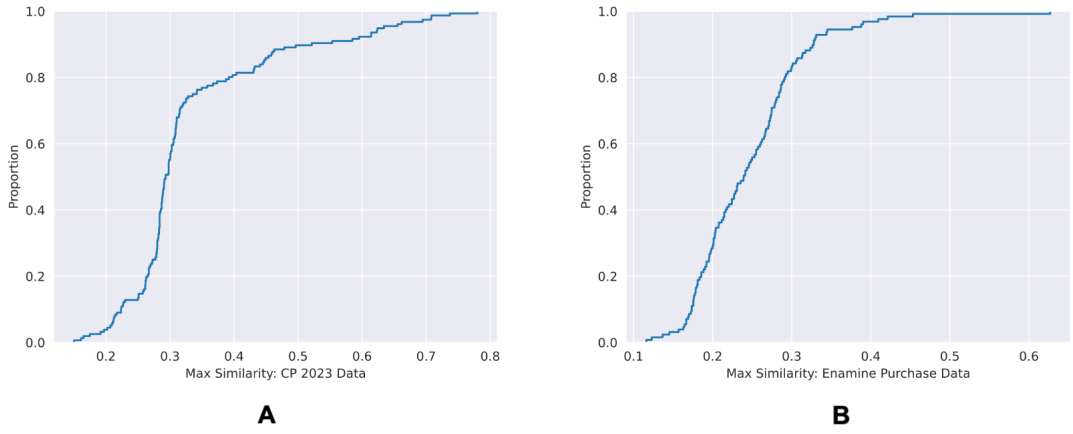


Figure 4.10.: ECDF plots of the maximum chemical structure similarities of time-split dataset (A) and the external dataset (B) with the training dataset.

4.4. Discussion

4.4.1. Lysosomotropic compounds tend to have higher logP

Even though logP was one of the two criteria for defining the PhysChem window used here, is the top descriptor in all 3 SHAP analyses of the Descriptor model performed on the 3 data sets (training, time-split, and the external validation). All of these analyses show a common trend that higher logP values tend to have higher SHAP values and *vice versa*. Such relation can be interpreted such that with the higher logP values, the model favors the lysosomotropic class, and similarly with the lower logP values, the model instead favors the non-lysosomotropic class.

This relation between logP values and lysosomotropism can also be noticed in the violin plot of the original lysosomotropic class distribution versus the logP values (Fig. 4.11). Here, across all the three datasets, the lysosomotropic compounds tend to have higher logP values compared to the non-lysosomotropic ones.

4.4.2. Non-lysosomotropic compounds tend to have lower molecular weights

The descriptor *HeavyAtomMolWt* was found important in the SHAP analyses of all the datasets and it was noted that the lower molecular weights have negative Shapley values. This means such compounds are more likely to predicted non-lysosomotropic.

This relationship between molecular weight and non-lysosomotropism can be observed in the violin plot of the original lysosomotropic class distribution versus the molecular weight of the compounds (Fig. 4.12).

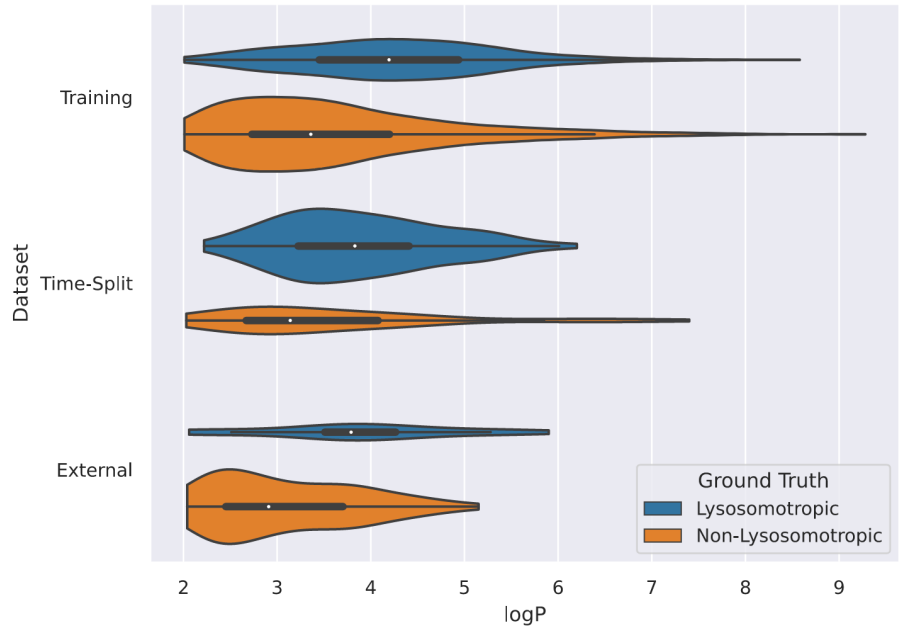


Figure 4.11.: Violin plot of logP values across all datasets.
The lysosomotropic classes are based on the cutoff of the compounds' lyso score. The violins are scaled based on the number of observations.

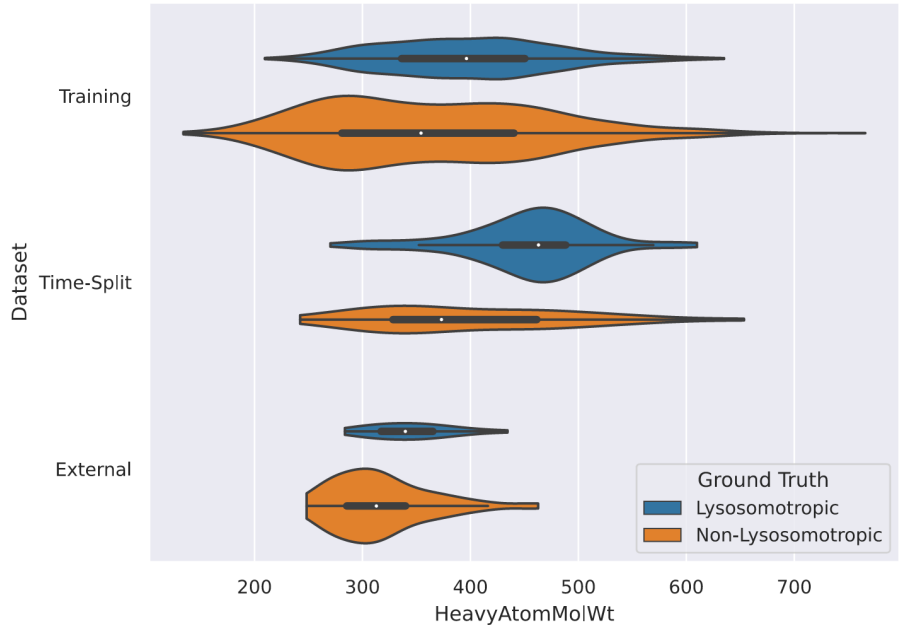


Figure 4.12.: Violin plot of heavy atom molecular weight values across all datasets.
The lysosomotropic classes are based on the cutoff of the compounds' lyso score. The violins are scaled based on the number of observations.

4.4.3. sp3-hybridized carbon atoms are found important

Different fingerprint bits encoding different *sp3*-hybridized carbon substructures are found important across all datasets by the Fingerprint model. Furthermore, the descriptor *FractionCSP3* is found important by the Descriptor model for all the datasets. The presence of *sp3*-hybridized carbon substructures is noted to impact the model output towards the lysosomotropic class, and *vice-versa*, in all of the cases.

4.4.4. Basic moieties are found important

It is consistently found by the descriptor-based SHAP analysis as well as by the *X-FP* analysis that basic moieties are driving a compound to be lysosomotropic. Though we restricted our selected compounds to be within a certain PhysChem window (logP $>$ 2, bpKa between 6.2 and 11), this lead to a restriction of the entire data towards basic moieties. However, the XML still might not have been capable of identifying this bias, but they correctly identified the fact that basic moieties are relevant for lysosomotropism.

4.4.5. Key substructures

No key substructures which induce either lysosomotropism or non-lysosomotropism were found here. In the MMPA, we do not identify any dominant chemical substructures which determine lysosomotropism in our data. Similarly, while examining the Fingerprint model with *X-FP*, the important bits usually encode *sp3*-hybridized carbon substructures and basic nitrogen substructures. We hypothesise that these substructures influence lysosomotropism by affecting the overall physicochemical properties of the compounds instead of causing specific chemical structure-based activity.

Many important bits encode substructures of the Morgan fingerprint radius of 0. This means that these substructures are single atoms, and thereby too unspecific to hypothesize any chemical structure-based activity.

4.4.6. The model performance

The Descriptor model significantly outperforms the Fingerprint model. Possible reasons for this initially unexpected drop in performance of the Fingerprint model could be as follows.

First, as highlighted in the previous section, due to the absence of any key substructures inducing lysosomotropism, the substructures found important by the Fingerprint model might not be specific enough to differentiate between lysosomotropic and non-lysosomotropic compounds. Furthermore, if lysosomotropism is mainly driven by logP and bpKa, then it will hard to find any substructure motifs that drive a logP/bpKa change

which in turn modulates lysosomotropism. This is due to the fact the very same (small, because it only considers the neighbors up to two bonds apart) substructure detected by a fingerprint can have very different logP/bpKa values based on their decoration.

The chemical diversity of the training and the test sets form a challenge to the performance of the Fingerprint model. This is especially pronounced when the presence or absence of the individual substructure (encoded as a bit in the fingerprint) has an impact on the biological readout, is influenced by the neighboring groups, the position in the molecule or by the stereochemistry, which is ignored by the fingerprints employed here.

The reasonable performance of the Descriptor model compared to the Fingerprint model further supports the notion that the lysosomotropism is primarily caused by physicochemical properties of a compound.

4.4.7. Concentration dependency of lysosomotropism

While investigating concentration dependency of lysosomotropism was not the focus of this work, we observed that not all the compounds show lysosomotropism even when tested at higher compound concentrations.

Out of 1369 non-lysosomotropic compounds present in the combined training set and time-split set, 682 of them were also tested at higher compound concentrations of either 30μM and 50μM in the CPA in addition to the standard 10μM compound concentration. While approximately 40$\%$ of such compounds (277 compounds) show lysosomotropism at higher concentration, the remaining 405 compounds remain non-lysosomotropic. Moreover, out of these 405 compounds which stay non-lysosomotropic at both standard and higher concentrations, 255 of them are bioactive (induction $>$5).

Nadanaciva *et al.* also reported that the two non-lysosomotropic compounds present in the PhysChem window in their analysis, especially risperidone, did not show lysosomotropism even at the highest tested concentration of 150μM.[71]

Further investigation in this area might offer insight on this observation.

4.4.8. Limitations

It is impractical to determine the logP values and the bpKa values of the compounds in the wet-lab at a bigger scale, at least in an academic setup. Thus, in our study the values of these descriptors are predicted. These descriptors are influential in the study. First, the compound selection is based on the compounds' presence in the PhysChem window, which is the cross-section of the logP values and the bpKa values. Second, these descriptors were found as the most important descriptors by the Descriptor model. Therefore,

our analyses might be affected due to the differences in the true and the predicted values of these descriptors.

While our internal data is a compilation of diverse compounds over many years, it cannot cover the full chemical space. Our models are trained on this finite data and the hypotheses derived from these results are therefore limited to this representation.

The *lyso score* of 75 or above, indicating biosimilarity, is a hard cut-off value to determine the lysosomotropic class of the compounds. The compounds around this cut-off value will be biosimilar, however, their lysosomotropic classes will be different. This could affect our analyses.

Lastly, the SHAP analyses do not show causation for the ground truth, rather they show what features were found important by a model in a dataset. Since our models' performances are limited, the SHAP analyses are also not definitive.

4.4.9. Conclusions

We used the morphological profiling data from the in-house Cell Painting Assay (CPA) to identify lysosomotropism in small molecules. We applied the *lyso score* to quantitatively score lysosomotropism. We confirm that mere presence of a compound in the well-established cross-section window of logP and bpKa values does not suffice it to be a lysosomotropic compound. By performing a MMPA, we were not able to detect key substructures that can be made responsible for a lysosomotropic effect.

Our ML models were trained on the internal dataset and tested on diverse validation sets. This ensured that these models' predictions and, by extension, the interpretable analyses done on them are general and not specific to the training set. These models revealed that the lysosomotropic effect is favored when the compounds have high lipophilicity, basicity, high number of basic amines and high number of *sp3*-hybridized carbons, and low TPSA.

4.4.10. LysoPredictor Web App

A user-friendly web application to predict lysosomotropism by small compounds is available on the CzodrowskiLab website.[110] This web application uses the Descriptor model in the backend to make predictions.

4.5. Materials and Methods

4.5.1. Data Selection for MMPA, and Model Training and Validation

The CPA protocol can be found in the methods section of Pahl *et al.* (2023).[21] In total, roughly 13,000 compounds were measured in the cell painting assay out of which roughly 450 compounds are known lysosomotropic active as reported in literature. These compounds are used as reference system for comparison of the cell painting profiles of the remaining compounds. Out the overall 13,000 tested compounds, 738 compounds are inside the PhysChem window and lysosomotropic. But there are 1,327 compounds are inside the PhysChem which are not lysosomotropic. A filtering strategy was employed on the internal CP data set till 2022. 10 μM compound concentration was selected. Toxic compounds and those flagged with purity alerts were excluded. Compounds with heavy atom count exceeding 50 were removed. Compounds whose calculated *logP* and *bpKa1* values were empty were removed. Final selection was then made on the compounds present in the PhysChem window - *logP* value greater than 2 and *bpKa1* value greater than 6.2 and less than 11. *apKa1* and *apKa2* were removed since they were redundant in the context of this study. In total, 2065 compounds were available for MMPA, and model training and cross validation.

The *bpKa1* limit of 6.5 from literature was lowered to 6.2 to enable the inclusion of a larger group of lysosomotropic compounds (53 compounds) present in the *bpKa1* range between 6.5 and 6.2.

Same steps were repeated to obtain compounds for the time-split validation set on the 2023 dataset.

Details about selected compounds from the time-split validation and the training data can be found here:[13,14,17,18,111] .

The corresponding code for the filtering strategy is available on GitHub.[112]

4.5.2. The mmpdb Algorithm

The MMPDB algorithm (version 2) published by Dalke *et al.* (2018),[93] which is freely available from the RDKit repository, was employed with default parameters. The program takes a set of SMILES of the compounds under analysis as input and outputs SMIRKS describing the molecular transformation for each MMP identified.[99] In brief, it consists of a three-step-procedure: Firstly, the compounds in the data set are decomposed into fragments following a well-defined protocol to derive all possible constant and variable parts, secondly all fragments are indexed and systematically compared to identify common substructures.[94,99] Lastly, the transformation rules are derived by evaluating all the MMPs with the same transformation (same variable A and B) at a specific

environment radius and aggregation of the associated property changes.[113]

4.5.3. Explainable Machine Learning

Molecular Descriptors and Molecular Fingerprints Calculations

All the molecular descriptors, except logP and bpKa, were calculated by the RDKit. logP and bpKa were calculated by ChemAxon Marvin *cxcalc* version *22.22.0* (https://www.chemaxon.com).

All the molecular fingerprints were generated by the RDKit. The Morgan fingerprints were calculated as bit vectors and the option to save the bit info was enabled to perform the SHAP analysis of the substructures using *X-FP*.

Modelling

The corresponding code for the modelling and cross-validation is available on GitHub.[112]

Chemical Space Similarity Calculation

The Morgan fingerprint of radius 4 were used to generate structural information of the compounds. Tanimoto similarity was used to compare fingerprints of two compounds, with the scale of similarity between 0 and 1 where 0 indicates no similarity and 1 indicates complete similarity.

For each compound present in a query dataset, chemical structure similarity against all of the compounds in the training dataset was calculated. The compound present in the training data which is structurally most similar to the query compound would therefore give the highest similarity score, and this score would be the query compound's maximum similarity score against the training dataset.

By plotting the maximum similarity scores of all the compounds of the query dataset as an ECDF plot, the percentage similarity to the training dataset can be observed.

External Validation

The *Liquid Stock Collection* from February 2023 comprising approximately 270 thousand compounds from the vendor Enamine was chosen. A compound standardization was performed which involved removal of compounds having a minimum heavy atom count of 20 and maximum heavy atom count of 50, *MedChem* filters, structure canonicalization, and duplicate removal.[114] Next, common compounds present in the internal MPI dataset

were removed. logP and bpKa were calculated by ChemAxon Marvin *cxcalc* version *22.22.0* (https://www.chemaxon.com) - the compounds present outside the PhysChem window were removed. To remove promiscuous compounds from the data in the following step, filtering proposed by Novartis Institutes of BioMedical Research (NIBR), which includes filters for the PAINS filter families A and B, was performed.[115,116]

The purchased compounds have at least a purity of 90 percent, measured by Liquid Chromatography-Mass Spectrometry (LC-MS). More details can be found in the Supplementary Information.

Maximum similarity calculation was performed against the internal MPI dataset and diverse compounds were short-listed. The original lysosomotropic class ratio present in the training dataset (65% non-lysosomotropic and 35% lysosomotropic) was aimed to be maintained, therefore, 127 compounds where 81 as non-lysosomotropic and 46 as lysosomotropic compounds predicted by the Descriptor model were short-listed and ordered. For these 127 compounds, the Fingerprint model classified 70 as non-lysosomotropic and 57 as lysosomotropic.

4.6. Acknowledgement

We thank Dr. Slava Ziegler for proofreading the manuscript and for valuable discussion, and Vinith Kowrithasan for the front- and back end development and maintenance of the LysoPredictor web-app.

5. Induction Web-App

5.1. Introduction

A CP profile of a compound contains information about morphological perturbations on cells exhibited by that compound. In relation with the CPA, such perturbations can be referred to as the bioactivity of a compound. However, different compounds perturb the cell morphology differently and to a various degrees. Putting together, a compound which perturb cell morphology greatly in comparison to another compound which does not, has a higher bioactivity than the other.

Induction, a measurement of compound activity in the CPA, was first reported by Christoforow *et al.* (2019).[13] It is the percentage of significantly CP altered features, and is calculated using the following formula:

$$\text{Induction } [\%] = \frac{\text{number of features with absolute values} > 3}{\text{total number of features}}$$

Compounds are considered bioactive, or simply, active in the CPA, if their induction is 5% or higher. This value of 5% or higher is an indication that meaningful morphological perturbations are induced by a compound on cell morphology.[1]

As shown previously in Section 1.1.2 and Figure 1.6, in the latest CPA processed dataset (June 26, 2023), out of 14,837 compounds, 4,778 are CPA active, while 10,059 are CPA inactive.

It is desirable for chemists to predict the induction of a (novel) compound before synthesizing it, because only bioactive compounds can be further analyzed in the CPA, primarily for target/MoA identification. Compound synthesis can be time and resource intensive, and having a predicted induction can allow chemists to prioritize their compounds for synthesis. This work discusses the development of a web-based application that uses a ML model in the background to predict whether a compound queried by a user can be CPA active or not.

5.2. Results

5.2.1. Model Performance

A binary DNN classifier was trained. The balanced accuracy of 0.76, and Cohen's Kappa of 0.46 was noted on the randomly split test set. The confusion matrix of this model's performance is shown in Table 5.1.

Table 5.1.: Confusion Matrix: Induction Model Performance

	Predicted Negative	**Predicted Positive**
Actual Negative	775	203
Actual Positive	652	1786

5.2.2. Front- and Backend

The web-app uses open-source Python libraries - Panel (v 0.12) and Param (v 1.12), both developed by the HoloViz community. Panel is a web-app framework - it allows creating applications with customizable user-interface elements like sliders, drop-down menus, buttons, etc. Param is a library which allows Python classes to have dynamically configurable parameters. Used in conjugation with Panel, Param is used to define or control properties of the components, such as triggering an activity when "Submit" button is pressed.[117]

Panel is used as the web-app framework at MPI Dortmund. With this induction web-app also developed in Panel, compatibility with the existing framework was maintained. Screenshots of the web-app are shown in Figure 5.1.

5.3. Methods

5.3.1. Data Processing

CPA data from December 2020 was accessed and processed as discussed in Section 2.2. The total number of compounds present in the processed dataset were 14,235, of which 10,152 were CPA inactive, and the rest were active.

Random train-test set split of 70%-30% was performed and stratified at the induction class to ensure splits with same class ratios. Test set was further randomly split with the ratio of 80%-20% into test set and validation set.

For model input, one- and two-dimensional molecular descriptors were calculated using RDKit (in total 196), and MF of radius 3 and 2048 bits were generated. The descriptors

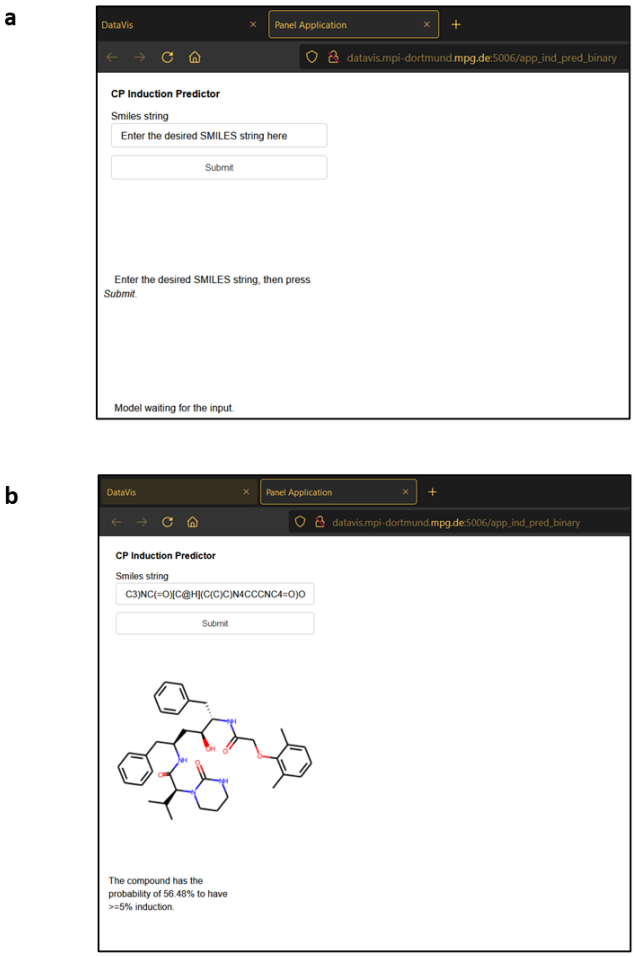


Figure 5.1.: Screenshots of the Induction Web-App hosted internally at MPI Dortmund.
a) Landing page.
b) Once user enters a valid SMILES string, a 2D structure of the compound is shown along with its prediction probability to show induction.

were scaled using Scikit-Learn's 'MinMaxScaler'. The scaled descriptors and MF bits were concatenated.

5.3.2. Model Architecture

A DNN model of architecture as shown in Figure 5.2 was trained. Softmax was used as the activation function in the last layer. Categorical cross-entropy was used as the loss function and ADAM as the optimizer. Class weights were introduced during the model training to control the class imbalance.

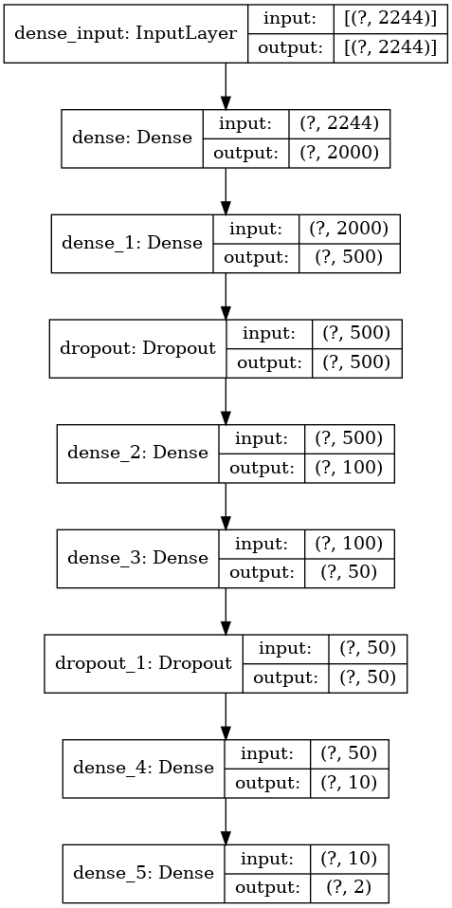


Figure 5.2.: Model architecture of the induction classifier DNN.

5.3.3. Front- and Backend

Saved DNN model, saved scalar - to scale the descriptors based on the training set, RDKit descriptors (196 of them) are loaded. Model is configured to be run only on CPU. Once user enters a SMILES, this SMILES is checked for validity. An error is returned if invalid input is provided. Once SMILES is validated, the 'X' is created - RDKit descriptors are created and scaled, and MF of radius 3 and bits 2048 are created, and concatenated. Since softmax activation is used in the last layer, a prediction probability is generated. The prediction probability of the 'CP Active' is returned to the user.

A Python class object is setup with Param parameters. A button object is defined as a Param event object. A view is setup which prompts user to input valid SMILES. Once the button is triggered by the user, the above described process is executed. If the SMILES is valid, a 2D structure of the compound is also returned.

5.4. Outlook

For chemists synthesizing new compounds and using CPA to predict their target/MoA, it is very useful to be able to conveniently predict whether such compounds might be active in the CPA during the planning phase. For this purpose, an easy-to-use web-app was developed on the existing internal platform. Such application allows chemists to use ML in their work to save time and resources, without any technical *know-hows*. To this end, they are provided with prediction probabilities of their queries, allowing them to discern the model prediction.

6. eXplainable FingerPrints (X-FP)

6.1. Introduction

In drug discovery, it is of great interest to have accurate models for predicting the activities of compounds while rationalizing these predictions for several reasons. Medicinal chemists appreciate such transparency because it allows them to trust these predictions. Cheminformaticians and data scientists also benefit from transparency, as it allows them to check against the "clever Hans effect" - models that make correct predictions for wrong reasons - and to further improve and debug their models. In addition, it allows researchers in general to perform knowledge mining - hypothesizing structure-property relationships based on model interpretation.[32,118,119]

The earliest QSAR models, developed around fifty years ago, were multiple linear regression models. Other methods, such as partial least-squares and decision trees were later utilized as well. Due to the their inherent nature, such models' predictions were comprehensible.[32] While these models offered mechanistic insight, they often lacked predictive power. Methods with higher predictive power, such as NNs, support vector machines, and ensemble tree methods, etc., were later utilized for QSAR. The latter methods were termed "black box" models as their predictions cannot be interpreted in a straight-forward manner. Generally, given such polarity - interpretable models with inferior accuracy and black box models with higher accuracy, a trade-off relation between model interpretability versus accuracy was suggested.[120]

The terms "interpretable/interpretability" and "explainable/explainability" are often used interchangeably (and will be in this work). However, there are certain differences between them that have been reported in the literature. In a nutshell, "interpretability" can be referred to as the ability to understand the decision-making process of a model by a human, whereas "explainability" can be referred to as the ability to understand the output of a model post hoc.[119] Decision trees and linear regression provide insight into model output and can be called "interpretable". Using feature attribution methods, such as SHAP, to understand the model output for a NN is therefore "explainable".

As models with high predictive power are becoming common in different aspects of drug discovery - QSAR, molecular design, chemical synthesis planning, protein structure prediction, etc. - XAI methods are also gaining attraction.[118] Shapley Additive Explanations (SHAP), proposed by Lundberg and Lee (2017), is a feature attribution method used for

XAI. It finds roots in game theory, where contributions of individual players were measured to a collaborative game. Over the last few years, use of SHAP is gaining grounds in XAI for QSAR in drug discovery.[83,118] More details of use of SHAP for feature importance are present in Section 4.3.3.

Molecular fingerprints such as Morgan Fingerprints (MF) are very popular for QSAR modeling in drug discovery.[121] However, performing XAI on ML models with MF as bit vectors as input is challenging. First, MF as bit vectors are dynamic - depending on the bit vector length and radius provided by the user, different substructures can be encoded on different bits. Second, multiple substructures can be encoded by the same bit. This phenomenon is known as bit collision. The frequency of such substructures depends mainly on the bit vector length and the chemical structure diversity of the data set. Thus, despite calculating feature importance to identify important bits using any relevant feature importance method, there is no straightforward way to identify the key substructure(s) and thus associate it with the activity in question. Finally, there is a lack of open-source, user-ready software that addresses these challenges while providing an intuitive summary for chemists and cheminformaticians alike.

This project introduces X-FP (eXplainable FingerPrints), an open-source Python package for performing explainable machine learning on MF-based models. The operation of X-FP is described in the following steps. For the given data set, it extracts the substructures encoded in the MF bits as accurate, canonical SMARTS. These substructures, along their frequencies, are linked to the results of the feature importance methods, such as SHAP. These results are presented in a detailed yet intuitive PDF report.

A validation study was conducted to see if X-FP could assist a chemist with explainable machine learning. Five well-established drug targets were selected. Data from CHEMBL was then accessed and compounds tested for their activity for these targets were selected. After preprocessing the data and binning the compounds as active or inactive, binary classification models were trained to predict activities using MF as input for each target. The best models were then subjected to X-FP analysis. For each model, the substructures encoded by the bits found to be important were visualized. These substructures were compared with the literature data for the corresponding targets, and the findings were discerned.

X-FP was developed by the author and Marcel Baltruschat. Hanna Rieger performed the validation study under the supervision of the author.

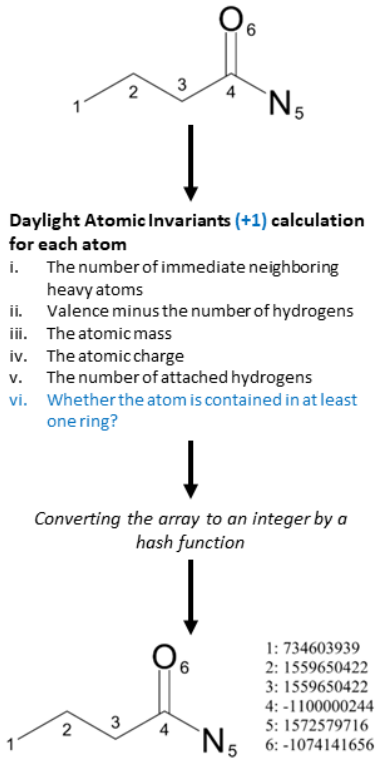


Figure 6.1.: Assignment of initial atom identifiers. Reprinted (adapted) with permission from Rogers, D., & Hahn, M. (2010). Extended-Connectivity Fingerprints. Journal of Chemical Information and Modeling, 50(5), 742–754. Copyright 2010 American Chemical Society.

6.2. Background

6.2.1. Morgan Fingerprints

Extended-connectivity fingerprints are circular fingerprints that capture molecular substructure information. They were first introduced by Accelrys in Pipeline Pilot in 2000.[38] In the open-source cheminformatics software library RDKit, their implementation is called Morgan Fingerprints (MF).[102] From now on, the term MF will be used interchangeably. MFs are well-established in cheminformatics and are used in numerous applications such as virtual screening, similarity search, used as model input for QSAR/ML methods, etc.[38] While the details of how MFs are computed are reported by Rogers and Hahn (2010), this section provides a summarized version.

In the first stage, an initial integer identifier is assigned to each atom of the query compound as shown in Figure 6.1. The next stage involves iterative updating of the atomic identifiers as shown in Figure 6.2. In the last stage, duplicate identifiers are removed.

The final identifiers are then converted to an integer by the hashing function. However,

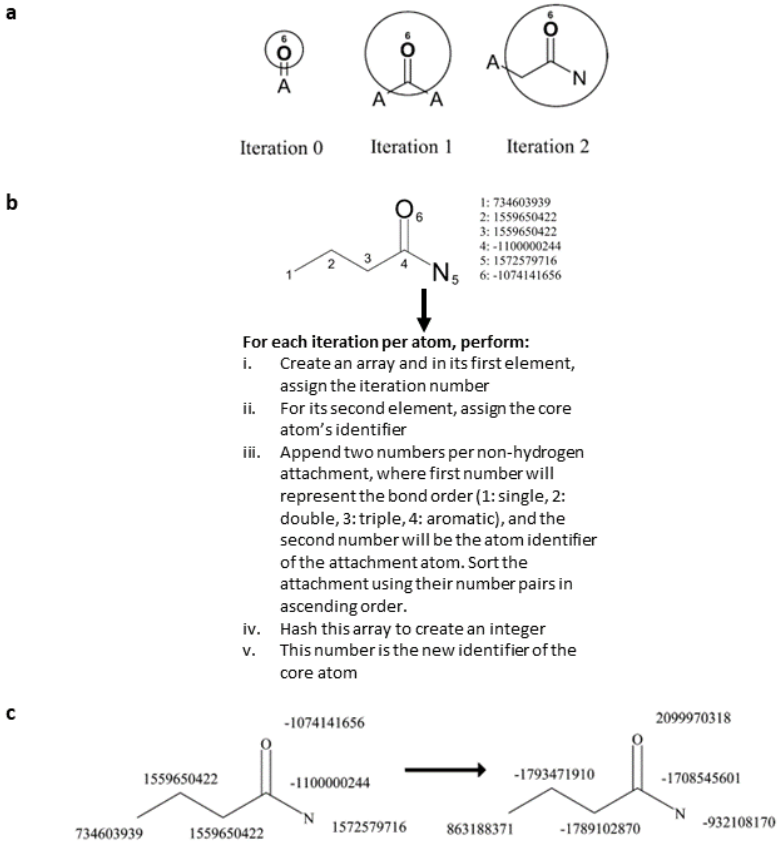


Figure 6.2.: A Morgan Fingerprint iteration
a) Effect of iteration process on information stored in the identifier of the oxygen atom in butyramide.
b) Updating the atom identifiers per iteration. After each iteration, each atom contains additional information about its neighbors as this neighbor's identifier will also gain more information per iteration.
c) Illustration of updated atom identifiers after an iteration. These new identifiers will be used in the next iteration.
Reprinted (adapted) with permission from Rogers, D., & Hahn, M. (2010). Extended-Connectivity Fingerprints. Journal of Chemical Information and Modeling, 50(5), 742–754. Copyright 2010 American Chemical Society.

it is also possible to "fold" such sparse vectors to a fixed bit vector length. While there is no consensus on this length, 1024, 2048, 4096 are commonly used lengths.

6.2.2. Drug Targets

hERG

human *Ether-a-go-go-Related* Gene (hERG) is a gene which codes for potassium-ion channel protein, called *Kv11.1*. This protein (also called hERG channel) plays an important role in cardiac action potential. Its inhibition may cause a disorder called long QT syndrome where there is an abnormally long interval between the start of the Q wave and the end of the T wave on an electrocardiogram. This abnormal interval can cause arrhythmia which can even lead to death. While the long QT syndrome may also occur due to genetic disorders, one such arrhythmia, called *Torsades de pointes* or TdP is specifically caused by certain drugs. Numerous drugs like sertindole, grepafloxacin and terfenadine were withdrawn from the market for causing drug-induced QT syndrome.[122,123]

Due to such severe adverse effects, hERG is a prominent target in medicinal chemistry as it is important to access the 'hERG liability' of the candidate compounds in the early drug discovery.

Cyclooxygenase 1 and 2

Cyclooxygenase is a key enzyme to convert arachidonic acid to proinflammatory prostaglandins. Non-steroidal Anti-inflammatory Drugs (NSAIDs) are drugs which are primarily used for anti-inflammation and pain relief. NSAIDs primarily target cyclooxygenase as their inhibition limits the production of prostaglandins.[124]

Cyclooxygenase has 2 isoforms - Cyclooxygenase 1 (COX-1) and Cyclooxygenase 2 (COX-2). COX-1 is primarily reported to have 'housekeeping' functions. In their review, Dubois *et al.* (1998) reported that apart from causing inflammation, prostaglandins in humans also regulates "blood clotting, ovulation, initiation of labor, bone metabolism, nerve growth and development, wound healing, kidney function, blood vessel tone and immune responses".[124] Such 'housekeeping' functions appear to be mediated by COX-1. On the contrast, inflammatory functions are reported to be primarily caused by COX-2.

Due to such crucial functions, COX-1 and COX-2 are important and well-studied drug targets.

Cyclin-dependent Kinase 2

Cyclin-dependent kinase 2 (CDK2) is a protein kinase enzyme, which is a core cell cycle regulator. It is active from the late G1 phase to the end of S-phase of the cell cycle. It is

also involved in DNA synthesis and G2 phase progression modulation. CDK2 is found to be significantly overactivated in many cancers. Abnormal activation of CDK2 leads to uncontrolled cell proliferation during carcinogenesis.[125] CDK2 is therefore an attractive anti-cancer drug target.

Thrombin

Thrombin is a serine-protease which plays a crucial role in blood coagulation cascade. It is generated from its precursor, prothrombin, by the activation of another enzyme, Factor Xa. In the coagulation cascade, thrombin has numerous activities, like conversion of soluble fibrinogen to insoluble fibrin strands. Apart from its functions in hemostasis, it is also acts as an anticoagulant. Due to these properties, it is an important drug target against the thromboembolic diseases and for the discovery of novel antithrombotic agents.[126]

6.3. Methods

6.3.1. X-FP: Software Design

Note: The terms molecular 'substructure' and molecular 'fragment' are often used interchangeably to define a section of interest in the structure of a compound. However, in this work, 'substructure' is used to define a chemical entity, while 'fragment' is used in a programming context - a part of an RDKit molecule object that has a root atom and an atomic environment of a fixed radius around it.

The intended users of X-FP are cheminformaticians and ML practitioners. While the output report it generates is designed to be intuitive for medicinal chemists as well. The functional part of the software can be figuratively divided into three modules. These modules are:

Fingerprint Manager

Fingerprint Manager is the core module where all MF related operations are performed. A class "FingerprintManager" is declared, in which all operations are performed. The operations are described in detail below.

1. Generation of MF

Here, user-defined configuration of MF bit vectors in *numpy* array format are generated. These configurations are:

- MF radius (default: 2)

- Number of bits (default: 2048)

- Should Feature MF be generated instead? (default: False)

- Should chirality be taken into account when generating MF? (default: False)

From now on, the term "MF" will refer to MF bit vectors in *numpy* array format, unless otherwise specified. The bit information is always determined during MF generation. Its use will be explained later.

The RDKit molecule objects from which MF are calculated are always canonized (RDKit-based). The user can provide input in the form of SMILES or RDKit molecule objects present in a Pandas DataFrame, or simply a single SMILES or RDKit molecule object at a time with the user configuration to generate MF. The SMILES are also always canonized.

2. Gathering information about on-bits

A dictionary is created that has bits as keys, and for values - each substructure encoded by that bit across the dataset. The latter information is stored as the RDKit molecule object, the atom index that sets the bit in that molecule object, and the radius around the atom index to capture that environment. The length of this dictionary is equal to the total number of bits set in the dataset.

3. Generating fragments from the substructure information

For further analysis, it is necessary to have the previously generated substructure information in a tangible form - such as SMARTS.

A nested loop approach is used, where the outer loop iterates over on-bits, and the inner loop iterates over a tuple containing the RDKit molecule objects, atom indices and the radii (values of the dictionary created in the previous step). The inner loop is run in a separate Python method.

In each iteration of the inner loop:

i. All the atoms present within the provided radius from the atom at the provided atom index in the provided molecule object, are determined.

ii. If connectivity variants are to be included (default: True), all atoms present within the next radius (provided radius + 1) around the atom index are determined. These atoms can be called the outer atoms. Note that the outer atoms do not include the atoms within the default radius, but only the neighboring atoms.

iii. Atom and bond symbols are calculated. If an atom is present in the outer atoms, a wildcard (∗) is introduced. Similarly, if bond is present in outer atoms, another wildcard () is introduced.

iv. Canonical SMARTS pattern are constructed from the fragments using the calculated atom and bond symbols from the provided RDKit molecule object, rooted at the provided atom index.

v. The newly constructed SMARTS pattern is checked against duplication.

The inner loop is the most computationally intensive step. However, it is possible to run it in parallel on multiple CPU cores if the user's machine has multiple CPU cores and they choose to do so.

In the end of this step, two dictionaries are created. First dictionary has on-bits as keys, and canonical SMARTS patterns (fragments) as values present per on-bit as values. Whereas the second dictionary has on-bits as keys, and RDKit molecule object, atom index and radius per on-bit as values.

4. Calculation of frequency of the fragments encoded by a query bit

For a query bit, a Pandas DataFrame is created in which the substructures / fragments encoded by this query bit are enumerated as canonical SMARTS. In addition, the total number of occurrences of each substructure - regardless of its frequency per compound - and its unique occurrences - present at least once per compound - are also present.

5. Drawing the fragments encoded by a query bit

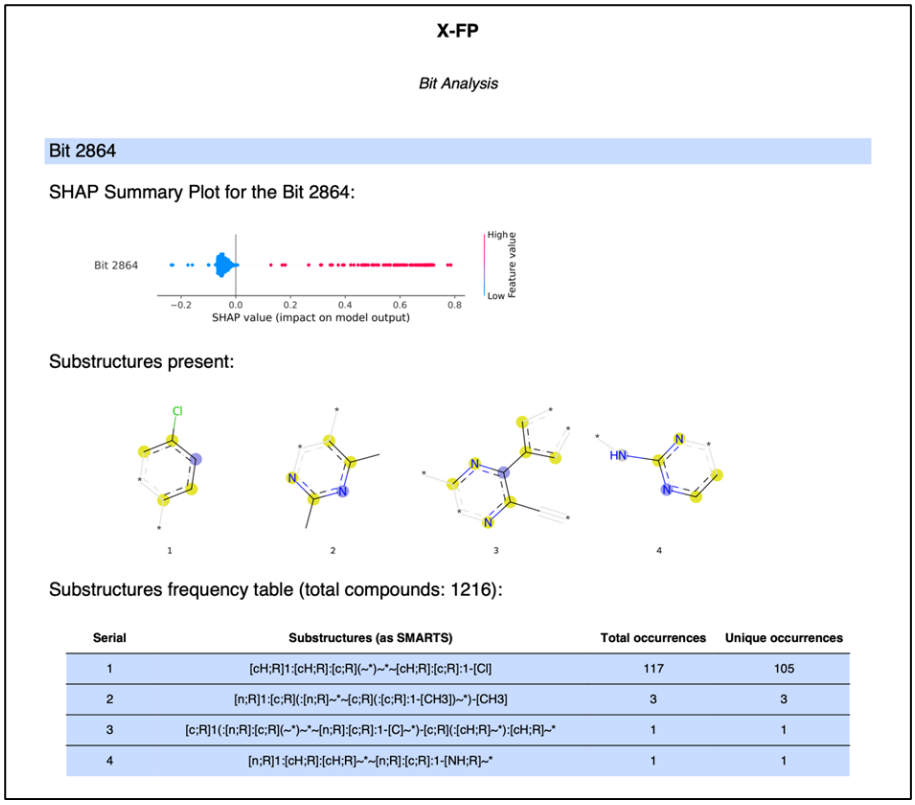
For a query bit, a SVG image is returned where all the substructures / fragments, encoded by this query bit, are present.

Report Maker Module

An important feature of X-FP is to provide modelers and medicinal chemists with a feasible yet detailed explanation of features. This report generation is accomplished by the Report Maker module. This report consists of an introduction page, chapters of bit analyses (one chapter per bit analyzed), and a custom end note providing further instructions to the user. Each chapter has 3 parts - the feature importance plot of the query bit, the 2D chemical structures of the substructures encoded by the query bit, and the frequency table of these substructures. An example of the chapter of the generated report is shown in Fig. 6.3.

This report generation module can be imported by any feature importance Python file as a parent Python class, allowing the latter to output its results as a PDF report. The Python library fpdf2 is used for PDF generation. The steps are described below.

A class called BitAnalysis is declared, which inherits FPDF from fpdf2 as a parent class. A header method is declared that displays the X-FP logo and the Czodrowski Lab logo on the cover page, and "X-FP" and "Bit Analysis" on the remaining pages. A footer method is declared that displays the user-defined report title, page number, and timestamp if requested. An intro page method is declared to generate an intro page of the report. This intro page always displays the overall feature importance plot, along with that plot's label, the custom report title, and the time stamp.



Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[cH;R]1:[cH;R]:[c;R](~*)~*~[cH;R]:[c;R]:1-[Cl]	117	105
2	[n;R]1:[c;R](:[n;R]~*~[c;R](:[c;R]:1-[CH3])~*)-[CH3]	3	3
3	[c;R]1(:[n;R]:[c;R](~*)~*~[n;R]:[c;R]:1-[C]~*)-[c;R](:[cH;R]~*):[cH;R]~*	1	1
4	[n;R]1:[cH;R]:[cH;R]~*~[n;R]:[c;R]:1-[NH;R]~*	1	1

Figure 6.3.: Example chapter on X-FP bit analysis.
Each chapter has 3 parts. This example shows the analysis of bit 2864.
The first part shows the feature importance plot of this bit. Here SHAP TreeExplainer is used for feature importance. This summary plot shows that when this bit is on, the model output is preferred to the positive class.
The second part shows the substructures encoded by this bit.
The third part is the frequency table showing the canonical SMARTS of these substructures along their occurrences.

The remaining pages are bit analyses. For each query bit, one method generates chapter titles while another prints the rest of the chapter details, as shown in Figure 6.3.

Feature Importance Module

X-FP allows advanced users to add feature importance methods such as SHAP, LIME, etc. in a specified way as a Python file to be used in X-FP. These feature importance methods, when added in such a way, encompass as the feature importance module.

In general, the workflow is as follows. Necessary packages and X-FP modules are imported. Then a new class is declared, which inherits the 'FingerprintManager' as a parent class. Next, the feature importance of each fingerprint bit is calculated. Based on the bits queried by the user, feature importance plots for those bits are generated individually, along with the top bits. Finally, the PDF report is generated using the 'BitAnalysis' class.

In the current version of X-FP (version 1.1.0 at the time of writing), SHAP TreeExplainer and XGBoost's built-in feature importance methods are included by default.

6.3.2. X-FP: Validation Study Design

Data Preparation and Preprocessing

Five exemplary, well-established drug discovery targets, as discussed in the previous section, were used in the validation study. CHEMBL was used as the data source for each target. The data for each target were curated as described in Volkamer Lab's TeachOpenCADD-T001.[127] Table 6.1 lists the targets, their UniProt IDs, and the date they were accessed from CHEMBL. For each target, the data consists of the compounds' CHEMBL ID, SMILES, IC50 values, and units of the IC50 values. The data were further standardized as shown by Axel Pahl.[114] This included retaining compounds with heavy atom numbers between 15 and 50, MedChem filtering, structure canonicalization, and duplicate removal.

pIC50 is the negative log of the IC50 value when converted to molar. To classify compounds as active or inactive for their respective targets, a pIC50 threshold of 5 was chosen. This corresponds to an IC50 of 10 μM.

Class separation for each target is shown in Table 6.2.

Modeling Strategy

The following modeling strategy was implemented for all targets.

MF as bits of radii 2, 3, and 4 were calculated as model inputs for all compounds. For each configuration of MF, two XGBoost binary classifiers were built - one with opti-

Table 6.1.: Targets used in the X-FP validation study, their UniProt IDs, and access date.

Target	Target ID (UniProt ID)	Date of Access
hERG	Q12890	19.04.23
COX-1	P23219	24.04.23
COX-2	P35354	03.05.23
CDK2	P24941	15.05.23
Thrombin	P00734	15.05.23

Table 6.2.: Summary of final data for each target

Target	Total	Actives	Inactives
hERG	6078	4061	2017
COX-1	1448	784	700
COX-2	3468	2816	652
CDK2	1473	1245	228
Thrombin	1499	1117	382

mized hyperparameters and one with default hyperparameters. The hyperparameter optimization was done using Optuna. A 'scale_pos_weight' argument, which takes the class weights to control class imbalance in the dataset, was provided to each model. A stratified train-test data splitting with a ratio of 80%-20% was performed.

The hyperparameter optimization strategy is as follows. A search space is defined for each hyperparameter. XGBoost Classifier model object is declared and stratified 5-fold cross-validation is performed. Balanced accuracy is calculated for each fold, and a mean balanced accuracy is determined once all 5 folds are modeled. Optuna is then used to optimize this mean balanced accuracy for 200 trials. With this hyperparameter optimization strategy, optimal hyperparameters were determined that give the best 5-CV performance. This optimization is performed only on the training data, allowing such models to be validated transparently on the test set.

6.4. Results

6.4.1. X-FP: Software Results

Open-Source Software Developed

X-FP is open source software - it is available to users and developers free of charge and encourages them to continue contributing to the development of the software. Therefore, it has several such features to help developers. It uses the MIT open source license, which allows developers to freely use, modify and distribute it. Most of the functionalities of the code are automatically tested using 'Pytest', making software maintenance

feasible. It has a contribution guide that provides developers with development guidelines. Each Python function/class method has a detailed docstring in 'numpydoc' style. *Sphinx* is used to automatically generate documentation from these docstrings. The code is formatted using the *Black* software library.[128] Since *Black* enforces a consistent code style, the readability of the code is improved.

User contributions are encouraged. Such contributions include reporting bugs, bug fixes, adding new features, improving existing documentation, improving code quality, improving existing automated tests, adding example use cases, etc.

Setup and Methodology

X-FP is available on GitHub (https://github.com/czodrowskilab/x-fp).

It can be conveniently installed locally by first cloning the GitHub repo, creating a conda-based environment (Anaconda, Miniconda, Mamba) using the provided YAML files, and then running the install command. Detailed instructions can be found in the "Installation" section of the README.MD file.

The minimal dependencies of the X-FP (version 1.1.0 as of writing) are Python, RDKit, Numpy, Pandas, Matplotlib, Seaborn, tqdm, FPDF2, JupyterLab, and SHAP.

6.4.2. X-FP: Validation Study Results

Model Metrics

Models' performances is shown in Table 6.3. All models are evaluated on random split test sets.

Important Bits and Substructures Identified

Note: The validation study was performed on a pre-release version of X-FP. The important difference between the current version and the pre-release version is that canonical SMARTS were not present in the generated report.

For each target, X-FP analysis was performed on its test set. The key substructures encoded by the most important bits were identified by their frequencies. Once these substructures were identified, they were compared to their activity for the target in the literature. Since SHAP TreeExplainer was used as the feature importance method in this study, it was also possible to identify how a bit, and by extension the key substructure it encodes, influences a prediction.

Table 6.4 summarizes the key substructures found per target and their scientific literature validation. In most cases, the key substructures that influence model predictions toward

Table 6.3.: X-FP validation study - performance of models on their respective test sets. Each row is a pair of models (with optimized hyperparameters and with default hyperparameters) trained on the MF of that radius. The best performing model for each target is highlighted.

Target	**MF**	**Tuned**		**Base**	
		Balanced Accuracy	Cohen's Kappa	Balanced Accuracy	Cohen's Kappa
hERG	MF2	0.698	0.406	**0.706**	**0.424**
	MF3	0.697	0.403	0.685	0.380
	MF4	0.695	0.401	0.695	0.400
COX1	MF2	0.767	0.521	0.731	0.461
	MF3	0.765	0.514	0.759	0.508
	MF4	0.781	0.542	**0.793**	**0.570**
COX2	MF2	0.704	0.454	0.715	0.477
	MF3	0.711	0.469	**0.716**	**0.478**
	MF4	0.713	0.475	0.707	0.456
CDK2	MF2	0.733	0.495	0.780	0.570
	MF3	0.745	0.527	**0.794**	**0.588**
	MF4	0.767	0.560	0.774	0.553
Thrombin	MF2	0.745	0.523	0.750	0.530
	MF3	0.761	0.557	0.774	0.582
	MF4	0.751	0.535	**0.777**	**0.585**

the positive class were also reported in the literature to cause inhibition of their respective targets. This good overlap between key substructures found using a XAI method and literature data is encouraging.

With the exception of hERG, key substructures affecting model predictions toward the negative class could not be verified in the literature. The main reason for this lack of verification was that very little information about such substructures was available. Since medicinal chemistry research is primarily focused on the identification of active compounds, their moieties and their mode of action, this limited information can be justified.

Table 6.4.: X-FP validation study - key substructures and their attributions. "Active" means that a substructure affects model prediction towards the positive class, which in this case means target inhibition. Opposite is true for the "inactive". Note that the sum of "Total" per target is 10, as top 10 bits were analyzed for each target.

Target	**Active**		**Inactive**	
	Total	Literature Verified	Total	Literature Verified
hERG	5	3	5	4
COX-1	6	2	4	1
COX-2	7	6	3	1
CDK2	6	5	4	0
Thrombin	5	4	5	0

Many 'simple' substructures were also identified as key substructures. These are usually encoded by MF of radius 1 and they contain very limited information - an atom and its bonds. In some cases, key substructures of higher radii which also contained these simple key substructures were present for the same target. While it is possible to switch off bits encoding such simple substructures while performing X-FP analysis, it was an active decision to **not** to do so in X-FP as such findings can (and should) be discerned by chemists. This avoids introducing bias in the analysis.

6.5. Future of X-FP

X-FP is intended to be an organic project. Some new features planned are the implementation of additional molecular fingerprints, the addition of more ready-to-use feature importance methods, and additional output implementations such as machine-readable output.

List of Abbreviations

ADAM	Adaptive Moment Estimation
AE	Autoencoder
bpKa	basic *pK*a
CDK2	Cyclin-dependent kinase 2
cGAN	Conditional Generative Adversarial Networks
clogP	calculated *logP*
COX-1	Cyclooxygenase 1
COX-2	Cyclooxygenase 2
CP	Cell Painting
CPA	Cell Painting Assay
DCM	Dark Chemical Matter
DHODH	Dihydroorotate Dehydrogenase
DNN	Deep Neural Network
ELBO	Evidence Lower Bound
GAN	Generative Adversarial Networks
GRU	Gated Recurrent Unit
hERG	human *Ether-a-go-go-Related* Gene
IC50	Half Maximal Inhibitory Concentration
KL	Kullback-Leibler
LC-MS	Liquid Chromatography-Mass Spectrometry
LIME	Local Interpretable Model-Agnostic Explanations
logP	Octanol/Water-Partition Coefficient
LTR	LysoTracker Red DND-99
MAE	Mean Absolute Error
MF	Morgan Fingerprints
ML	Machine Learning
MMP	Matched Molecular Pair
MMPA	Matched Molecular Pair Analysis
MoA	Mode of Action
MSE	Mean Squared Error
NN	Neural Network
NP	Natural Products
NSAIDs	Non-steroidal Anti-inflammatory Drugs
PAINS	Pan Assay Interference
PCA	Principal Component Analysis
PNP	Pseudo-Natural Products
QED	Quantitative Estimate of Drug-Likeness

QSAR	Quantitative Structure-Activity Relationship
ReLU	Rectified Linear Unit
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
SAG	Smoothened agonist
SAR	Structure Activity Relationship
SAS	Structure-Activity Similarity
SELFIES	Self-Referencing Embedded Strings
SHAP	Shapley Additive Explanations
SMARTS	SMILES Arbitrary Target Specification
SMILES	Simplified Molecular Input Line Entry System
SPR	Structure Property Relationship
SVG	Scalable Vector Graphics
TPSA	Topological Polar Surface Area
VAE	Variational Autoencoder
XAI	Explainable Artificial Intelligence
XGBoost	Extreme Gradient Boosting
XML	Explainable Machine Learning

Bibliography

[1] S. Ziegler, S. Sievers, H. Waldmann, "Morphological profiling of small molecules", *Cell Chemical Biology* **2021**, *28*, 300–319.

[2] M. A. Bray, S. Singh, H. Han, C. T. Davis, B. Borgeson, C. Hartland, M. Kost-Alimova, S. M. Gustafsdottir, C. C. Gibson, A. E. Carpenter, "Cell Painting, a high-content image-based assay for morphological profiling using multiplexed fluorescent dyes", *Nature Protocols* **2016**, *11*, 1757–1774.

[3] S. M. Gustafsdottir, V. Ljosa, K. L. Sokolnicki, J. A. Wilson, D. Walpita, M. M. Kemp, K. P. Seiler, H. A. Carrel, T. R. Golub, S. L. Schreiber, P. A. Clemons, A. E. Carpenter, A. F. Shamji, "Multiplex cytological profiling assay to measure diverse cellular states." *PloS one* **2013**, *8*, e80999.

[4] J. C. Caicedo, S. Singh, A. E. Carpenter, Applications in image-based profiling of perturbations, **2016**.

[5] S. N. Chandrasekaran, H. Ceulemans, J. D. Boyd, A. E. Carpenter, "Image-based profiling for drug discovery: due for a machine-learning upgrade?", *Nature Reviews Drug Discovery* **2021**, *20*, 145–159.

[6] M. H. Rohban, S. Singh, X. Wu, J. B. Berthet, M. A. Bray, Y. Shrestha, X. Varelas, J. S. Boehm, A. E. Carpenter, "Systematic morphological profiling of human gene and allele function via cell painting", *eLife* **2017**, *6*, 1–23.

[7] J. Simm, G. Klambauer, A. Arany, M. Steijaert, J. K. Wegner, E. Gustin, V. Chupakhin, Y. T. Chong, J. Vialard, P. Buijnsters, I. Velter, A. Vapirev, S. Singh, A. E. Carpenter, R. Wuyts, S. Hochreiter, Y. Moreau, H. Ceulemans, "Repurposing High-Throughput Image Assays Enables Biological Activity Prediction for Drug Discovery", *Cell Chemical Biology* **2018**, *25*, 611–618.e3.

[8] M. Hofmarcher, E. Rumetshofer, D. A. Clevert, S. Hochreiter, G. Klambauer, "Accurate Prediction of Biological Assays with High-Throughput Microscopy Images and Convolutional Networks", *Journal of Chemical Information and Modeling* **2019**, *59*, 1163–1171.

[9] N. Moshkov, T. Becker, K. Yang, P. Horvath, V. Dancik, B. K. Wagner, P. A. Clemons, S. Singh, A. E. Carpenter, J. C. Caicedo, "Predicting compound activity from phenotypic profiles and chemical structures", *Nature Communications* **2023**, *14*, DOI 10.1038/s41467-023-37570-1.

[10] S. Seal, H. Yang, L. Vollmers, A. Bender, "Comparison of Cellular Morphological Descriptors and Molecular Fingerprints for the Prediction of Cytotoxicity- And Proliferation-Related Assays", *Chemical Research in Toxicology* **2021**, *34*, 422–437.

[11] S. Seal, J. Carreras-Puigvert, M. A. Trapotsi, H. Yang, O. Spjuth, A. Bender, "Integrating cell morphology with gene expression and chemical structure to aid mitochondrial tox-

icity detection", *Communications Biology* **2022**, *5*, DOI 10.1038/s42003-022-03763-5.

[12] S. J. Warchal, J. C. Dawson, E. Shepherd, A. F. Munro, R. E. Hughes, A. Makda, N. O. Carragher, "High content phenotypic screening identifies serotonin receptor modulators with selective activity upon breast cancer cell cycle and cytokine signaling pathways", *Bioorganic and Medicinal Chemistry* **2020**, *28*, DOI 10.1016/j.bmc.2019.115209.

[13] A. Christoforow, J. Wilke, A. Binici, A. Pahl, C. Ostermann, S. Sievers, H. Waldmann, "Design, Synthesis, and Phenotypic Profiling of Pyrano-Furo-Pyridone Pseudo Natural Products", *Angewandte Chemie International Edition* **2019**, *58*, 14715–14723.

[14] D. J. Foley, S. Zinken, D. Corkery, L. Laraia, A. Pahl, Y.-W. Wu, H. Waldmann, "Phenotyping Reveals Targets of a Pseudo-Natural-Product Autophagy Inhibitor", *Angewandte Chemie International Edition* **2020**, *59*, 12470–12476.

[15] T. Schneidewind, A. Brause, A. Pahl, A. Burhop, T. Mejuch, S. Sievers, H. Waldmann, S. Ziegler, "Morphological Profiling Identifies a Common Mode of Action for Small Molecules with Different Targets", *ChemBioChem* **2020**, *21*, 3197–3207.

[16] T. Schneidewind, A. Brause, B. Schölermann, S. Sievers, A. Pahl, M. G. Sankar, M. Winzker, P. Janning, K. Kumar, S. Ziegler, H. Waldmann, "Combined morphological and proteome profiling reveals target-independent impairment of cholesterol homeostasis", *Cell Chemical Biology* **2021**, *28*, 1780–1794.e5.

[17] J. Liu, G. S. Cremosnik, F. Otte, A. Pahl, S. Sievers, C. Strohmann, H. Waldmann, "Design, Synthesis, and Biological Evaluation of Chemically and Biologically Diverse Pyrroquinoline Pseudo Natural Products." *Angewandte Chemie (International ed. in English)* **2021**, *60*, 4648–4656.

[18] M. Grigalunas, A. Burhop, S. Zinken, A. Pahl, J.-m. Gally, N. Wild, Y. Mantel, S. Sievers, D. J. Foley, R. Scheel, C. Strohmann, A. P. Antonchick, H. Waldmann, "Natural product fragment combination to performance-diverse pseudo-natural products", *Nature Communications* **2021**, *12*, 1883.

[19] B. Schölermann, J. Bonowski, M. Grigalunas, A. Burhop, Y. Xie, J. G. F. Hoock, J. Liu, M. Dow, A. Nelson, C. Nowak, A. Pahl, S. Sievers, S. Ziegler, "Identification of Dihydroorotate Dehydrogenase Inhibitors Using the Cell Painting Assay." *Chembiochem : a European journal of chemical biology* **2022**, *23*, e202200475.

[20] M. Akbarzadeh, I. Deipenwisch, B. Schoelermann, A. Pahl, S. Sievers, S. Ziegler, H. Waldmann, "Morphological profiling by means of the Cell Painting assay enables identification of tubulin-targeting compounds", *Cell Chemical Biology* **2022**, *29*, 1053–1064.e3.

[21] A. Pahl, B. Schölermann, P. Lampe, M. Rusch, M. Dow, C. Hedberg, A. Nelson, S. Sievers, H. Waldmann, S. Ziegler, "Morphological subprofile analysis for bioactivity annotation of small molecules", *Cell Chemical Biology* **2023**, 1–35.

[22] S. R. Adariani, D. Agne, S. Koska, A. Burhop, J. Warmers, P. Janning, M. Metz, A. Pahl, S. Sievers, H. Waldmann, S. Ziegler, "Detection of a Mitochondrial Stress Phenotype using the Cell Painting Assay", DOI 10.1101/2023.11.08.565491.

[23] J. Liu, S. Mallick, Y. Xie, C. Grassin, B. Lucas, B. Schölermann, A. Pahl, R. Scheel, C. Strohmann, C. Protzel, T. Berg, C. Merten, S. Ziegler, H. Waldmann, "Morphological Pro-

filing Identifies the Motor Protein Eg5 as Cellular Target of Spirooxindoles", *Angewandte Chemie - International Edition* **2023**, *62*, DOI 10.1002/anie.202301955.

[24] S. Zinken, A. Pahl, M. Grigalunas, H. Waldmann, "Phenotypic profiling enables the targeted design of a novel pseudo-natural product class", *Tetrahedron* **2023**, *143*, 133553.

[25] A. Pahl, J. Liu, S. Patil, S. R. Adariani, B. Schölermann, J. Warmers, J. Bonowski, S. Koska, Y. Akbulut, C. Seitz, S. Sievers, S. Ziegler, H. Waldmann, "Illuminating Dark Chemical Matter Using the Cell Painting Assay", *Journal of Medicinal Chemistry* **2024**, DOI 10.1021/acs.jmedchem.4c00160.

[26] J. Xie, A. Pahl, A. Krzyzanowski, A. Krupp, J. Liu, S. Koska, B. Schölermann, R. Zhang, J. Bonowski, S. Sievers, C. Strohmann, S. Ziegler, M. Grigalunas, H. Waldmann, "Synthetic Matching of Complex Monoterpene Indole Alkaloid Chemical Space", *Angewandte Chemie - International Edition* **2023**, *62*, DOI 10.1002/anie.202310222.

[27] L. Laraia, H. Waldmann, "Natural product inspired compound collections: evolutionary principle, chemical synthesis, phenotypic screening, and target identification", *Drug Discovery Today: Technologies* **2017**, *23*, 75–82.

[28] G. Karageorgis, D. J. Foley, L. Laraia, H. Waldmann, "Principle and design of pseudo-natural products", *Nature Chemistry* **2020**, *12*, 227–235.

[29] J.-M. Gally, A. Pahl, H. Waldmann, "Identifying bioactivity of pseudo-natural products using the Cell Painting assay", *Arkivoc* **2020**, *2021*, (Ed.: B. Stanovnik), 89–104.

[30] M. H. Woehrmann, W. M. Bray, J. K. Durbin, S. C. Nisam, A. K. Michael, E. Glassey, J. M. Stuart, R. S. Lokey, "Large-scale cytological profiling for functional analysis of bioactive compounds", *Molecular BioSystems* **2013**, *9*, 2604–2617.

[31] T. Engel, Basic overview of chemoinformatics, **2006**.

[32] P. Polishchuk, "Interpretation of Quantitative Structure-Activity Relationship Models: Past, Present, and Future", *Journal of Chemical Information and Modeling* **2017**, *57*, 2618–2639.

[33] A. Volkamer, S. Riniker, E. Nittinger, J. Lanini, F. Grisoni, E. Evertsson, R. Rodríguez-Pérez, N. Schneider, "Machine learning for small molecule drug discovery in academia and industry", *Artificial Intelligence in the Life Sciences* **2023**, *3*, 100056.

[34] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, http://www.deeplearningbook.org, MIT Press, **2016**.

[35] D. Weininger, "SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules", *Journal of Chemical Information and Computer Sciences* **1988**, *28*, 31–36.

[36] M. Krenn, Q. Ai, S. Barthel, N. Carson, A. Frei, N. C. Frey, P. Friederich, T. Gaudin, A. A. Gayle, K. M. Jablonka, R. F. Lameiro, D. Lemm, A. Lo, S. M. Moosavi, J. M. Nápoles-Duarte, A. K. Nigam, R. Pollice, K. Rajan, U. Schatzschneider, P. Schwaller, M. Skreta, B. Smit, F. Strieth-Kalthoff, C. Sun, G. Tom, G. Falk von Rudorff, A. Wang, A. D. White, A. Young, R. Yu, A. Aspuru-Guzik, "SELFIES and the future of molecular string representations", *Patterns* **2022**, *3*, DOI 10.1016/j.patter.2022.100588.

[37] M. Krenn, F. Häse, A. K. Nigam, P. Friederich, A. Aspuru-Guzik, "Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation", *Machine Learning: Science and Technology* **2020**, *1*, DOI 10.1088/2632-2153/aba947.

[38] D. Rogers, M. Hahn, "Extended-Connectivity Fingerprints", *Journal of Chemical Information and Modeling* **2010**, *50*, 742–754.

[39] R. Todeschini, V. Consonni, Molecular Descriptors for Cheminformatics, Wiley-VCH, **2009**.

[40] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, G. Varoquaux in ECML PKDD Workshop: Languages for Data Mining and Machine Learning, **2013**, pp. 108–122.

[41] A. M. Wassermann, M. Wawer, J. Bajorath, "Activity landscape representations for structure-activity relationship analysis", *Journal of Medicinal Chemistry* **2010**, *53*, 8209–8223.

[42] D. V. Tilborg, A. Alenicheva, F. Grisoni, "Exposing the Limitations of Molecular Machine Learning with Activity Cliffs", *Journal of Chemical Information and Modeling* **2022**, *62*, 5938–5951.

[43] G. M. Maggiora, On outliers and activity cliffs - Why QSAR often disappoints, **2006**.

[44] M. Baltruschat, P. Czodrowski, "Machine learning meets pKa", *F1000Research* **2020**, *9*, DOI 10.12688/f1000research.22090.1.

[45] A. Sanchez-Fernandez, E. Rumetshofer, S. Hochreiter, G. Klambauer, "CLOOME: contrastive learning unlocks bioimaging databases for queries with chemical structures", *Nature Communications* **2023**, *14*, 7339.

[46] N. Brown, M. Fiscato, M. H. Segler, A. C. Vaucher, "GuacaMol: Benchmarking Models for de Novo Molecular Design", *Journal of Chemical Information and Modeling* **2019**, *59*, 1096–1108.

[47] O. Méndez-Lucio, P. Marin-zapata, J. Wichard, D. Rouquié, D.-A. Clevert, "Cell morphology-guided de novo hit design by conditioning generative adversarial networks on phenotypic image features", **2020**, Preprint at https://doi.org/10.26434/chemrxiv.1159.

[48] L. Weng, "From Autoencoder to Beta-VAE", *lilianweng.github.io* **2018**.

[49] G. E. Hinton, R. R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks", *Science* **2006**, *313*, 504–507.

[50] D. P. Kingma, M. Welling, "Auto-Encoding Variational Bayes", *CoRR* **2013**, *abs/1312.6114*, DOI 10.48550/arXiv.1312.6114.

[51] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, "Generative Adversarial Networks", *Machine Learning* **2014**, *112*, 5135–5161.

[52] L. Weng, "From GAN to WGAN", *lilianweng.github.io* **2017**.

[53] M. Mirza, S. Osindero, "Conditional Generative Adversarial Nets", **2014**, 1–7.

[54] R. Winter, F. Montanari, F. Noé, D. A. Clevert, "Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations", *Chemical Science* **2019**, *10*, 1692–1701.

[55] Filter Catalog, Accessed: [3 May 2024].

[56] J. Brownlee, "How to Develop a Conditional GAN (cGAN) From Scratch", *MachineLearningMastery* **2020**.

[57] P. A. M. Zapata, O. Méndez-Lucio, T. Le, C. J. Beese, J. Wichard, D. Rouquié, D.-A. Clevert, "Cell morphology-guided de novo hit design by conditioning GANs on phenotypic image features", *Digital Discovery* **2023**, *2*, 91–102.

[58] C. de Duve, T. de Barsy, B. Poole, A. Trouet, P. Tulkens, F. van Hoof, "Lysosomotropic Agents", *Biochemical Pharmacology* **1974**, *23*, 2495–2531.

[59] S. Pisonero-Vaquero, D. L. Medina, "Lysosomotropic Drugs: Pharmacological Tools to Study Lysosomal Function", *Current Drug Metabolism* **2017**, *18*, 1147–1158.

[60] O. F. Kuzu, M. Toprak, M. A. Noory, G. P. Robertson, "Effect of lysosomotropic molecules on cellular homeostasis", *Pharmacological Research* **2017**, *117*, 177–184.

[61] M. Blaess, L. Kaiser, M. Sauer, R. Csuk, H.-P. Deigner, "COVID-19/SARS-CoV-2 Infection: Lysosomes and Lysosomotropism Implicate New Treatment Strategies and Personal Risks", *International Journal of Molecular Sciences* **2020**, *21*, 4953.

[62] M. J. Vincent, E. Bergeron, S. Benjannet, B. R. Erickson, P. E. Rollin, T. G. Ksiazek, N. G. Seidah, S. T. Nichol, "Chloroquine is a potent inhibitor of SARS coronavirus infection and spread", *Virology Journal* **2005**, *2*, 1–10.

[63] U. Norinder, A. Tuck, K. Norgren, V. M. Kos, "Existing highly accumulating lysosomotropic drugs with potential for repurposing to target COVID-19", *Biomedicine and Pharmacotherapy* **2020**, *130*, 110582.

[64] E. Keyaerts, L. Vijgen, P. Maes, J. Neyts, M. V. Ranst, "In vitro inhibition of severe acute respiratory syndrome coronavirus by chloroquine", *Biochemical and Biophysical Research Communications* **2004**, *323*, 264–268.

[65] C. A. Devaux, J. M. Rolain, P. Colson, D. Raoult, "New insights on the antiviral effects of chloroquine against coronavirus: what to expect for COVID-19?", *International Journal of Antimicrobial Agents* **2020**, *55*, 105938.

[66] "Drug-induced phospholipidosis confounds drug repurposing for SARS-CoV-2", *Science* **2021**, *373*, 541–547.

[67] A. M. Henao-Restrepo, H. Pan, R. Peto, M. P. Preziosi, V. Sathiyamoorthy, *et al.*, "Remdesivir and three other drugs for hospitalised patients with COVID-19: final results of the WHO Solidarity randomised trial and updated meta-analyses", *The Lancet* **2022**, *399*, 1941–1953.

[68] Chloroquine or Hydroxychloroquine and/or Azithromycin, National Institutes of Health (NIH) COVID-19 Treatment Guidelines. Accessed: 2023-02-25, **2021**.

[69] Table. Chloroquine or Hydroxychloroquine and/or Azithromycin: Selected Clinical Data, National Institutes of Health (NIH) COVID-19 Treatment Guidelines. Accessed: 2023-02-25, **2021**.

[70] F. Marceau, M. T. Bawolak, R. Lodge, J. Bouthillier, A. Gagné-Henley, R. C.-Gaudreault, G. Morissette, "Cation trapping by cellular acidic compartments: Beyond the concept of lysosomotropic drugs", *Toxicology and Applied Pharmacology* **2012**, *259*, 1–12.

[71] S. Nadanaciva, S. Lu, D. F. Gebhard, B. A. Jessen, W. D. Pennie, Y. Will, "A high content screening assay for identifying lysosomotropic compounds", *Toxicology in Vitro* **2011**, *25*, 715–723.

[72] B. Zhitomirsky, Y. G. Assaraf, "Lysosomal sequestration of hydrophobic weak base chemotherapeutics triggers lysosomal biogenesis and lysosome-dependent cancer multidrug resistance", *Oncotarget* **2015**, *6*, 1143–1156.

[73] S. Lu, T. Sung, N. Lin, R. T. Abraham, B. A. Jessen, "Lysosomal adaptation: How cells respond to lysosomotropic compounds", *PLoS ONE* **2017**, *12*, 1–22.

[74] A. Ufuk, F. Assmus, L. Francis, J. Plumb, V. Damian, M. Gertz, J. B. Houston, A. Galetin, "In Vitro and in Silico Tools To Assess Extent of Cellular Uptake and Lysosomal Seques-

tration of Respiratory Drugs in Human Alveolar Macrophages", *Molecular Pharmaceutics* **2017**, *14*, 1033–1046.

[75] M. V. Schmitt, P. Lienau, G. Fricker, A. Reichel, "Quantitation of Lysosomal Trapping of Basic Lipophilic Compounds Using In Vitro Assays and In Silico Predictions Based on the Determination of the Full pH Profile of the Endo-/Lysosomal System in Rat Hepatocytes." *Drug Metabolism and Disposition* **2019**, *47*, 49–57.

[76] U. Norinder, V. M. Kos, "QSAR models for predicting five levels of cellular accumulation of lysosomotropic macrocycles", *International Journal of Molecular Sciences* **2019**, *20*, 1–13.

[77] H. Hu, A. Tjaden, S. Knapp, A. A. Antolin, S. Müller, "A machine learning and live-cell imaging tool kit uncovers small molecules induced phospholipidosis." *Cell Chemical Biology* **2023**, 1–18.

[78] S. Ohkuma, B. Poole, "Cytoplasmic vacuolation of mouse peritoneal macrophages and the uptake into lysosomes of weakly basic substances", *Journal of Cell Biology* **1981**, *90*, 656–664.

[79] H. Chen, O. Engkvist, Y. Wang, M. Olivecrona, T. Blaschke, "The rise of deep learning in drug discovery", *Drug Discovery Today* **2018**, *23*, 1241–1250.

[80] P. Schneider, W. P. Walters, A. T. Plowright, N. Sieroka, J. Listgarten, R. A. Goodnow, J. Fisher, J. M. Jansen, J. S. Duca, T. S. Rush, et al., "Rethinking drug design in the Artificial Intelligence Era", *Nature Reviews Drug Discovery* **2019**, *19*, 353–364.

[81] M. T. Ribeiro, S. Singh, C. Guestrin in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016, **2016**, pp. 1135–1144.

[82] S. Lundberg, S.-I. Lee, "A Unified Approach to Interpreting Model Predictions", *Advances in Neural Information Processing Systems* **2017**, *2017-Decem*, 4766–4775.

[83] R. Rodríguez-Pérez, J. Bajorath, "Interpretation of Compound Activity Predictions from Complex Machine Learning Models Using Local Approximations and Shapley Values", *Journal of Medicinal Chemistry* **2020**, *63*, 8761–8777.

[84] J. Jiménez-Luna, M. Skalic, N. Weskamp, G. Schneider, "Coloring Molecules with Explainable Artificial Intelligence for Preclinical Relevance Assessment", *Journal of Chemical Information and Modeling* **2021**, *61*, 1083–1094.

[85] T. Harren, H. Matter, G. Hessler, M. Rarey, C. Grebner, "Interpretation of Structure–Activity Relationships in Real-World Drug Design Data Sets Using Explainable Artificial Intelligence", *Journal of Chemical Information and Modeling* **2022**, *62*, 447–462.

[86] L. Laraia, G. Garivet, D. J. Foley, N. Kaiser, S. Müller, S. Zinken, T. Pinkert, J. Wilke, D. Corkery, A. Pahl, S. Sievers, P. Janning, C. Arenz, Y. Wu, R. Rodriguez, H. Waldmann, "Image-Based Morphological Profiling Identifies a Lysosomotropic, Iron-Sequestering Autophagy Inhibitor", *Angewandte Chemie International Edition* **2020**, *59*, 5721–5729.

[87] K. Kumar, H. Waldmann, "Synthesis of Natural Product Inspired Compound Collections", *Angewandte Chemie International Edition* **2009**, *48*, 3224–3242.

[88] Imatinib, DrugBank. Accessed: 2023-03-07.

[89] Toremifene, DrugBank. Accessed: 2023-03-07.

[90] Clozapine, DrugBank. Accessed: 2023-03-07.

[91] P. W. Kenney, J. Sadowski, Structure Modification in Chemical Databases, Wiley-VCH, **2004**, p. 493.

[92] "Lead Optimization Using Matched Molecular Pairs: Inclusion of Contextual Information for Enhanced Prediction of hERG Inhibition, Solubility, and Lipophilicity", *Journal of Chemical Information and Modeling* **2010**, *50*, 1872–1886.

[93] A. Dalke, J. Hert, C. Kramer, "Mmpdb: An Open-Source Matched Molecular Pair Platform for Large Multiproperty Data Sets", *Journal of Chemical Information and Modeling* **2018**, *58*, 902–910.

[94] A. M. Wassermann, D. Dimova, P. Iyer, J. Bajorath, "Advances in Computational Medicinal Chemistry: Matched Molecular Pair Analysis", *Drug Development Research* **2012**, *73*, 518–527.

[95] A. G. Dossetter, E. J. Griffen, A. G. Leach, "Matched Molecular Pair Analysis in Drug Discovery", *Drug Discovery Today* **2013**, *18*, 724–731.

[96] C. Tyrchan, E. Evertsson, "Matched Molecular Pair Analysis in Short: Algorithms, Applications and Limitations", *Computational and Structural Biotechnology Journal* **2017**, *15*, 86–90.

[97] E. Griffen, A. G. Leach, G. R. Robb, D. J. Warner, "Matched molecular pairs as a medicinal chemistry tool", *Journal of Medicinal Chemistry* **2011**, *54*, 7739–7750.

[98] M. Awale, J. Hert, L. Guasch, S. Riniker, C. Kramer, "The Playbooks of Medicinal Chemistry Design Moves", *Journal of Chemical Information and Modeling* **2021**, *61*, 729–742.

[99] J. Hussain, C. Rea, "Computationally efficient algorithm to identify matched molecular pairs (MMPs) in large data sets", *Journal of Chemical Information and Modeling* **2010**, *50*, 339–348.

[100] H. L. Morgan, "The Generation of a Unique Machine Description for Chemical Structures-A Technique Developed at Chemical Abstracts Service." *Journal of Chemical Documentation* **1965**, *5*, 107–113.

[101] P. Gedeck, B. Rohde, C. Bartels, "QSAR - How Good Is It in Practice? Comparison of Descriptor Sets on an Unbiased Cross Section of Corporate Data Sets", *Journal of Chemical Information and Modeling* **2006**, *46*, 1924–1936.

[102] rdkit/rdkit: 2022_09_5 (Q3 2022) Release, **2023**.

[103] M. Bramer in Principles of Data Mining, Springer London, **2013**, pp. 121–136.

[104] T. Chen, C. Guestrin in Proc. 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, **2016**, pp. 785–794.

[105] T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama in Proc. 25th ACM SIGKDD Conf. **2019**.

[106] L. S. Shapley in Contributions to the Theory of Games (AM-28), Volume II, Princeton University Press, **1953**, pp. 307–318.

[107] C. Humer, H. Heberle, F. Montanari, T. Wolf, F. Huber, R. Henderson, J. Heinrich, M. Streit, "ChemInformatics Model Explorer (CIME): exploratory analysis of chemical model explanations", *Journal of Cheminformatics* **2022**, *14*, 1–14.

[108] S. M. Lundberg, G. Erion, H. Chen, A. DeGrave, J. M. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, S. I. Lee, "From local explanations to global understanding with explainable AI for trees", *Nature Machine Intelligence* **2020**, *2*, 56–67.

[109] A. Tandon, M. Baltruschat, X-FP: eXplainable FingerPrints, X-FP GitHub repository. Accessed: 2023 2023-11-30.

[110] Lysosomotropism Predictor WebApp, CzodrowskiLab Homepage. Accessed: 2023-10-24, **2023**.

[111] S. Zimmermann, M. Akbarzadeh, F. Otte, C. Strohmann, M. G. Sankar, S. Ziegler, A. Pahl, S. Sievers, K. Kumar, "A Scaffold-Diversity Synthesis of Biologically Intriguing Cyclic Sulfonamides", *Chemistry – A European Journal* **2019**, *25*, 15498–15503.

[112] Lysosomotropic Project GitHub Repo, CzodrowskiLab Lyso_Project_Open GitHub repository. Accessed: 2023-11-30, **2023**.

[113] M. Awale, S. Riniker, C. Kramer, "Matched Molecular Series Analysis for ADME Property Prediction", *Journal of Chemical Information and Modeling* **2020**, *60*, 2903–2914.

[114] A. Pahl, mpimp-comas/2022_grigalunas_smo_anta: Release for publication, version 1.0.0, **2022**.

[115] N. Schneider, A. Schuffenhauer, NIBR Substructure Filters Python Script, RDKit Contrib NIBRSubstructureFilters GitHub repository. Accessed: 2023-11-30, **2020**.

[116] A. Schuffenhauer, N. Schneider, S. Hintermann, D. Auld, J. Blank, S. Cotesta, C. Engeloch, N. Fechner, C. Gaul, J. Giovannoni, J. Jansen, J. Joslin, P. Krastel, E. Lounkine, J. Manchester, L. G. Monovich, A. P. Pelliccioli, M. Schwarze, M. D. Shultz, N. Stiefl, D. K. Baeschlin, "Evolution of Novartis' Small Molecule Screening Deck Design", *Journal of Medicinal Chemistry* **2020**, *63*, 14425–14447.

[117] H. Developers, HoloViz, Accessed: [27 April 2024], **2021**.

[118] J. Jiménez-Luna, F. Grisoni, G. Schneider, "Drug discovery with explainable artificial intelligence", *Nature Machine Intelligence* **2020**, *2*, 573–584.

[119] I. Ponzoni, J. A. P. Prosper, N. E. Campillo, "Explainable artificial intelligence: A taxonomy and guidelines for its application to drug discovery", *Wiley Interdisciplinary Reviews: Computational Molecular Science* **2023**, *13*, DOI 10.1002/wcms.1681.

[120] R. Guha, "On the interpretation and interpretability of quantitative structure-activity relationship models", *Journal of Computer-Aided Molecular Design* **2008**, *22*, 857–871.

[121] M. McGibbon, S. Shave, J. Dong, Y. Gao, D. R. Houston, J. Xie, Y. Yang, P. Schwaller, V. Blay, "From intuition to AI: evolution of small molecule representations in drug discovery", *Briefings in Bioinformatics* **2023**, *25*, 1–13.

[122] C. Jamieson, E. M. Moir, Z. Rankovic, G. Wishart, "Medicinal Chemistry of hERG Optimizations: Highlights and Hang-Ups", *Journal of Medicinal Chemistry* **2006**, *49*, 5029–5046.

[123] C. Zhang, Y. Zhou, S. Gu, Z. Wu, W. Wu, C. Liu, K. Wang, G. Liu, W. Li, P. W. Lee, Y. Tang, "In silico prediction of hERG potassium channel blockage by chemical category approaches", *Toxicology Research* **2016**, *5*, 570–582.

[124] R. N. DuBois, S. B. Abramson, L. Crofford, R. A. Gupta, L. S. Simon, L. B. A. Putte, P. E. Lipsky, "Cyclooxygenase in biology and disease", *The FASEB Journal* **1998**, *12*, 1063–1073.

[125] "Inhibition of the CDK2 and Cyclin A complex leads to autophagic degradation of CDK2 in cancer cells", *Nature Communications* **2022**, *13*, DOI 10.1038/s41467-022-30264-0.

[126] A. Schwienhorst, Direct thrombin inhibitors - A survey of recent developments, **2006**.

[127] D. Sydow, A. Morger, M. Driller, A. Volkamer, "TeachOpenCADD: a teaching platform for computer-aided drug design using open source packages and data", *Journal of Cheminformatics* **2019**, *11*, 29.

[128] Langa, Łukasz, contributors to Black, Black: The uncompromising Python code formatter, version 1.2.0, **2023**.

7. Supplementary Information

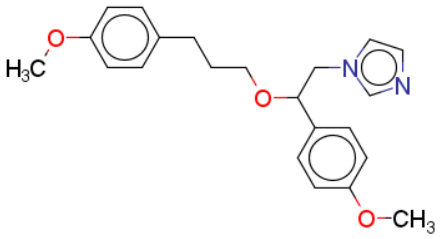


Figure S1.: Example reference compound from Tubulin Biocluster.
SMILES: COc1ccc(CCCOC(Cn2ccnc2)c2ccc(OC)cc2)cc1

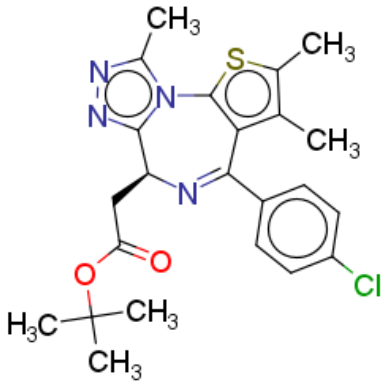


Figure S2.: Example reference compound from BET Biocluster.
SMILES: Cc1sc2c(c1C)C(c1ccc(Cl)cc1)=N[C@@H](CC(=O)OC(C)(C)C)c1nnc(C)n1-2

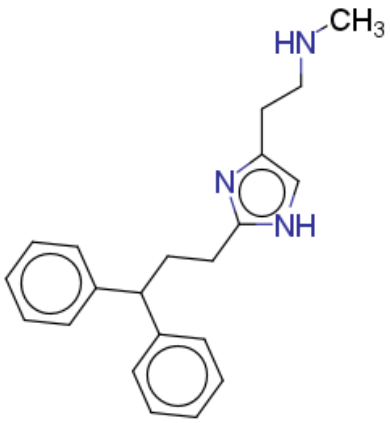


Figure S3.: Example reference compound from LCH Biocluster.
SMILES: CNCCc1c[nH]c(CCC(c2ccccc2)c2ccccc2)n1

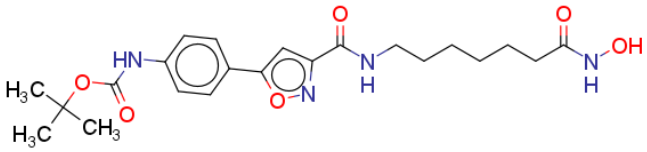


Figure S4.: Example reference compound from HDAC Biocluster.
SMILES: CC(C)(C)OC(=O)Nc1ccc(-c2cc(C(=O)NCCCCCCC(=O)NO)no2)cc1

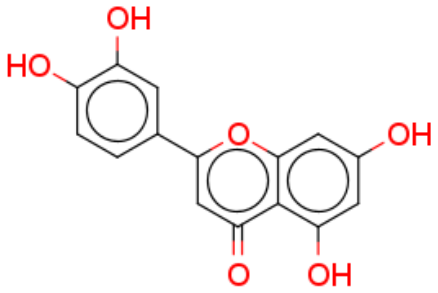


Figure S5.: Example reference compound from Protein Synth Biocluster.
SMILES: O=c1cc(-c2ccc(O)c(O)c2)oc2cc(O)cc(O)c12

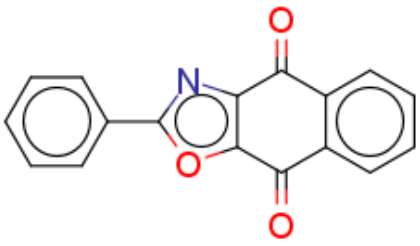


Figure S6.: Example reference compound from Mito Stress Biocluster.
SMILES: O=C1c2ccccc2C(=O)c2oc(-c3ccccc3)nc21

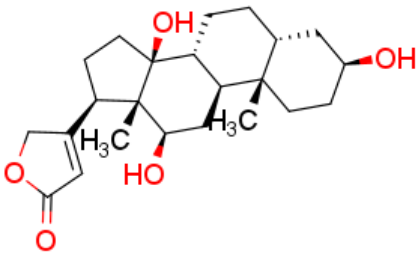


Figure S7.: Example reference compound from Na_K-ATPase Biocluster.
SMILES: C[C@]12CC[C@H](O)C[C@H]1CC[C@@H]1[C@@H]
2C[C@@H](O)[C@]2(C)[C@@H](C3=CC(=O)OC3)CC[C@]12O

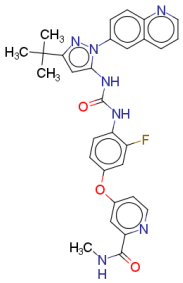


Figure S8.: Example reference compound from MTOR Biocluster.
SMILES: CNC(=O)c1cc(Oc2ccc(NC(=O)Nc3cc(C(C)(C)C)nn3-c3ccc4ncccc4c3)c(F)c2)ccn1

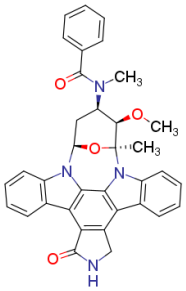


Figure S9.: Example reference compound from HSP90 Biocluster.
SMILES: CO[C@@H]1[C@H](N(C)C(=O)c2ccccc2)C[C@H]2O[C@]1(C)n1c3ccccc3c3c4c(c5c6ccccc6n2c5c31)C(=O)NC4

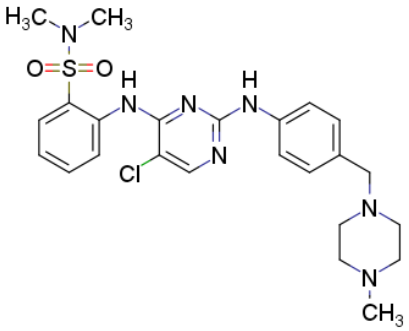


Figure S10.: Example reference compound from DNA Synthase Biocluster.
SMILES: CN1CCN(Cc2ccc(Nc3ncc(Cl)c(Nc4ccccc4S(=O)(=O)N(C)C)n3)cc2)CC1

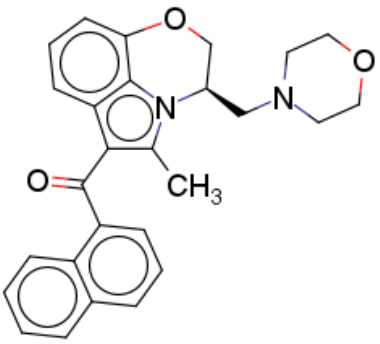


Figure S11.: Example reference compound from Uncoupler Biocluster.
SMILES: Cc1c(C(=O)c2cccc3ccccc23)c2cccc3c2n1[C@H](CN1CCOCC1)CO3

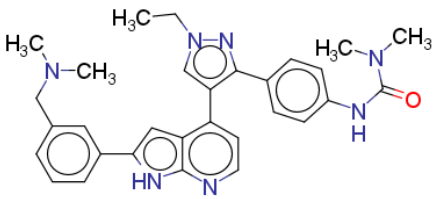


Figure S12.: Example reference compound from Aurora Kinase Biocluster.
SMILES: CCn1cc(-c2ccnc3[nH]c(-c4cccc(CN(C)C)c4)cc23)c(-c2ccc(NC(=O)N(C)C)cc2)n1

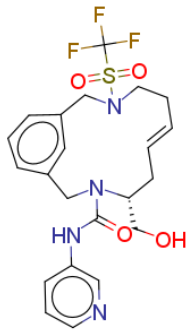


Figure S13.: Example reference compound from Pyrimidine Synthase Biocluster.
SMILES: O=C(Nc1cccnc1)N1Cc2cccc(c2)CN(S(=O)(=O)C(F)(F)F)CC/C=C/C[C@@H]1CO

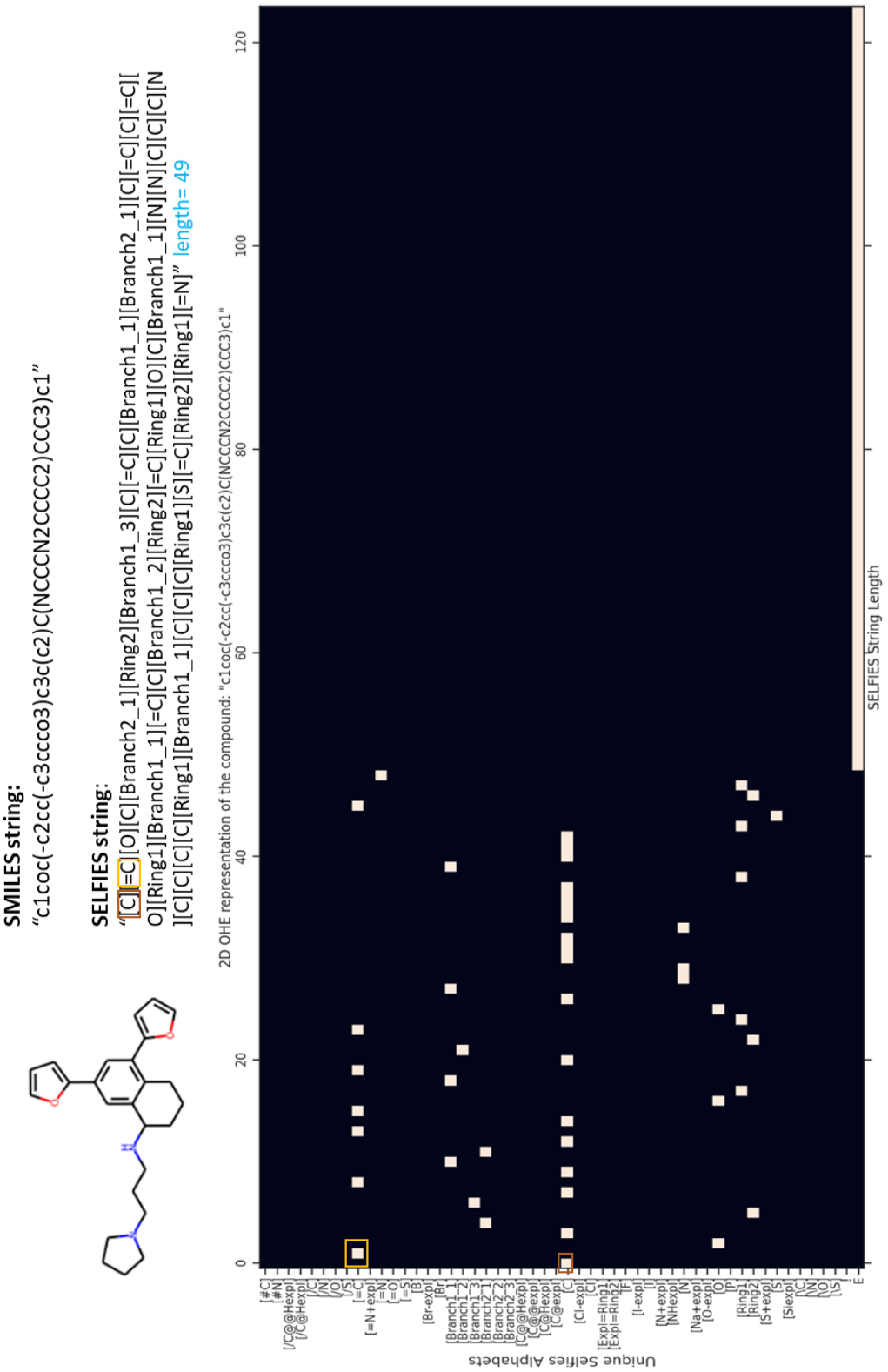


Figure S14.: Visualization of two-dimensional one-hot encoded SELFIES. Each unit on x-axis can only have one bit switched on - denoting the unique SELFIES alphabet encoded at that position in the string. Alphabet 'E' is used as for padding. The first 2 elements of the exemplary molecule are highlighted in the plot.

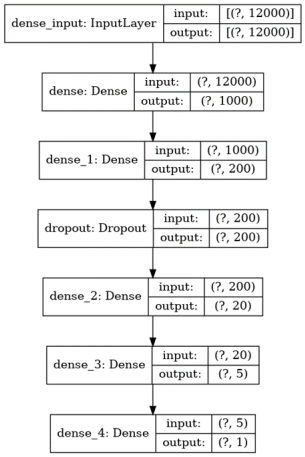


Figure S15.: Neural Network (NN) plot for the regression model with SELFIES as input. The question mark denotes the sample size. Models with similar architecture were used for Morgan Fingerprints (MF) as input.

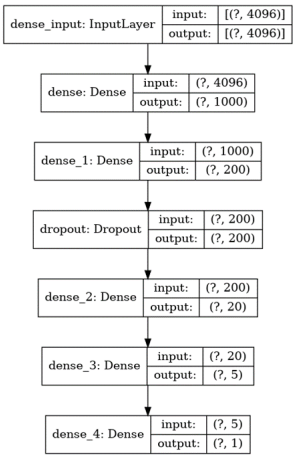


Figure S16.: Neural Network (NN) plot for the regression model with MF as input. The question mark denotes the sample size.

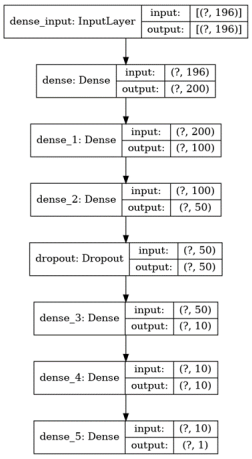


Figure S17.: Neural Network (NN) plot for the regression model with RDKit descriptors as input. The question mark denotes the sample size.

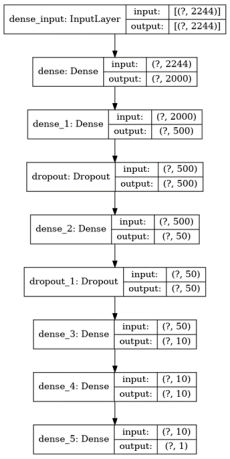


Figure S18.: Neural Network (NN) plot for the regression model with MF + RDKit descriptors as input. The question mark denotes the sample size.

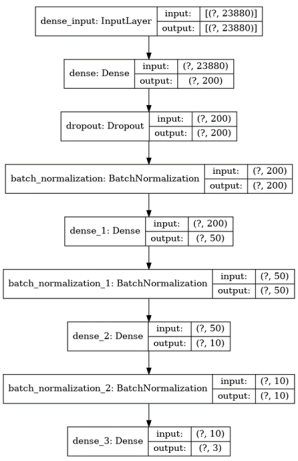


Figure S19.: Neural Network (NN) plot for the classification model with 1D-OHE SELFIES as input. The question mark denotes the sample size.

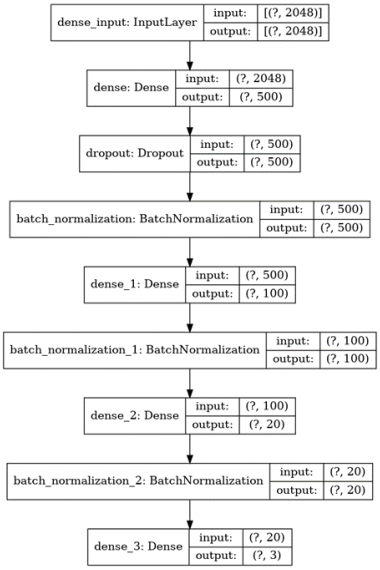


Figure S20.: Neural Network (NN) plot for the classification model with MF as input. The question mark denotes the sample size.

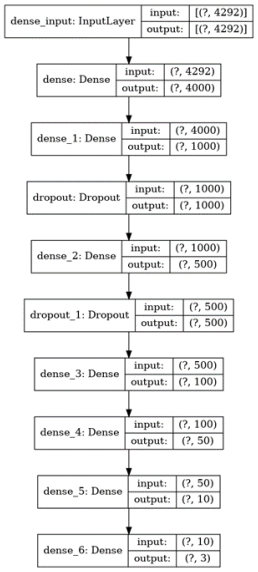


Figure S21.: Neural Network (NN) plot for the classification model with MF + RDKit descriptors as input. The question mark denotes the sample size.

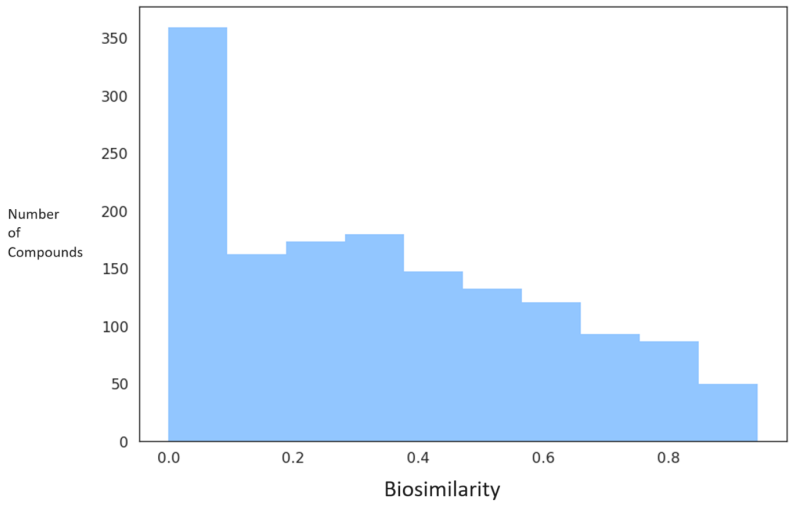


Figure S22.: Ensemble model performance evaluation by density plot. On x-axis, similarity between predicted CP profile and original CP profile is present. Peaks near low similarity value indicate poor profile similarities which in turn indicate poor model performance.

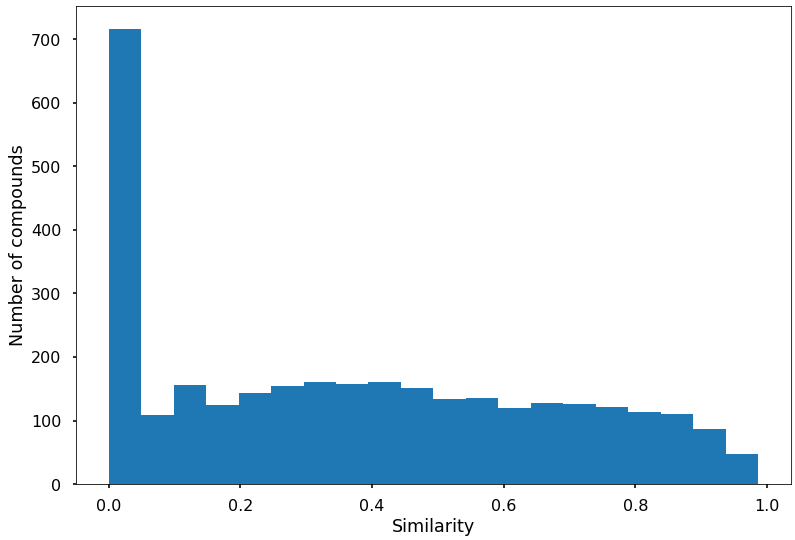


Figure S23.: Model performance of Optuna-tuned multi regression model. Evaluation by density plot. On x-axis, similarity between predicted CP profile and original CP profile is present. Peaks near low similarity value indicate poor profile similarities which in turn indicate poor model performance.

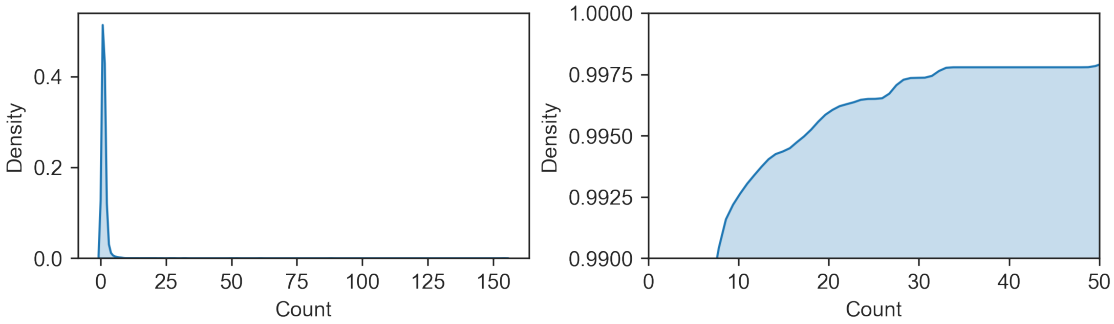


Figure S24.: MMPA: (Cumulative) Density distribution of counts (number of occurrence) of transformations.

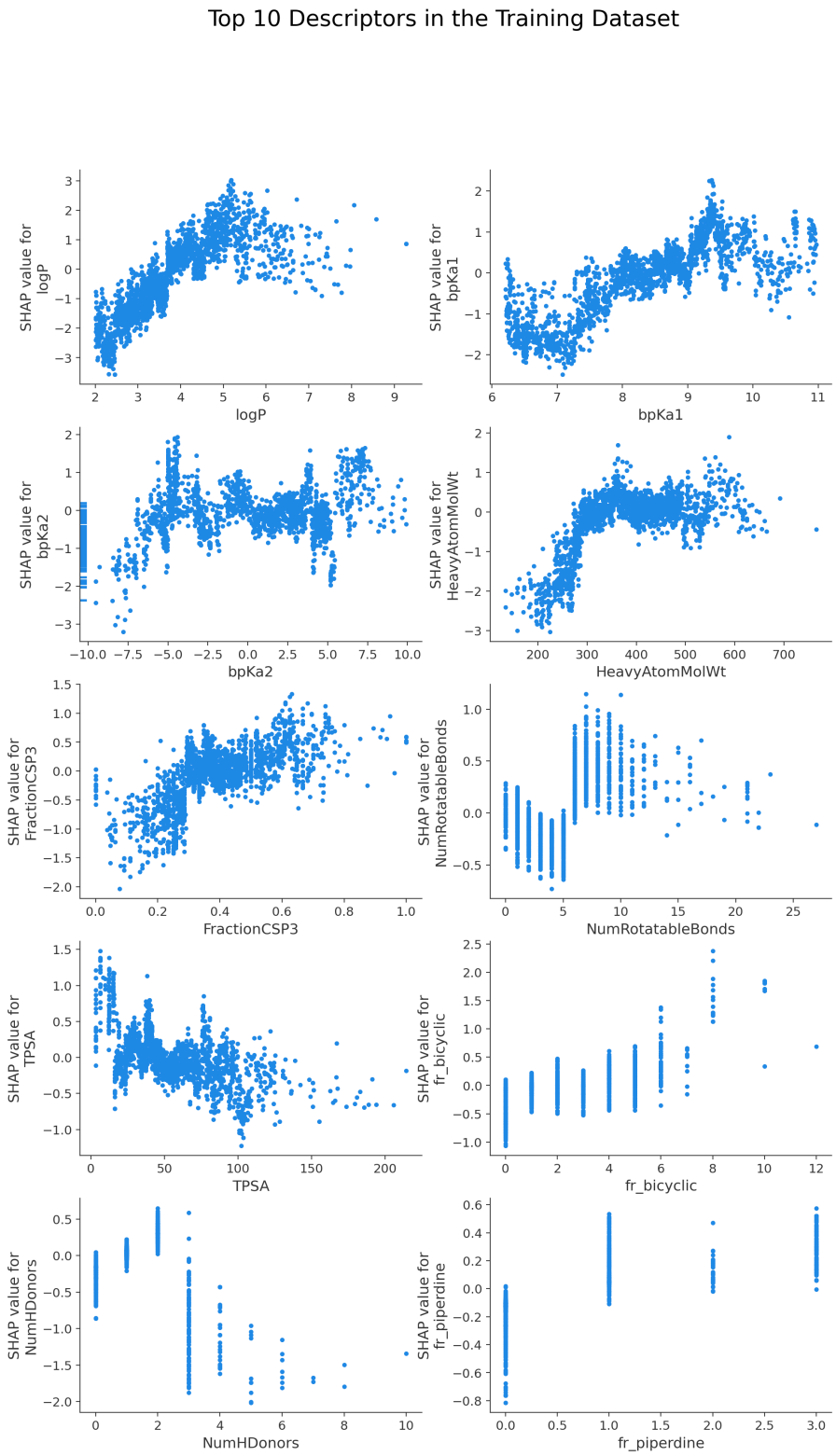


Figure S25.: Descriptor model: Top 10 descriptors' Dependence Plots in the training dataset.

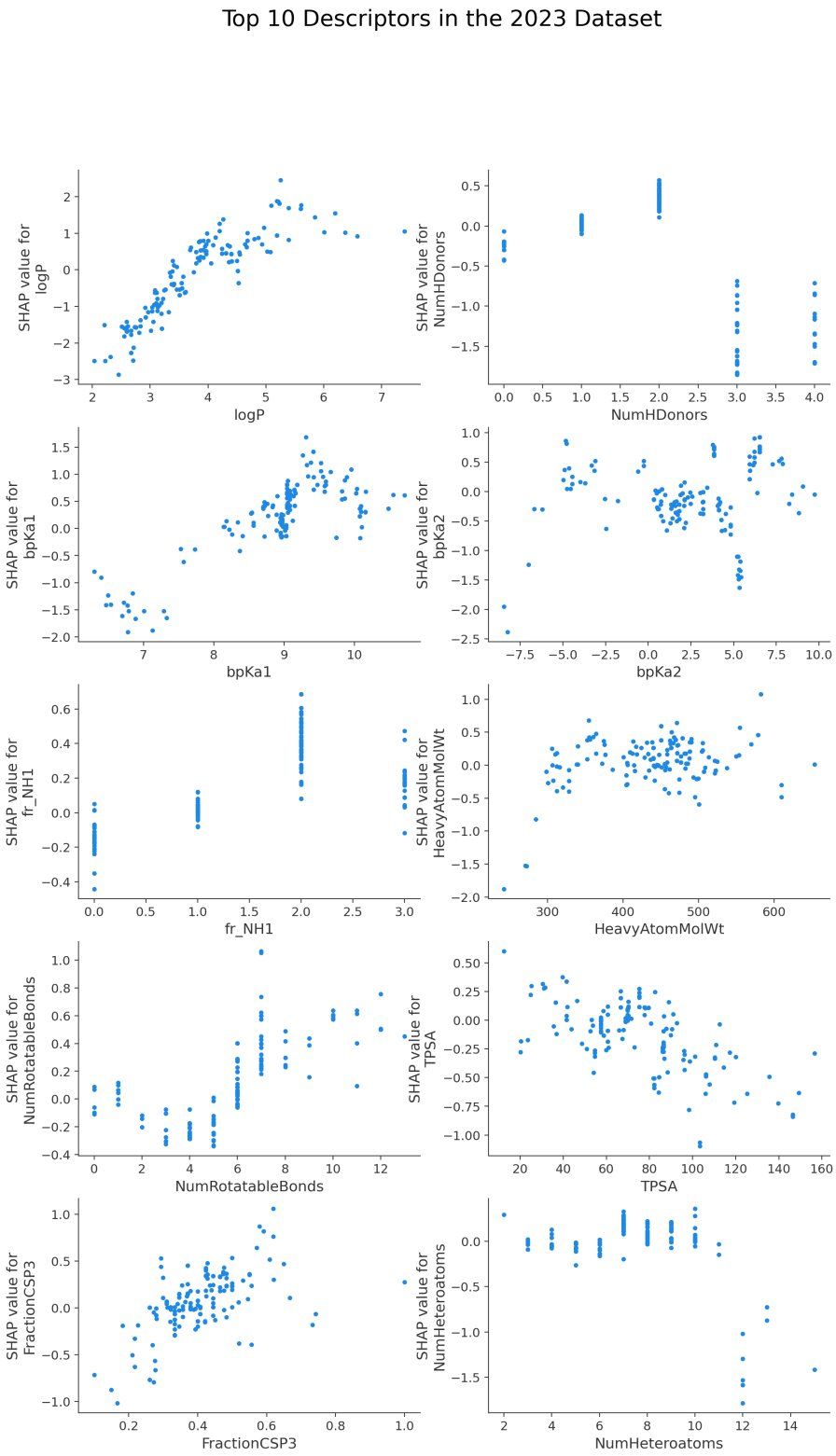


Figure S26.: Descriptor model: Top 10 descriptors' Dependence Plots in the time-split dataset.

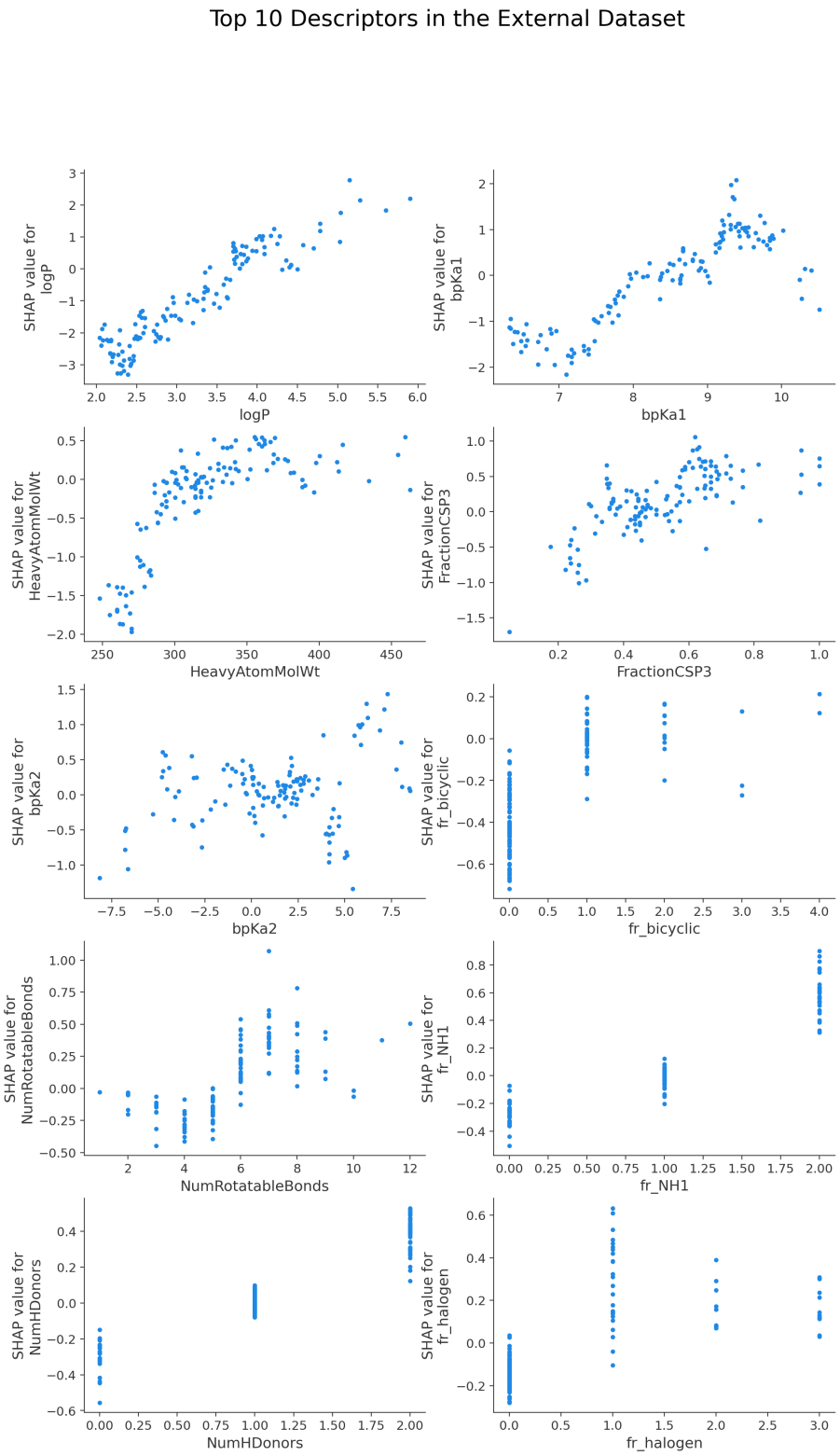
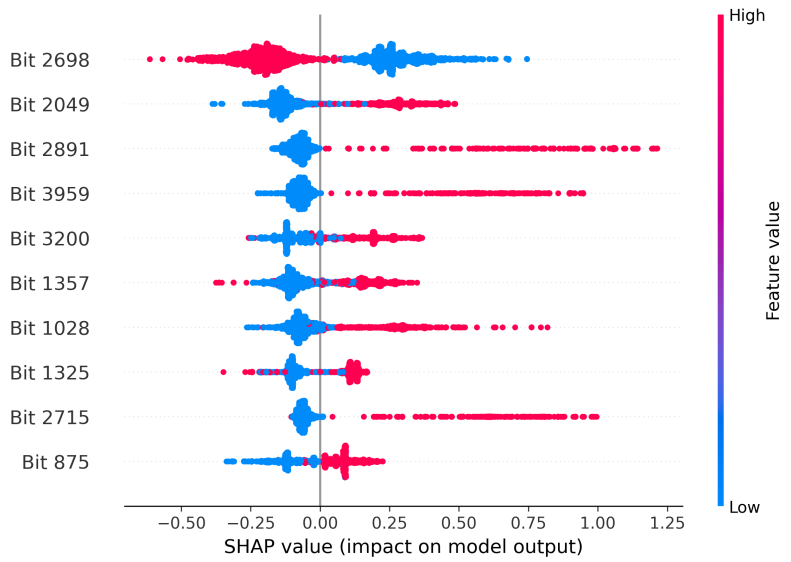


Figure S27.: Descriptor model: Top 10 descriptors' Dependence Plots in the external dataset.

Figure S28.: Fingerprint model: X-FP report for the Training Set

Training Data: X-FP Report



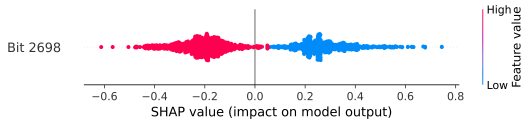
SHAP Summary Plot for the Top 10 Morgan Fingerprint Bits

X-FP

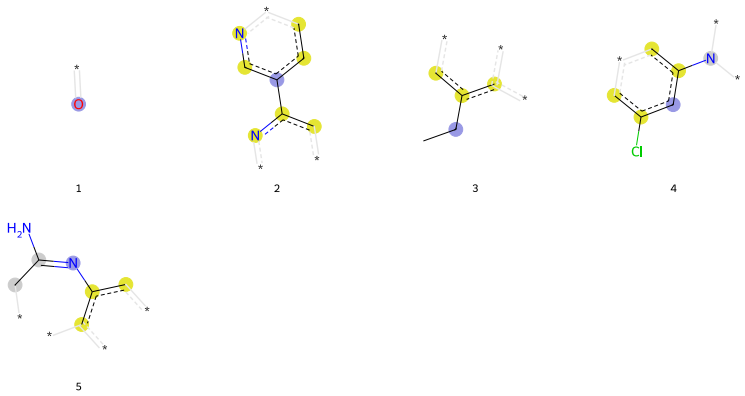
Bit Analysis

Bit 2698

SHAP Summary Plot for the Bit 2698:



Substructures present:



Substructures frequency table (total compounds: 2065):

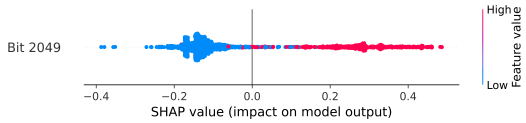
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[O]~*	6335	1743
2	[c;R]1(:[cH;R]:[cH;R]~*~[n;R]:[cH;R]:1)-[c;R](:[cH;R]~*):[n;R]~*	6	6
3	[CH2](-[CH3])-[c;R](:[cH;R]~*):[c;R](~*)~*	19	16
4	[cH;R]1:[c;R](:[cH;R]~*~[cH;R]:[c;R]:1-[N;R](~*)~*)-[Cl]	7	7
5	[N;R](=[C;R](-[NH2])-[CH2;R]~*)-[c;R](:[cH;R]~*):[c;R](~*)~*	1	1

X-FP

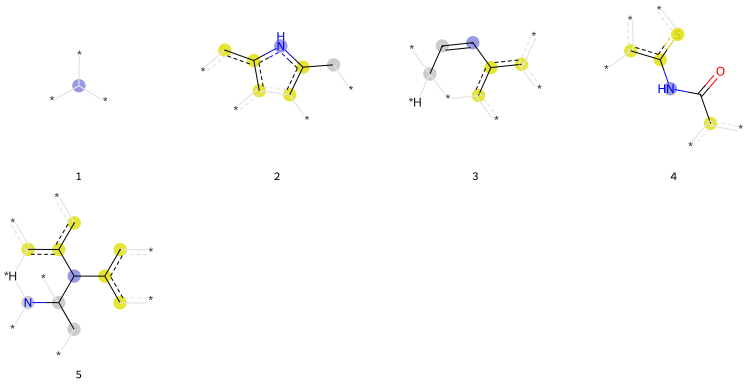
Bit Analysis

Bit 2049

SHAP Summary Plot for the Bit 2049:



Substructures present:



Substructures frequency table (total compounds: 2065):

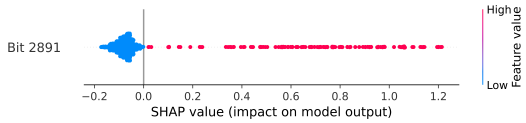
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH](~*)(~*)~*	3287	1463
2	[nH;R]1:[c;R](:[c;R](:[c;R]:1:[cH;R]~*)~*)-[CH2;R]~*	41	41
3	[CH;R]1-[c;R](:[c;R](~*)~*~[C;R](-[CH;R]=1)(~*)~*):[c;R](~*)~*	2	2
4	[NH](-[C](=[O])-[c;R](~*)~*)-[c;R](:[s;R]~*):[c;R](~*)~*	1	1
5	[CH;R]1(-[CH;R](-[N;R](~*)~*~[c;R](:[c;R]-1:[cH;R]~*)~*)-[CH;R](~*)~*)-[c;R](:[cH;R]~*):[cH;R]~*	3	3

X-FP

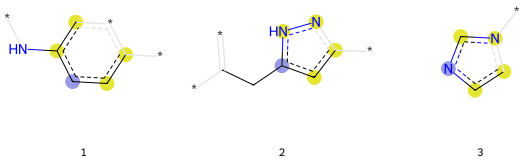
Bit Analysis

Bit 2891

SHAP Summary Plot for the Bit 2891:



Substructures present:



1 2 3

Substructures frequency table (total compounds: 2065):

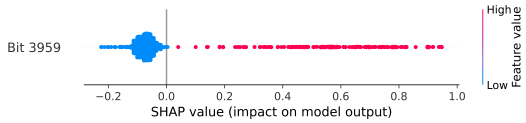
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[cH;R]1:[cH;R]:[c;R](~*)~*~[cH;R]:[c;R]:1-[NH]~*	221	170
2	[c;R]1(:[cH;R]:[c;R](:[n;R]:[nH;R]:1)~*)-[CH2]-[C](~*)~*	1	1
3	[n;R]1:[cH;R]:[cH;R]:[n;R](:[cH;R]:1)~*	12	11

X-FP

Bit Analysis

Bit 3959

SHAP Summary Plot for the Bit 3959:



Substructures present:

1 2

Substructures frequency table (total compounds: 2065):

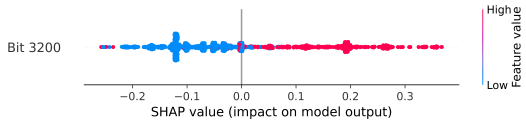
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH2](-[CH2]~*)-[CH2]~*	1251	650
2	[CH2](-[CH2]-[CH2]~*)-[CH](-[OH])-[c;R](~*)~*	2	2

X-FP

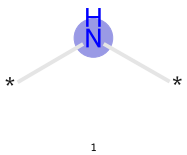
Bit Analysis

Bit 3200

SHAP Summary Plot for the Bit 3200:



Substructures present:



Substructures frequency table (total compounds: 2065):

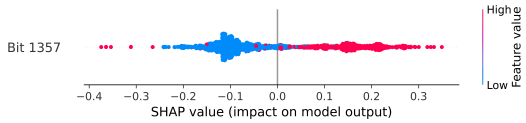
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[NH](~*)~*	1334	1033

X-FP

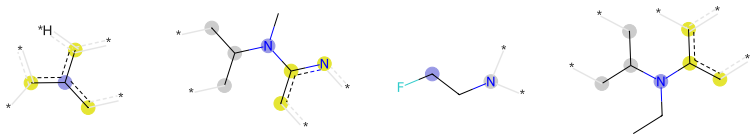
Bit Analysis

Bit 1357

SHAP Summary Plot for the Bit 1357:



Substructures present:



Substructures frequency table (total compounds: 2065):

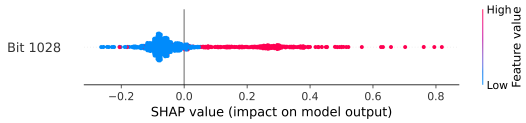
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[c;R](:[cH;R]~*)(:[c;R](~*)~*):[c;R](~*)~*	729	687
2	[N](-[CH3])(-[CH;R](-[CH2;R]~*)-[CH2;R]~*)-[c;R](:[cH;R]~*):[n;R]~*	5	5
3	[CH2](-[F])-[CH2]-[N;R](~*)~*	1	1
4	[N](-[CH2]-[CH3])(-[CH;R](-[CH2;R]~*)-[CH2;R]~*)-[c;R](:[cH;R]~*):[c;R](~*)~*	2	2

X-FP

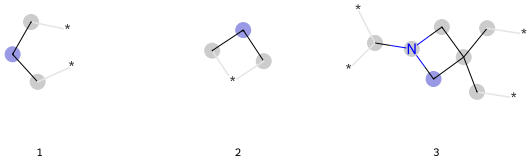
Bit Analysis

Bit 1028

SHAP Summary Plot for the Bit 1028:



Substructures present:



Substructures frequency table (total compounds: 2065):

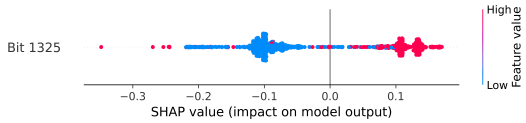
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH2;R](-[CH2;R]~*)-[CH2;R]~*	810	474
2	[CH2;R]1-[CH2;R]~*~[CH2;R]-1	13	13
3	[CH2;R]1-[N;R](-[CH2;R]-[C;R]-1(-[CH2;R]~*)-[CH2;R]~*)-[CH;R](~*)~*	1	1

X-FP

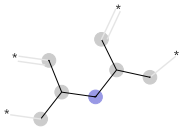
Bit Analysis

Bit 1325

SHAP Summary Plot for the Bit 1325:



Substructures present:



Substructures frequency table (total compounds: 2065):

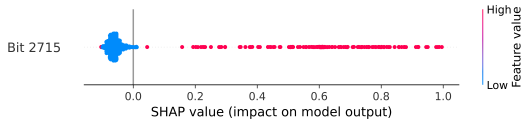
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH2;R](-[CH2;R]~*)-[CH;R](~*)~*	1093	779
2	[CH2;R]1-[CH2;R]-[CH;R]-1~*	23	22
3	[CH2;R]1-[CH2;R]~*~[CH;R]-1~*	6	6
4	[CH2](-[CH2]-[CH2]~*)-[O]-[CH2]~*	6	3
5	[CH2](-[CH;R](-[CH2;R]~*)-[CH;R]~*)-[CH;R](-[CH2;R]~*)-[CH;R]~*	1	1

X-FP

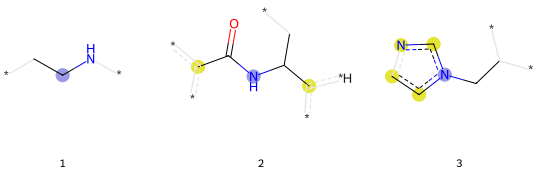
Bit Analysis

Bit 2715

SHAP Summary Plot for the Bit 2715:



Substructures present:



Substructures frequency table (total compounds: 2065):

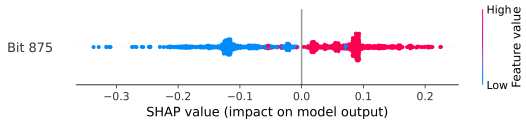
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH2](-[CH2]~*)-[NH]~*	349	304
2	[NH](-[CH](-[CH2]~*)-[c;R](~*)~*)-[C](=[O])-[c;R](~*)~*	1	1
3	[n;R]1(:[cH;R]:[cH;R]:[n;R]:[cH;R]:1)-[CH2]-[CH](~*)~*	6	6

X-FP

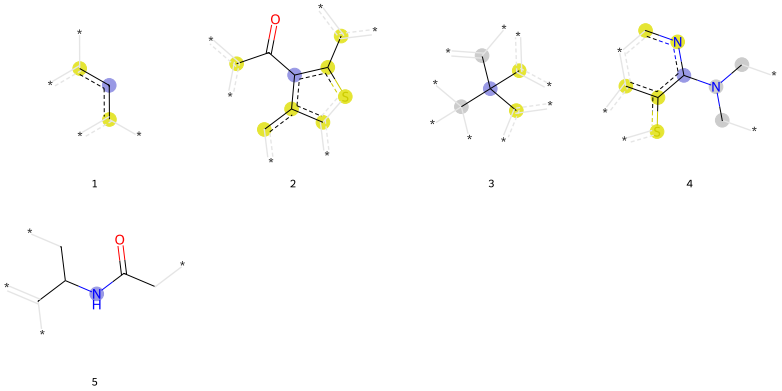
Bit Analysis

Bit 875

SHAP Summary Plot for the Bit 875:



Substructures present:



Substructures frequency table (total compounds: 2065):

Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[cH;R](:[c;R](~*)~*):[c;R](~*)~*	1809	1185
2	[c;R]1(:[c;R](:[s;R]:[c;R](:[c;R]:1:[cH;R]~*)~*)-[c;R](~*)~*)-[C](=[O])-[c;R](~*)~*	1	1
3	[C;R](-[C;R](~*)~*)(-[c;R](~*)~*)(-[c;R](~*)~*)-[C;R](~*)(~*)~*	27	27
4	[c;R]1(:[n;R]:[cH;R]~*~[c;R](:[c;R]:1:[s;R]~*)~*)-[N;R](-[CH2;R]~*)-[CH2;R]~*	2	2
5	[NH](-[CH](-[CH2]~*)-[C](~*)~*)-[C](=[O])-[CH2]~*	12	12

X-FP

Bit Analysis

Notes about substructures rendering:

- The molecule fragment is drawn with the atoms in the same positions as in the original molecule.
- The central atom is highlighted in blue.
- Aromatic atoms are highlighted in yellow.
- Aliphatic ring atoms are highlighted in dark grey.
- Atoms/bonds that are drawn in light grey indicate pieces of the structure that influence the atoms' connectivity invariants but that are not directly part of the fingerprint.

Notes about feature importance:

The feature importance here is done by SHAP TreeExplainer. Following are the notes on interpreting the beeswarm SHAP Summary Plots:

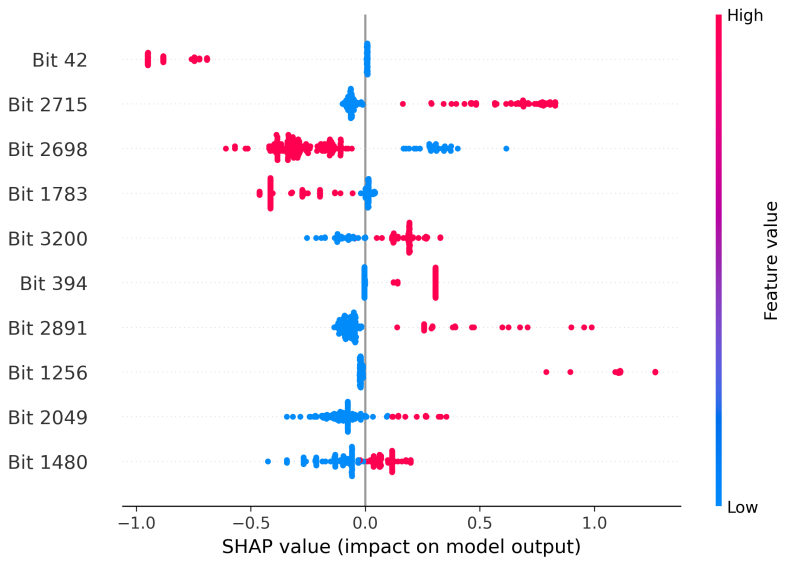
- Each dot is the SHAP value of a bit of a molecule (terms used interchangeably are: sample, observation, data point, entry, compound, etc.) in the dataset.
- The dots representing the same SHAP values of a bit overlap. However, the frequency of these values can be observed as the density of the dots along the x-axis.
- Morgan Fingerprint bits are binary in nature. A red dot shows that the bit is switched on for a molecule, meaning that at least one substructure present in this molecule is encoded by this bit. A blue dot means that the bit is switched off for a molecule and this means that no substructures are present in this molecule which are encoded by this bit.
- For binary classification, SHAP values of a Morgan Fingerprint bit greater than zero describe that the presence or absence of this feature contributes towards the model prediction classes' second category (usually labelled as '1'). While the SHAP values less than zero describe that the presence or absence of this feature contributes towards the first category (usually labelled as '0').

Caution:

- Bar plots will be used instead of beeswarm plots as a default SHAP Summary Plot in case of multi-class classification. X-FP analysis for multi-class classification is still under testing and should be completely avoided.
- X-FP analysis for regression models is also under testing, and while such models and their SHAP analysis are compatible, they should be also avoided for time-being.

Figure S29.: Fingerprint model: X-FP report for the Time-Split Set

Time-Split Data: X-FP Report



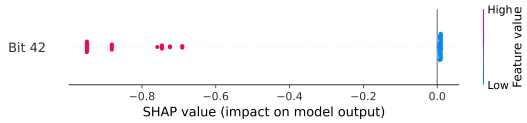
SHAP Summary Plot for the Top 10 Morgan Fingerprint Bits

X-FP

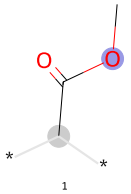
Bit Analysis

Bit 42

SHAP Summary Plot for the Bit 42:



Substructures present:



Substructures frequency table (total compounds: 156):

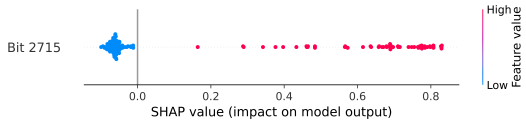
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[O](-[CH3])-[C](=[O])-[CH;R](~*)~*	73	73

X-FP

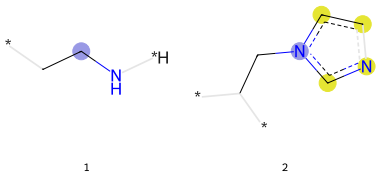
Bit Analysis

Bit 2715

SHAP Summary Plot for the Bit 2715:



Substructures present:



Substructures frequency table (total compounds: 156):

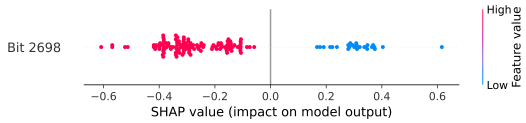
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH2](-[CH2]~*)-[NH]~*	84	71
2	[n;R]1(:[cH;R]:[cH;R]:[n;R]:[cH;R]:1)-[CH2]-[CH](~*)~*	1	1

X-FP

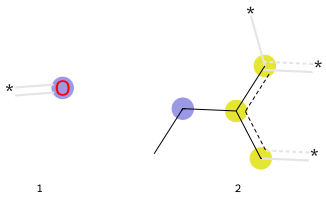
Bit Analysis

Bit 2698

SHAP Summary Plot for the Bit 2698:



Substructures present:



Substructures frequency table (total compounds: 156):

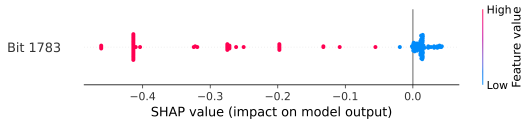
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[O]~*	542	149
2	[CH2](-[CH3])-[c;R](:[cH;R]~*):[c;R](~*)~*	1	1

X-FP

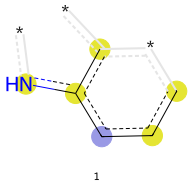
Bit Analysis

Bit 1783

SHAP Summary Plot for the Bit 1783:



Substructures present:



Substructures frequency table (total compounds: 156):

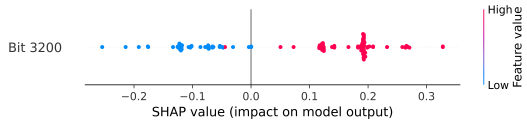
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[cH;R]1:[cH;R]:[cH;R]~*~[c;R](:[c;R]:1:[nH;R]~*)~*	76	76

X-FP

Bit Analysis

Bit 3200

SHAP Summary Plot for the Bit 3200:



Substructures present:

Substructures frequency table (total compounds: 156):

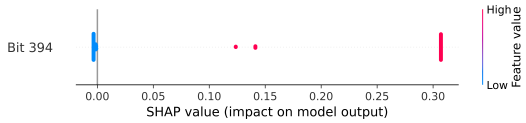
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[NH](~*)~*	174	122

X-FP

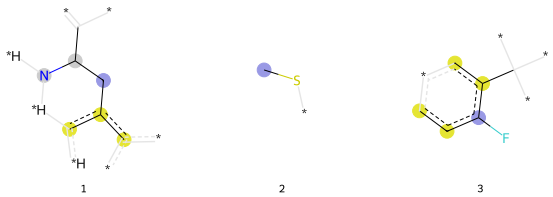
Bit Analysis

Bit 394

SHAP Summary Plot for the Bit 394:



Substructures present:



Substructures frequency table (total compounds: 156):

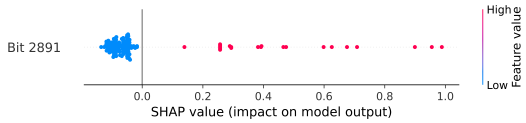
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH2;R]1-[CH;R](-[N;R](~*)~*~[c;R](:[c;R]-1:[c;R](~*)~*)-[C](~*)~*	73	73
2	[CH3]-[S]~*	2	2
3	[c;R]1(:[cH;R]:[cH;R]~*~[cH;R]:[c;R]:1-[C](~*)(~*)~*)-[F]	1	1

X-FP

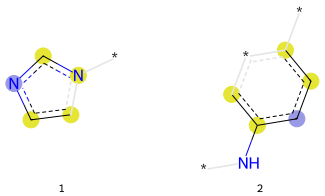
Bit Analysis

Bit 2891

SHAP Summary Plot for the Bit 2891:



Substructures present:



Substructures frequency table (total compounds: 156):

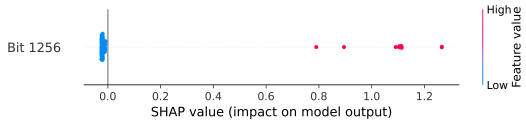
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[n;R]1:[cH;R]:[cH;R]:[n;R](:[cH;R]:1)~*	11	11
2	[cH;R]1:[cH;R]:[c;R](~*)~*~[cH;R]:[c;R]:1-[NH]~*	12	12

X-FP

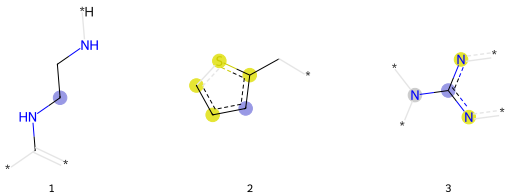
Bit Analysis

Bit 1256

SHAP Summary Plot for the Bit 1256:



Substructures present:



Substructures frequency table (total compounds: 156):

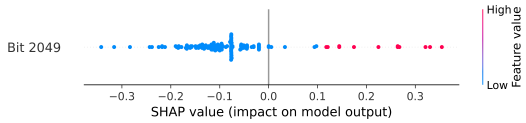
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH2](-[CH2]-[NH]~*)-[NH]-[C](~*)~*	20	10
2	[cH;R]1:[cH;R]:[cH;R]:[s;R]:[c;R]:1-[CH2]~*	3	3
3	[c;R](:[n;R]~*)(:[n;R]~*)-[N;R](~*)~*	3	2

X-FP

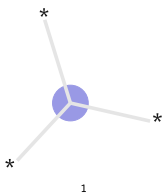
Bit Analysis

Bit 2049

SHAP Summary Plot for the Bit 2049:



Substructures present:



Substructures frequency table (total compounds: 156):

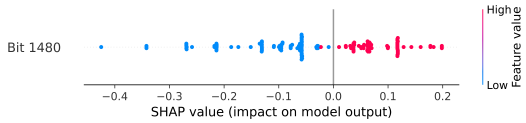
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH](~*)(~*)~*	430	127

X-FP

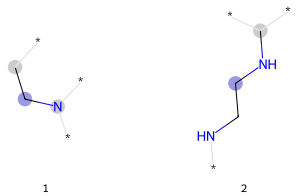
Bit Analysis

Bit 1480

SHAP Summary Plot for the Bit 1480:



Substructures present:



Substructures frequency table (total compounds: 156):

Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH2;R](-[CH2;R]~*)-[N;R](~*)~*	134	61
2	[CH2](-[CH2]-[NH]~*)-[NH]-[CH;R](~*)~*	10	10

X-FP

Bit Analysis

Notes about substructures rendering:

- The molecule fragment is drawn with the atoms in the same positions as in the original molecule.
- The central atom is highlighted in blue.
- Aromatic atoms are highlighted in yellow.
- Aliphatic ring atoms are highlighted in dark grey.
- Atoms/bonds that are drawn in light grey indicate pieces of the structure that influence the atoms' connectivity invariants but that are not directly part of the fingerprint.

Notes about feature importance:

The feature importance here is done by SHAP TreeExplainer. Following are the notes on interpreting the beeswarm SHAP Summary Plots:

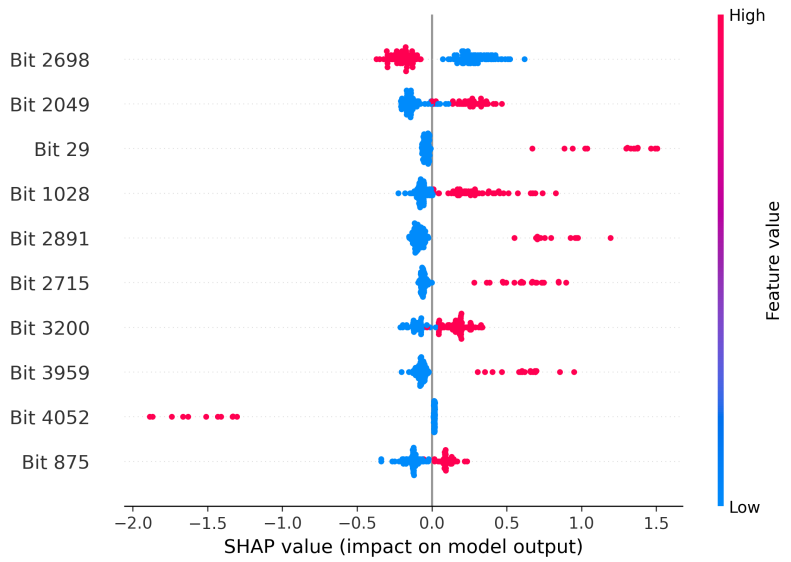
- Each dot is the SHAP value of a bit of a molecule (terms used interchangeably are: sample, observation, data point, entry, compound, etc.) in the dataset.
- The dots representing the same SHAP values of a bit overlap. However, the frequency of these values can be observed as the density of the dots along the x-axis.
- Morgan Fingerprint bits are binary in nature. A red dot shows that the bit is switched on for a molecule, meaning that at least one substructure present in this molecule is encoded by this bit. A blue dot means that the bit is switched off for a molecule and this means that no substructures are present in this molecule which are encoded by this bit.
- For binary classification, SHAP values of a Morgan Fingerprint bit greater than zero describe that the presence or absence of this feature contributes towards the model prediction classes' second category (usually labelled as '1'). While the SHAP values less than zero describe that the presence or absence of this feature contributes towards the first category (usually labelled as '0').

Caution:

- Bar plots will be used instead of beeswarm plots as a default SHAP Summary Plot in case of multi-class classification. X-FP analysis for multi-class classification is still under testing and should be completely avoided.
- X-FP analysis for regression models is also under testing, and while such models and their SHAP analysis are compatible, they should be also avoided for time-being.

Figure S30.: Fingerprint model: X-FP report for the External Set

Purchase Data: X-FP Report



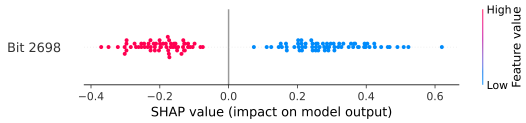
SHAP Summary Plot for the Top 10 Morgan Fingerprint Bits

X-FP

Bit Analysis

Bit 2698

SHAP Summary Plot for the Bit 2698:



Substructures present:

Substructures frequency table (total compounds: 127):

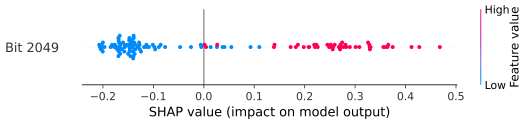
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[O]~*	236	95

X-FP

Bit Analysis

Bit 2049

SHAP Summary Plot for the Bit 2049:



Substructures present:

Substructures frequency table (total compounds: 127):

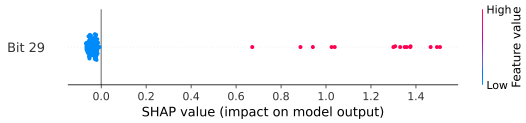
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH](~*)(~*)~*	157	87

X-FP

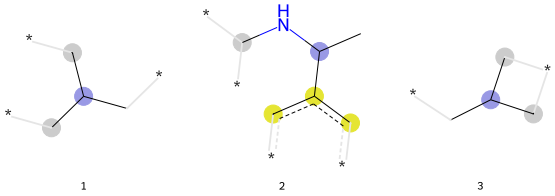
Bit Analysis

Bit 29

SHAP Summary Plot for the Bit 29:



Substructures present:



Substructures frequency table (total compounds: 127):

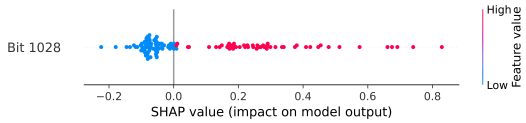
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH;R](-[CH2;R]~*)(-[CH2;R]~*)-[CH2]~*	20	15
2	[CH](-[CH3])(-[NH]-[CH;R](~*)~*)-[c;R](:[cH;R]~*):[cH;R]~*	1	1
3	[CH;R]1(-[CH2;R]~*~[CH2;R]-1)-[CH2]~*	1	1

X-FP

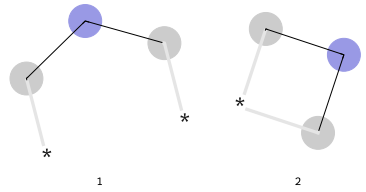
Bit Analysis

Bit 1028

SHAP Summary Plot for the Bit 1028:



Substructures present:



Substructures frequency table (total compounds: 127):

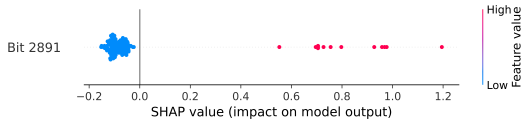
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH2;R](-[CH2;R]~*)-[CH2;R]~*	94	53
2	[CH2;R]1-[CH2;R]~*~[CH2;R]-1	1	1

X-FP

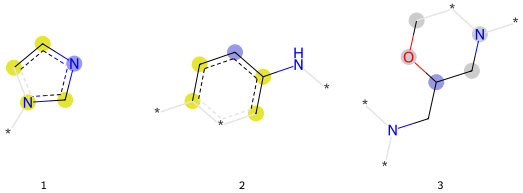
Bit Analysis

Bit 2891

SHAP Summary Plot for the Bit 2891:



Substructures present:



Substructures frequency table (total compounds: 127):

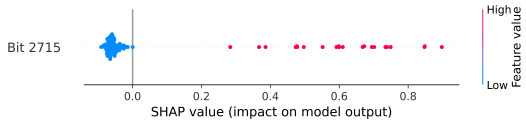
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[n;R]1:[cH;R]:[cH;R]:[n;R](:[cH;R]:1)~*	2	2
2	[cH;R]1:[cH;R]:[c;R](~*)~*~[cH;R]:[c;R]:1-[NH]~*	10	10
3	[CH;R]1(-[CH2;R]-[N;R](~*)~*~[CH2;R]-[O;R]-1)-[CH2]-[N](~*)~*	1	1

X-FP

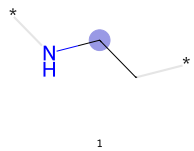
Bit Analysis

Bit 2715

SHAP Summary Plot for the Bit 2715:



Substructures present:



Substructures frequency table (total compounds: 127):

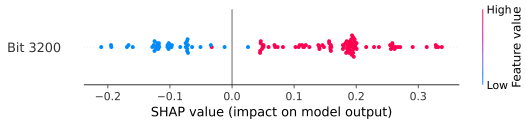
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH2](-[CH2]~*)-[NH]~*	28	26

X-FP

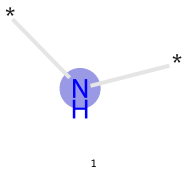
Bit Analysis

Bit 3200

SHAP Summary Plot for the Bit 3200:



Substructures present:



Substructures frequency table (total compounds: 127):

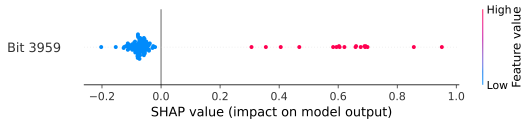
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[NH](~*)~*	108	91

X-FP

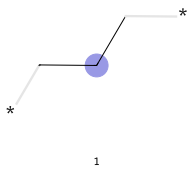
Bit Analysis

Bit 3959

SHAP Summary Plot for the Bit 3959:



Substructures present:



Substructures frequency table (total compounds: 127):

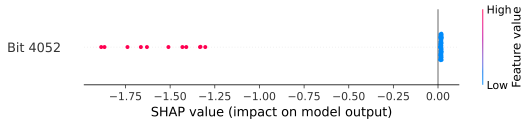
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[CH2](-[CH2]~*)-[CH2]~*	115	64

X-FP

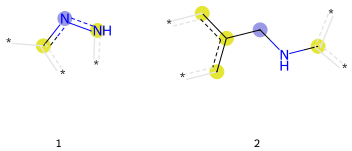
Bit Analysis

Bit 4052

SHAP Summary Plot for the Bit 4052:



Substructures present:



1 2

Substructures frequency table (total compounds: 127):

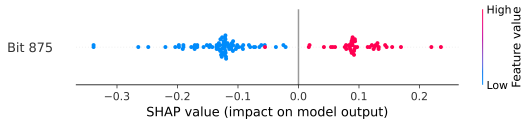
Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[n;R](:[nH;R]~*):[c;R](~*)~*	9	9
2	[CH2](-[NH]-[c;R](~*)~*)-[c;R](:[cH;R]~*):[cH;R]~*	2	2

X-FP

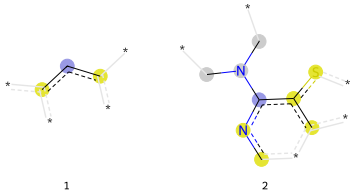
Bit Analysis

Bit 875

SHAP Summary Plot for the Bit 875:



Substructures present:



Substructures frequency table (total compounds: 127):

Serial	Substructures (as SMARTS)	Total occurrences	Unique occurrences
1	[cH;R](:[c;R](~*)~*):[c;R](~*)~*	56	46
2	[c;R]1(:[n;R]:[cH;R]~*~[c;R](:[c;R]:1:[s;R]~*)~*)-[N;R](-[CH2;R]~*)-[CH2;R]~*	1	1

X-FP

Bit Analysis

Notes about substructures rendering:

- The molecule fragment is drawn with the atoms in the same positions as in the original molecule.
- The central atom is highlighted in blue.
- Aromatic atoms are highlighted in yellow.
- Aliphatic ring atoms are highlighted in dark grey.
- Atoms/bonds that are drawn in light grey indicate pieces of the structure that influence the atoms' connectivity invariants but that are not directly part of the fingerprint.

Notes about feature importance:

The feature importance here is done by SHAP TreeExplainer. Following are the notes on interpreting the beeswarm SHAP Summary Plots:

- Each dot is the SHAP value of a bit of a molecule (terms used interchangeably are: sample, observation, data point, entry, compound, etc.) in the dataset.
- The dots representing the same SHAP values of a bit overlap. However, the frequency of these values can be observed as the density of the dots along the x-axis.
- Morgan Fingerprint bits are binary in nature. A red dot shows that the bit is switched on for a molecule, meaning that at least one substructure present in this molecule is encoded by this bit. A blue dot means that the bit is switched off for a molecule and this means that no substructures are present in this molecule which are encoded by this bit.
- For binary classification, SHAP values of a Morgan Fingerprint bit greater than zero describe that the presence or absence of this feature contributes towards the model prediction classes' second category (usually labelled as '1'). While the SHAP values less than zero describe that the presence or absence of this feature contributes towards the first category (usually labelled as '0').

Caution:

- Bar plots will be used instead of beeswarm plots as a default SHAP Summary Plot in case of multi-class classification. X-FP analysis for multi-class classification is still under testing and should be completely avoided.
- X-FP analysis for regression models is also under testing, and while such models and their SHAP analysis are compatible, they should be also avoided for time-being.

Table S1.: List of selected intuitive RDKit descriptors (version 2022.09.05)

1 NumValenceElectrons
2 NumRadicalElectrons
3 NumAliphaticCarbocycles
4 NumAliphaticHeterocycles
5 NumAliphaticRings
6 NumAromaticCarbocycles
7 NumAromaticHeterocycles
8 NumAromaticRings
9 NumHAcceptors
10 NumHDonors
11 NumHeteroatoms
12 NumRotatableBonds
13 NumSaturatedCarbocycles
14 NumSaturatedHeterocycles
15 NumSaturatedRings
16 HeavyAtomMolWt
17 TPSA
18 FractionCSP3
19 HeavyAtomCount
20 NHOHCount
21 NOCount
22 RingCount
23 fr_Al_COO
24 fr_Al_OH
25 fr_Al_OH_noTert
26 fr_ArN
27 fr_Ar_COO
28 fr_Ar_N
29 fr_Ar_NH
30 fr_Ar_OH
31 fr_COO
32 fr_COO2
33 fr_C_O
34 fr_C_O_noCOO
35 fr_C_S
36 fr_HOCCN
37 fr_Imine
38 fr_NH0
39 fr_NH1
40 fr_NH2

```
41 fr_N_O
42 fr_Ndealkylation1
43 fr_Ndealkylation2
44 fr_Nhpyrrole
45 fr_SH
46 fr_aldehyde
47 fr_alkyl_carbamate
48 fr_alkyl_halide
49 fr_allylic_oxid
50 fr_amide
51 fr_amidine
52 fr_aniline
53 fr_aryl_methyl
54 fr_azide
55 fr_azo
56 fr_barbitur
57 fr_benzene
58 fr_benzodiazepine
59 fr_bicyclic
60 fr_diazo
61 fr_dihydropyridine
62 fr_epoxide
63 fr_ester
64 fr_ether
65 fr_furan
66 fr_guanido
67 fr_halogen
68 fr_hdrzine
69 fr_hdrzone
70 fr_imidazole
71 fr_imide
72 fr_isocyan
73 fr_isothiocyan
74 fr_ketone
75 fr_ketone_Topliss
76 fr_lactam
77 fr_lactone
78 fr_methoxy
79 fr_morpholine
80 fr_nitrile
81 fr_nitro
82 fr_nitro_arom
```

```
fr_nitro_arom_nonortho
fr_nitroso
fr_oxazole
fr_oxime
fr_para_hydroxylation
fr_phenol
fr_phenol_noOrthoHbond
fr_phos_acid
fr_phos_ester
fr_piperdine
fr_piperzine
fr_priamide
fr_prisulfonamd
fr_pyridine
fr_quatN
fr_sulfide
fr_sulfonamd
fr_sulfone
fr_term_acetylene
fr_tetrazole
fr_thiazole
fr_thiocyan
fr_thiophene
fr_unbrch_alkane
fr_urea
```

A. Acknowledgments

B. Curriculum Vitae