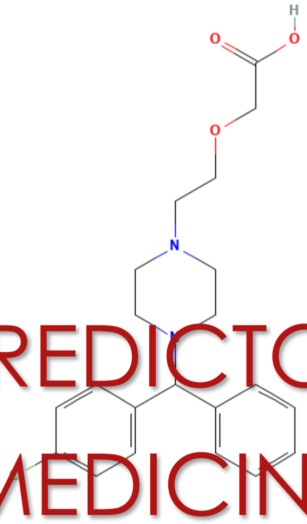
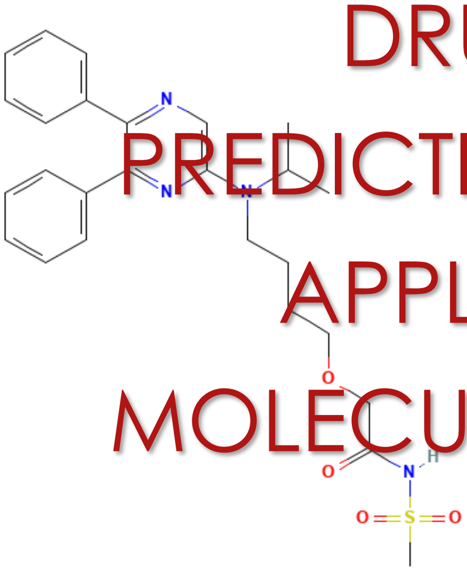



DRUG PREDICTOR, PREDICTING MEDICINAL APPLICATION FROM MOLECULAR STRUCTURE

A drug discovery tool

Master Dissertation

This document serves as the accompanying documentation for the Python application 'Drug Predictor.' Both the document and the app constitute the Master's final project for the Master's degree in Data Science program at KSchool

José Rodríguez Couceiro
jose.r.couceiro@gmail.com

Contents

INTRODUCTION

Drug discovery seeks to identify novel compounds with specific chemical attributes for treating diseases. Over recent years, this quest has seen a significant integration of computer science, notably with the widespread adoption of machine learning techniques. Presently, predictive models rooted in Machine Learning have gained considerable prominence as a cost and time-efficient step before preclinical studies in the pursuit of discovering new drugs.

The greatest complexity in drug discovery is found in the inherently complex process itself and the strict necessary regulations imposed by governing bodies. Currently, the discovery and development of new pharmaceuticals remains a long and exorbitantly expensive process. Typically, the development timeline for a new drug spans a decade to fifteen years of relentless research and testing. The sheer volume of potentially viable molecules for new drug exploration renders wet lab experiments impracticable except for a tiny fraction of them.

The development of new molecules with improved properties holds immense importance in diverse sectors including energy, agriculture, and medicine. However, the sheer vastness of potential molecules to investigate, often denoted as chemical space, is overwhelmingly extensive. Even when narrowing down the chemical space to compounds that follow "Lipinski's rule of five", a standard criterion in drug development, there still exist an astounding 10^{60} possible chemical structures[1].

However, in the past decade, the advancement of Information and Communication Technologies, coupled with a surge in computational capabilities, has paved the way for novel *in silico* methodologies for screening extensive drug libraries. This prelude to preclinical studies not only reduces costs but also broadens the scope of the quest for new medications. Within this context, Machine Learning (ML) techniques have emerged as pivotal players in the pharmaceutical sector, offering the potential to expedite and automate the analysis of the copious data now available. ML falls under the realm of Artificial Intelligence (AI) and focuses on crafting and applying computer algorithms that can learn from unprocessed data to perform specific tasks subsequently. The primary tasks executed by AI algorithms encompass classification, regression, clustering, or pattern recognition within vast datasets. A diverse array of ML methods has been harnessed within the pharmaceutical industry to forecast new molecular traits, biological activities, drug interactions, and adverse effects. Notable examples of these techniques include Naive Bayes, Support Vector Machines, Random Forest, and, more recently, Deep Neural Networks[2].

Biological Challenges of Machine Learning in Drug Discovery

A drug can be defined as a molecule that interacts with a functional entity within an organism, known as a therapeutic target or molecular target, thereby altering its behavior in some manner. The pursuit of new compounds capable of modifying disease progression or enhancing existing treatments remains a primary objective in the fields of chemistry and biology.

[1] Bohacek, R. S., McMartin, C., & Guida, W. C. (1996). The art and practice of structure-based drug design: A molecular modeling perspective. Medicinal Research Reviews, 16(1), 3–50. https://doi.org/10.1002/(SICI)1098-1128(199601)16:1<3::AID-MED1>3.0.CO;2-6

[2] Carracedo-Reboredo, P., Liñares-Blanco, J., Rodríguez-Fernández, N., Cedrón, F., Novoa, F. J., Carballal, A., Maojo, V., Pazos, A., & Fernandez-Lozano, C. (2021). A review on machine learning approaches and trends in drug discovery. In Computational and Structural Biotechnology Journal (Vol. 19, pp. 4538–4558). Elsevier B.V. https://doi.org/10.1016/j.csbj.2021.08.011

The process of developing a new drug can span up to 12 years, with an estimated average cost of approximately one billion euros until it reaches the market. The extensive time and expenses incurred are largely attributed to the substantial number of molecules that fail at one or more stages of development, with only about 1 in 5,000 drugs ultimately making it to market.

These statistics highlight the complexity and costliness of discovering and developing new drugs. Traditionally, this process relied solely on experimental methods. However, recent technological advancements have given rise to the term *in silico,* a concept now common in biology laboratories. It refers to experiments conducted virtually through computer simulations of biological processes, distinct from experiments conducted directly on living organisms (*in vivo*) or in artificial environments outside the organism (*in vitro*).

The intricacies of modern biology have rendered these computational approaches indispensable for biological experimentation, as they allow the processing of vast amounts of data, thereby streamlining and accelerating the drug development process.

The journey to develop a new drug commences with the search for potential candidates, referred to as "hits", through high-throughput screening. Hits are molecules or compounds exhibiting biological activity against a therapeutic or molecular target, the specific agent within the body which the drug is intended to interact with. Following this phase, “leads” are generated through the validation and structural refinement of the selected molecules to enhance their potency against the target. Additionally, they are expected to exhibit favorable pharmacokinetic properties, including adequate absorption, distribution, metabolism, and elimination (ADME) rates, as well as low toxicity and minimal adverse effects.

MACHINE LEARNING METHODOLOGY IN DRUG DISCOVERY

The application of an ML methodology is transversal in any field of research but, specifically, we can delineate the following steps in the ML methodology applied in drug discovery:

Data Collection: The initial step entails acquiring a dataset with specific characteristics. Aside from physical-chemical properties that facilitate absorption, specificity, and low toxicity, the dataset must also feature attributes that allow the easy production and handling in a laboratory setting. This consideration is crucial since the pharmaceutical industry typically deals with small molecules and peptides rather than large proteins or highly complex compounds. To facilitate the handling and analysis of these compounds, the SMILES format is employed for representing small molecule sequences, while FASTA is used for peptide structure representation (**Fig.1**).

Currently, there exist numerous publicly accessible repositories, such as DrugBank, https://go.drugbank.com/ [3], PubChem, https://pubchem.ncbi.nlm.nih.gov/ [4], ChEMBL, https://www.ebi.ac.uk/chembl/ [5], or ZINC, https://zinc.docking.org/ [6], which store copious amounts of valuable data pertinent to drug discovery.

Labeling of different compounds is of paramount importance. Although some ML models operate without labels, supervised learning models are more frequently employed in drug discovery. In such cases, the labels, as defined by researchers, play a vital role in the experimental process.

Mathematical Descriptor Generation: Mathematical descriptors are generated to form a dataset amenable for processing by the ML model. Typically, these descriptors are called fingerprints, and consist of binary sequences encoding the structure and functionality of molecules. This dataset is divided into two subsets: one with a higher percentage of data dedicated to model training, and a smaller one devoted to model testing (**Fig.1**).

Subset Variable Optimization: Within the training set, the search of the most pertinent subset of variables can be carried out, retaining only the essential information (**Fig.1**). Typically, during mathematical descriptor generation, a surplus of numerical variables is presented. The primary objective here is to minimize unnecessary or redundant variables. Several techniques, such as PCA, t-SNE, FS, Autoencoder, etc., can be employed. Feature selection (FS) techniques extract a subset of features from the original set, preserving variable content. This ensures

[3] Wishart, D. S., Feunang, Y. D., Guo, A. C., Lo, E. J., Marcu, A., Grant, J. R., Sajed, T., Johnson, D., Li, C., Sayeeda, Z., Assempour, N., Iynkkaran, I., Liu, Y., Maciejewski, A., Gale, N., Wilson, A., Chin, L., Cummings, R., Le, D., … Wilson, M. (2018). DrugBank 5.0: a major update to the DrugBank database for 2018. Nucleic Acids Research, 46(D1), D1074–D1082. https://doi.org/10.1093/nar/gkx1037

[4] Kim, S., Chen, J., Cheng, T., Gindulyte, A., He, J., He, S., Li, Q., Shoemaker, B. A., Thiessen, P. A., Yu, B., Zaslavsky, L., Zhang, J., & Bolton, E. E. (2019). PubChem 2019 update: improved access to chemical data. Nucleic Acids Research, 47(D1), D1102–D1109. https://doi.org/10.1093/nar/gky1033

[5] Gaulton, A., Bellis, L. J., Bento, A. P., Chambers, J., Davies, M., Hersey, A., Light, Y., McGlinchey, S., Michalovich, D., Al-Lazikani, B., & Overington, J. P. (2012). ChEMBL: a large-scale bioactivity database for drug discovery. Nucleic Acids Research, 40(D1), D1100–D1107. https://doi.org/10.1093/nar/gkr777

[6] Sterling, T., & Irwin, J. J. (2015). ZINC 15 – Ligand Discovery for Everyone. Journal of Chemical Information and Modeling, 55(11), 2324–2337. https://doi.org/10.1021/acs.jcim.5b00559

biological interpretability and is thus widely favored by researchers in their experimental designs[7].

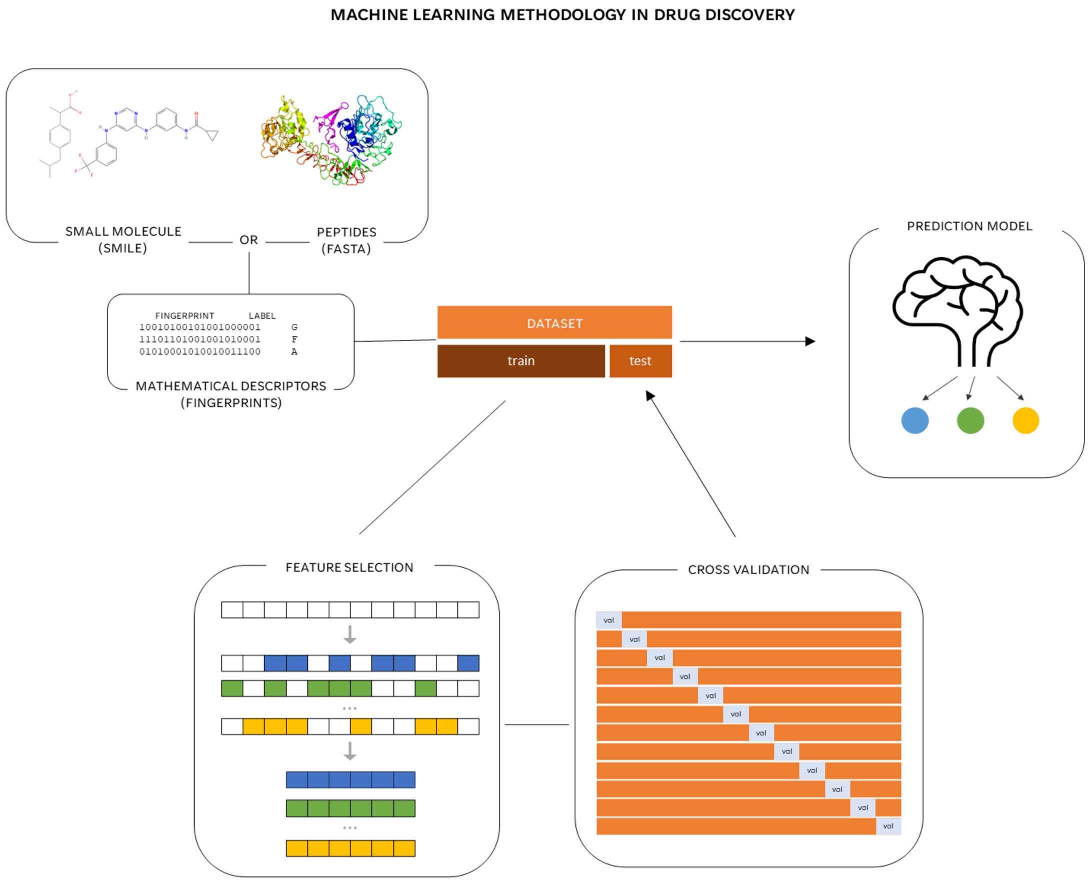


Fig.1. Diagram of the machine learning methodology in drug discovery

Model Training: After identifying the optimal subset of variables, the model is trained. This requires the careful selection of algorithms and their parameters, considering the problem at hand and the volume and nature of available data. Multiple experimental runs are performed using the training data, with a focus on avoiding overtraining to ensure model validity with unknown data. Common practices include the application of cross-validation (CV) techniques to gauge the model's generalization during training. CV assesses performance and estimates performance with unseen data. In each experiment run, the original dataset is once again divided into training and validation subsets. **Fig.1** illustrates the CV technique with 12 runs, where the blue set represents the training set, and the red set denotes the validation set.

Model Validation: Finally, the test set, derived from the original dataset (**Fig.1**), is reinstated, and a conclusive validation of the best model emerging from the CV process is conducted. If the validation results bear statistical significance, it signifies the creation of a novel predictive drug model.

[7] Saeys, Y., Inza, I., & Larrañaga, P. (2007). A review of feature selection techniques in bioinformatics. Bioinformatics, 23(19), 2507–2517. https://doi.org/10.1093/bioinformatics/btm344

Machine learning techniques have found applications across diverse fields, resulting in an increased number of publications, particularly in recent years. However, open-access machine learning publications related to drug development remain relatively sparse.

Input data in molecular Machine Learning predictions

A crucial aspect of model training centers on how molecules are represented through descriptors capable of capturing their inherent properties and structural characteristics. The literature offers an extensive array of molecular descriptors, spanning from straightforward molecule attributes to intricate three-dimensional and complex molecular fingerprint formulations, often stored as vectors containing hundreds or thousands of elements.

It is generally accepted that compounds that share similar physicochemical properties often exhibit comparable biological activities. This principle suggests that molecules with structural resemblances may have analogous therapeutic effects. Consequently, it should be possible, based on structural similarity, to predict a drug's medical indication area[8].

The degree of structural similarity between two compounds can be quantified using the Tanimoto coefficient, with a score higher than 0.85 indicating significant likeness. Notably, when applying this similarity measure, correct ATC code predictions have been achieved for 81% of molecules listed in the PubChem database, highlighting the practicality of using structural similarity for medical indication prediction in drug research.

Quantitative Structure-Activity Relationship (QSAR) models are grounded in these principles. These models establish numerical relationships between the chemical structures of molecules and their biological activity, thus predicting the biological properties of new compounds based on their chemical structure and existing empirical data about the molecule's functionality.

QSAR models amalgamate computer and statistical techniques to theoretically forecast biological activity, facilitating the design of potential new drugs without the trial-and-error process of organic synthesis. Since QSAR exists solely in a virtual environment, it obviates the need for physical resources like equipment, instruments, materials, and laboratory personnel. QSAR modeling such as this, proves to be an especially effective approach when there is a scarcity of adequate experimental data and facilities[8].

To undertake a QSAR study, three types of information are requisite[9]:

1. **Molecular Structure**: Data on the molecular structure of various compounds.
2. **Biological Activity Data**: Information regarding the biological activity of each of the compounds.
3. **Physicochemical Properties**: These properties are defined by a set of numerical variables derived from the molecular structure, virtually generated using computational techniques.

QSAR models are typically used in two different ways:

- In a **prospective approach**, the results, often presented as equations or QSAR models, enable the prediction of the biological activity of as-yet-unsynthesized compounds. These virtual compounds are created quickly and easily, but under the condition of sharing structural characteristics with empirically known

[8] Gramatica, P. (2020). Principles of QSAR Modeling. International Journal of Quantitative Structure-Property Relationships, 5(3), 61–97. https://doi.org/10.4018/IJQSPR.20200701.oa1

[9] Cronin, M. T. D., & Schultz, T. W. (2003). Pitfalls in QSAR. Journal of Molecular Structure: THEOCHEM, 622(1–2), 39–51. https://doi.org/10.1016/S0166-1280(02)00616-4

molecules in order to adhere to the established chemical rules and patterns, or descriptor value ranges.

- The **retrospective approach**, on the other hand, analyzes existing molecules (those subjected to synthesis and bioassays) to discern non-obvious relationships between structures and biological activities.

Molecular simulations conducted via computational tools significantly reduce the time compared to synthesizing and bioassaying new compounds, which could take months or even years. This speed allows for the quick generation of results, which can then be directly translated into ongoing projects of the synthesis laboratory. Consequently, QSAR predicts entirely new, previously unseen structures and presents them to organic chemists for bioassays that either confirm or challenge the values predicted by the QSAR model. In an ideal scenario, this iterative cycle yields superior candidates compared to pure trial-and-error methods. This approach saves time, money, resources, and mitigates the risk of drug development failures.

Advantages of QSAR include its cost-effectiveness since it doesn't require laboratory instruments or chemical reagents. Additionally, free software tools are available for model generation, often equipped with user-friendly interfaces. The construction of molecules and descriptor calculations can also be extremely swift. On the downside, drawbacks encompass the need for training in computational methodologies, dealing with various operating systems and graphical interfaces, managing databases, software development, and resolving computational issues such as compatibility, updates and data formats, along with the necessity of having biological activity data from a single source.

Evolution of Machine Learning in Drug Discovery Over Time

The journey of Machine Learning (ML) algorithms in drug discovery can be traced back to 1964 when Hansch et al.[10] introduced the Hansch equation. This pioneering linear regression model utilized physicochemical descriptors like hydrophobicity, electronic parameters, and steric properties to describe the 2D structure-activity relationship, marking the inception of Quantitative Structure-Activity Relationship (QSAR) studies.

In 1998, Ajay et al.[11] introduced the concept of "Drug-likeness", presenting a model capable of accurately predicting whether a molecule could be considered a drug or not. This achievement, utilizing 1D and 2D molecular descriptors, marked a significant milestone in drug discovery driven by ML algorithms.

Before the year 2000, there were relatively few publications related to ML in drug discovery, primarily due to limited data availability. However, with advancements in biotechnology and computational techniques, a wealth of molecular data became accessible to the public. Initiatives like PubChem, ZINC (initiated in 2004), DrugBank, and ChEMBL (established in 2006 and 2008, respectively) provided standardized repositories of molecular information, transforming the landscape of drug discovery (**Fig.2.**).

The availability of these extensive public databases paved the way for the development and training of new ML models aimed at aiding drug screening. As observed in **Fig.2**, there was a substantial surge in the utilization of ML algorithms between 2004 and 2008. Among these algorithms, Support Vector Machines (SVM) emerged as the most prominent and widely adopted. Additionally, there was a notable turning point in the application of neural networks,

[10] Hansch, Corwin., & Fujita, Toshio. (1964). p -σ-π Analysis. A Method for the Correlation of Biological Activity and Chemical Structure. Journal of the American Chemical Society, 86(8), 1616–1626. https://doi.org/10.1021/ja01062a035

[11] Ajay, Walters, W. P., & Murcko, M. A. (1998). Can We Learn to Distinguish between "Drug-like" and "Nondrug-like" Molecules? Journal of Medicinal Chemistry, 41(18), 3314–3324. https://doi.org/10.1021/jm970666c

particularly after the release of the TensorFlow library in 2008. This development led to a remarkable increase in the usage of Artificial Neural Networks (ANNs) and deep learning models. The versatility of ANNs contributed to this growth, exemplified by models like the Graph Neural Network Model in 2009[12], which enabled the application of molecular graphs as inputs, expanding the scope of research in drug discovery. Subsequent innovations, such as Molecular Graph Convolutions in 2016[13], further advanced the use of convolutional networks for analyzing molecular graphs[14].

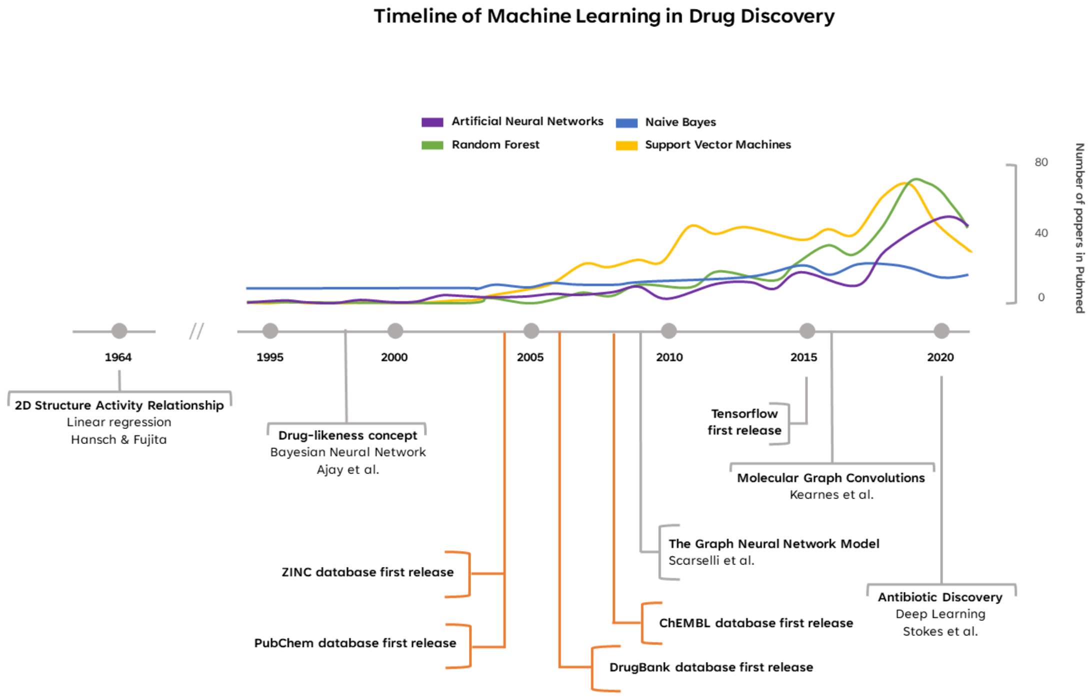


Fig.2. Timeline of machine learning milestones in drug discovery (adapted from Carracedo et al.[15]).

[12] Scarselli, F., Gori, M., Ah Chung Tsoi, Hagenbuchner, M., & Monfardini, G. (2009). The Graph Neural Network Model. IEEE Transactions on Neural Networks, 20(1), 61–80. https://doi.org/10.1109/TNN.2008.2005605

[13] Kearnes, S., McCloskey, K., Berndl, M., Pande, V., & Riley, P. (2016). Molecular graph convolutions: moving beyond fingerprints. Journal of Computer-Aided Molecular Design, 30(8), 595–608. https://doi.org/10.1007/s10822-016-9938-8

[14] Stokes, J. M., Yang, K., Swanson, K., Jin, W., Cubillos-Ruiz, A., Donghia, N. M., MacNair, C. R., French, S., Carfrae, L. A., Bloom-Ackermann, Z., Tran, V. M., Chiappino-Pepe, A., Badran, A. H., Andrews, I. W., Chory, E. J., Church, G. M., Brown, E. D., Jaakkola, T. S., Barzilay, R., & Collins, J. J. (2020). A Deep Learning Approach to Antibiotic Discovery. Cell, 180(4), 688-702.e13. https://doi.org/10.1016/j.cell.2020.01.021

[15] Carracedo-Reboredo, P., Liñares-Blanco, J., Rodríguez-Fernández, N., Cedrón, F., Novoa, F. J., Carballal, A., Maojo, V., Pazos, A., & Fernandez-Lozano, C. (2021). A review on machine learning approaches and trends in drug discovery. In Computational and Structural Biotechnology Journal (Vol. 19, pp. 4538–4558). Elsevier B.V. https://doi.org/10.1016/j.csbj.2021.08.011

MOLECULAR DESCRIPTORS (MD)

MDs are indispensable components in various research areas. They are numerical representations of molecules that quantitatively capture their physicochemical information. To achieve this, the information embedded in a molecule's structure is theoretically converted into one or more numerical values. These values are then employed to establish quantitative relationships with biological activities or other experimental attributes. However, it's important to note that MDs don't encapsulate all the information within a molecule, but rather a portion of it that can be extracted through experimental measurements.

Since their initial application, thousands of molecular descriptors have been defined, each encoding molecules in different ways. They can provide a generic overview of the entire molecule (1D descriptors), which involves simpler calculations, or define properties based on two- and three-dimensional (2D and 3D) structures. The latter offer more specific characteristics but require more complex calculations.

It has been argued that the existing array of atomic and molecular descriptors is sufficient for drug discovery purposes. However, one of the reasons for model inadequacy may lie in inappropriate selection of structural descriptors. This might be due to wrongdoings in the selection process or the limitations of a specific descriptor in describing the phenomenon being researched. Hence, there is a continuous quest for new structural or atomic descriptors that can be utilized in QSAR-based model studies.

Fingerprinting

Fingerprints (FPs) represent a distinctive type of molecular descriptors, enabling the swift and efficient portrayal of a molecule's structure through a sequence or vector of fixed-length bits. These bits signify the presence or absence of internal substructures or functional groups. This method of molecular coding proves highly effective for the storage, manipulation, and comparison of data containing molecular information. Nonetheless, fingerprints derived from chemical structures may overlook the biological context, creating a disconnect between molecular structure and biological activity[16].

A diverse array of FPs exists, ranging from the most basic, which catalog 2D substructures (e.g., MACCS), to more sophisticated versions that incorporate 3D details about molecular conformation[17]. Below is a concise overview of the Fingerprint Patterns (FPs) employed in this study:

Dictionary type fingerprints

In dictionary-based fingerprints, also known as structural keys fingerprints, each designated bit position signifies the existence (1) or non-existence (0) of predetermined functional groups or substructure patterns. The fingerprinting algorithms typically employ hash functions to transform organized fragments into n-bit strings. These key-associated bitstrings are apt for bitwise evaluation of molecular structural attributes,

[16] Yang, J., Cai, Y., Zhao, K., Xie, H., & Chen, X. (2022). Concepts and applications of chemical fingerprint for hit and lead screening. Drug Discovery Today, 27(11), 103356. https://doi.org/10.1016/j.drudis.2022.103356

[17] Carracedo-Reboredo, P., Liñares-Blanco, J., Rodríguez-Fernández, N., Cedrón, F., Novoa, F. J., Carballal, A., Maojo, V., Pazos, A., & Fernandez-Lozano, C. (2021). A review on machine learning approaches and trends in drug discovery. In Computational and Structural Biotechnology Journal (Vol. 19, pp. 4538–4558). Elsevier B.V. https://doi.org/10.1016/j.csbj.2021.08.011

enabling swift filtration and exploration of molecular structures within chemical repositories.

MACCS keys

MACCS (Molecular ACCess System) Keys are among the most widely utilized structural keys. They are occasionally referred to as MDL keys, named after the company that developed them. Two sets of MACCS keys are available, one with 960 keys and the other containing a subset of 166 keys. However, only the definitions of the shorter 166 keys are publicly accessible and have been integrated into open-source chemoinformatics software packages.

PubChem fingerprints

PubChem houses a substantial volume of molecular data that can be freely accessed and downloaded. In the PubChem database, a substructure refers to a fragment of a chemical structure for which PubChem generates a fingerprint. PubChem Fingerprints are 881-bit long structural keys employed by PubChem for conducting similarity searches.

Circular fingerprints

Circular fingerprints produce individual structural fragments that have a circular shape. The algorithm for circular fingerprints typically focuses on each non-hydrogen atom or molecular fragment within a compound. It iteratively extends the molecular fragment to its neighboring atoms based on predefined rules until all fragments of the entire compound are exhaustively enumerated. These dynamically generated fingerprint patterns exhibit elevated specificity for compounds with intricate structures, such as natural products.

Extended Connectivity Fingerprints (Morgan Fingerprints)

Extended Connectivity Fingerprints (ECFP) are topological fingerprints designed for molecular characterization, particularly tailored for structure-activity modeling. ECFPs offer advantages such as rapid computation, adaptability to various molecular characteristics (including stereochemistry), and the ability to represent both the presence and absence of functionality. They are widely used in drug discovery for similarity searches and are valuable for constructing QSAR models, as they capture atomic environments around each atom in a molecule.

Topological fingerprints:

Topological fingerprints stem from graph theory principles. In brief, a molecular graph is a mathematical representation that encapsulates a molecule's topological and physicochemical traits. Typical topological characteristics of compounds encompass: (1) atom categorization; (2) bonding pattern of non-hydrogen atoms; (3) topological distance between atom pairs; (4) atom eccentricity; and (5) bond and atom weights determined through defined custom methodologies.

Topological Torsion Fingerprints

Topological torsion fingerprints were initially introduced to provide short-range information about the torsion angles within a molecule, complementing the predominantly long-range relationships captured by atom pair fingerprints.

Atom Pair Fingerprints

Atom pairs are fingerprints rooted in topological routes, representing all conceivable connectivity routes defined by a specific fingerprint within an input compound. Their primary focus centers on the chemical connectivity information of synthetic compounds. The 2D version of these fingerprints, employed in this study, is defined in terms of the atomic environment and the shortest path separations between all pairs of atoms within the topological representation of a structure. It encodes 780 pairs of atoms across various topological distances.

Avalon Fingerprints

Avalon fingerprints were developed by the Novartis Avalon Datawarehouse Project to support various applications in drug discovery and molecular design. Avalon fingerprints are hybrid fingerprints in which both the path-based and substructure-based features are calculated to comprehensively quantify the physical molecular properties[18].

Lipinski rule and ADMET

The concept of drug similarity, based on the analysis of the physicochemical properties and structural characteristics of existing or potential compounds, has been widely employed to screen out compounds with undesirable attributes in terms of Administration, Distribution, Metabolism, Elimination, and Toxicity (ADMET). Understanding the various phases of ADME that a drug undergoes after administration to an individual is fundamental in the development of new compounds[19]. Even if a drug is demonstrated to be extremely efficient in its interaction with the target, any alterations in these stages (e.g., excretion issues, increased distribution in obese individuals, absorption challenges due to gastrointestinal pathology, or metabolic issues arising from liver malfunction) can impact the final plasma concentration of the drug, thereby altering the expected organism response. Thus, it is crucial in the early research stages to predict the pharmacokinetic properties of a compound.

In order to establish criteria for identifying molecules with promising drug potential and suitable ADMET properties, Dr. Christopher Lipinski introduced what is now known as the Lipinski rule. This set of guidelines, alternatively referred to as Pfizer's rule of five or simply the rule of five (RO5), correlates physicochemical properties with pharmacokinetic behaviors.

According to Lipinski's rule, a compound deemed suitable for oral drug administration should adhere to specific criteria, allowing for no more than one violation of the following parameters: a maximum of 5 hydrogen bond donors, a maximum of 10 hydrogen bond acceptors, a molecular weight not exceeding 500 Da, and an octanol-water partition coefficient (log P) not surpassing 5.

These criteria are not arbitrary but rather align with physiological responses within the organism. For example, the octanol-water partition coefficient is a key indicator of a substance's hydrophobicity or affinity for lipids when dissolved in water. Compounds with high log P values tend to accumulate in the lipid-rich parts of organisms, potentially causing toxicity. Conversely, compounds with low log P values distribute more readily in water or air and can be eliminated from the organism without significant accumulation. A similar rationale guides the restriction on

[18] Gedeck, P., Rohde, B., & Bartels, C. (2006). 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, 46(5), 1924–1936. https://doi.org/10.1021/ci050413p

[19] Burton ME. (2006). Applied pharmacokinetics & pharmacodynamics: principles of therapeutic drug monitoring. Lippincott Williams & Wilkins.

molecular size to not exceed 500 Da, since molecules of bigger size could have troubles circulating the blood stream, and the limitation on hydrogen bond-forming capacity, as molecules with a high propensity for hydrogen bonding struggle to cross cellular membranes[20].

[20] Lipinski, C. A., Lombardo, F., Dominy, B. W., & Feeney, P. J. (2001). Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings 1PII of original article: S0169-409X(96)00423-1. The article was originally published in Advanced Drug Delivery Reviews 23 (1997) 3–25. 1. Advanced Drug Delivery Reviews, 46(1–3), 3–26. https://doi.org/10.1016/S0169-409X(00)00129-0

STATE OF THE ART, OBJECTIVE AND RATIONALE

While several studies have demonstrated the superior performance of Deep Neural Network (DNN) models over other Machine Learning (ML) algorithms in specific cases[21] [22] [23], their adoption in the industry remains limited due to their inherent complexity compared to simpler ML models. Notably, these studies dealt with input data comprising around 10^5 compounds, each with efficacy data related to the inhibition of target proteins.

Efforts to harness DNNs for predicting the biological activity of chemical compounds labeled with medical indications, which are datasets that comprise just a few thousand molecules, have yielded success in recent research, such as the work conducted by Chen et al. (2012)[24] and Lumini et al. (2019)[25], which outperformed previous studies utilizing alternative ML approaches. Notably, Gitter et al. (2019)[26] broke new ground by successfully applying a 2D Convolutional Neural Network (CNN) to this specific problem.

The primary objective of this project is to implement a 1D CNN into a drug discovery methodology that predicts the medical indication of a compound based on its structure, as encoded in molecular fingerprints. Given that these fingerprints are binary arrays representing the three-dimensional structure of molecules, we believe that the application of a 1D CNN has the potential to deliver results on par with a 2D CNN. Importantly, this approach offers advantages in terms of simpler programming and reduced computational resource requirements.

[21] Ramsundar, B., Liu, B., Wu, Z., Verras, A., Tudor, M., Sheridan, R. P., & Pande, V. (2017). Is Multitask Deep Learning Practical for Pharma? Journal of Chemical Information and Modeling, 57(8), 2068–2076. https://doi.org/10.1021/acs.jcim.7b00146

[22] Korotcov, A., Tkachenko, V., Russo, D. P., & Ekins, S. (2017). Comparison of Deep Learning with Multiple Machine Learning Methods and Metrics Using Diverse Drug Discovery Data Sets. Molecular Pharmaceutics, 14(12), 4462–4475. https://doi.org/10.1021/acs.molpharmaceut.7b00578

[23] Lenselink, E. B., ten Dijke, N., Bongers, B., Papadatos, G., van Vlijmen, H. W. T., Kowalczyk, W., Ijzerman, A. P., & van Westen, G. J. P. (2017). Beyond the hype: deep neural networks outperform established methods using a ChEMBL bioactivity benchmark set. Journal of Cheminformatics, 9(1). https://doi.org/10.1186/s13321-017-0232-0

[24] Chen, L., Zeng, W. M., Cai, Y. D., Feng, K. Y., & Chou, K. C. (2012). Predicting anatomical therapeutic chemical (ATC) classification of drugs by integrating chemical-chemical interactions and similarities. PLoS ONE, 7(4). https://doi.org/10.1371/journal.pone.0035254

[25] Lumini, A., & Nanni, L. (2019). Convolutional Neural Networks for ATC Classification. Current Pharmaceutical Design, 24(34), 4007–4012. https://doi.org/10.2174/1381612824666181112113438

[26] Meyer, J. G., Liu, S., Miller, I. J., Coon, J. J., & Gitter, A. (2019). Learning Drug Functions from Chemical Structures with Convolutional Neural Networks and Random Forests. Journal of Chemical Information and Modeling. https://doi.org/10.1021/acs.jcim.9b00236

CODING AND APP DEVELOPMENT

Raw data

Selection of labels

For the purpose of this work, a database which includes therapeutic indication labels of molecules was needed since a supervised model was going to be implemented. The Anatomical Therapeutic Chemical (ATC) classification system was selected as label since it is a World Health Organization (WHO) standard for medical indication. In this system, the active substances are divided into different groups according to the organ or system on which they act and their therapeutic, pharmacological and chemical properties. Drugs are classified in groups at five different levels. In the 1st level, the system has fourteen main anatomical or pharmacological groups (**Table 1**).

ATC Code
A: Alimentary tract and metabolism
B: Blood and blood forming organs
C: Cardiovascular system
D: Dermatologicals
G: Genito urinary system and sex hormones
H: Systemic hormonal preparations excluding sex hormones and insulins
J: Antiinfectives for systemic use
L: Antineoplastic and immunomodulating agents
M: Musculo-skeletal system
N: Nervous system
P: Antiparasitic products insecticides and repellents
R: Respiratory system
S: Sensory organs
V: Various

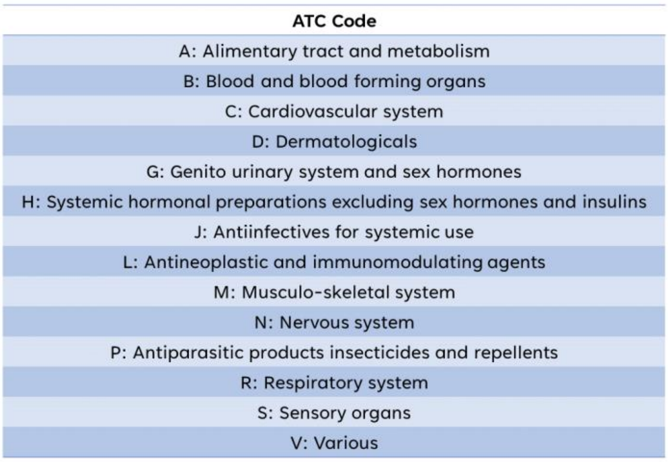
Table 1. The Anatomical Therapeutic Chemical classification (ATC code)

Subsequent levels ascribe molecules to different pharmacological groups each step more specific until reaching the 5th level, which is unique to each molecule. Therefore, the first level of ATC classification was chosen as label for our model. However, to ensure we could include as many molecules as possible, we modified this code to accommodate molecules from databases that used different labeling schemes (as described below and Table 1).

Data sources

Nowadays, several publicly accessible repositories are available, such as DrugBank (https://go.drugbank.com/), PubChem (https://pubchem.ncbi.nlm.nih.gov/), ChEMBL (https://www.ebi.ac.uk/chembl/), and ZINC (https://zinc.docking.org/). DrugBank contains 3,024 molecules labeled with an ATC code out of its repository of 16,000 molecules. PubChem offers 4,491 molecules with ATC labels from its extensive database of approximately 17 million compounds. Both databases were used in this project with the aim of obtaining the highest number of labeled molecules as possible.

Despite our choice of ATC as the standard label for our model, previous studies had managed to extract up to 6,935 molecule records with medical indications from PubChem by using the Medical Subject Headings (MeSH)[27]. We incorporated the dataset from this previous work (referred to as “Gitter dataset”) into our data and merged it with the DrugBank and PubChem datasets to identify non-redundant compounds. Since Gitter's dataset did not use ATC labels, the categories present in it were either matched or added to the ATC code, resulting in a modified ATC (MATC) system comprising 15 categories (**Table 2**).



ATC Code
A: Alimentary tract and metabolism
B: Blood and blood forming organs
C: Cardiovascular system
D: Dermatologicals
G: Genito urinary system and sex hormones
H: Systemic hormonal preparations excluding sex hormones and insulins
J: Antiinfectives for systemic use
L: Antineoplastic and immunomodulating agents
M: Musculo-skeletal system
N: Nervous system
P: Antiparasitic products insecticides and repellents
R: Respiratory system
S: Sensory organs
V: Various

Table 2. Modified ATC (MATC code) used in this study.

Development of a predicting application

The project was entirely developed in Python, utilizing version 3.10.12. In addition to standard data science libraries such as Pandas, ScikitLearn, and Keras/Tensorflow, two chemistry-specific packages were essential: PubChemPy, a library with functions related to the control of the PubChem API, and RDKit toolkit, a suite of packages enabling various chemical computations.

For the backend, a Kedro app was constructed to handle all the necessary coding, processing the raw data and producing a 1D CNN model. On the frontend, two complementary Python applications were developed. These applications utilize the data provided by the Kedro application to present information to the end user via a Streamlit interface:

1. Drug Predictor: this app employs the model created by the Kedro app to generate predictions based on the data inputted by the end user.
2. Visualization: this app displays graphs to the end user, showcasing information regarding the model's training and performance.

Throughout this section, the steps followed for the construction of these applications will be detailed.

Data processing

Preprocessing of raw datasets

The preprocessing of the raw datasets aimed to obtain a dataset containing the following information for each molecule:

[27] Meyer, J. G., Liu, S., Miller, I. J., Coon, J. J., & Gitter, A. (2019). Learning Drug Functions from Chemical Structures with Convolutional Neural Networks and Random Forests. Journal of Chemical Information and Modeling. https://doi.org/10.1021/acs.jcim.9b00236

- A distinctive descriptor: CID number provided by PubChem or another unique identifier.
- Isomeric SMILES: a singular representation of the molecule in a unified string.
- A designated label: either the ATC code or other label.

In anticipation of subsequent analyses, it was deemed prudent to additionally capture the properties utilized in determining Lipinski's rule, which encompass H bond acceptor count, H bond donor count, molecular weight, and Log P. Furthermore, a Boolean indicator 'RuleFive', signifying whether the molecule complies with Lipinski's rule, was obtained either by extracting it from the original dataset or through deduction from the values.

Drugbank dataset

The Drugbank dataset was initially provided in XML format, necessitating the utilization of the 'xmlschema' package to parse it into a list. Each element within this list represented a dictionary encapsulating the properties of an individual molecule. These dictionaries' keys were used to extract the relevant properties. This way, small molecules were selected from the dataset while bigger biomolecules (catalogized as 'biotech' in the dataset) were discarded, resulting in a subset of 12,227 molecules out of the original 15,235. Out of this smaller set, the information of interest was extracted, using the molecule's InChiKey as unique identifier, since the CID number was not provided.

PubChem dataset

The PubChem dataset was available for download in CSV format from the database webpage. All the necessary properties could be obtained by selecting the corresponding columns. However, two properties, the ATC code and 'RuleFive', required special attention. The ATC Code was obtained by sending a request to the PubChem API using the 'request' library. On the other hand, a function was crafted to deduce the 'RuleFive' value from the Lipinski properties.

Finally, the **Gitter dataset** did not necessitate any preprocessing steps as all the data necessary was already available in it.

Processing of intermediary datasets

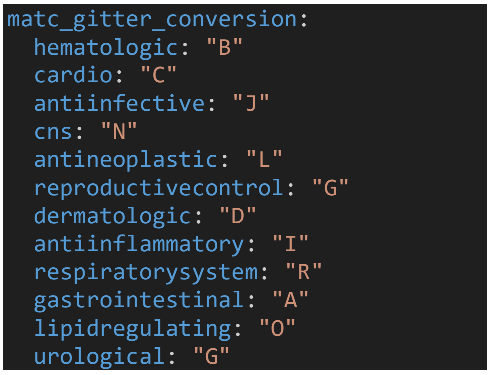
The processing of the intermediary datasets aimed at standardizing the three data sources to ensure they contain the same columns for seamless integration. Although a detailed explanation of eliminating unnecessary columns was omitted for brevity, the standardization comprised the following steps:

- The use of The CID number as unique identifier
- Setting the MATC code as label
- Adding a column which includes the explanation of the MATC code.

Gitter dataset

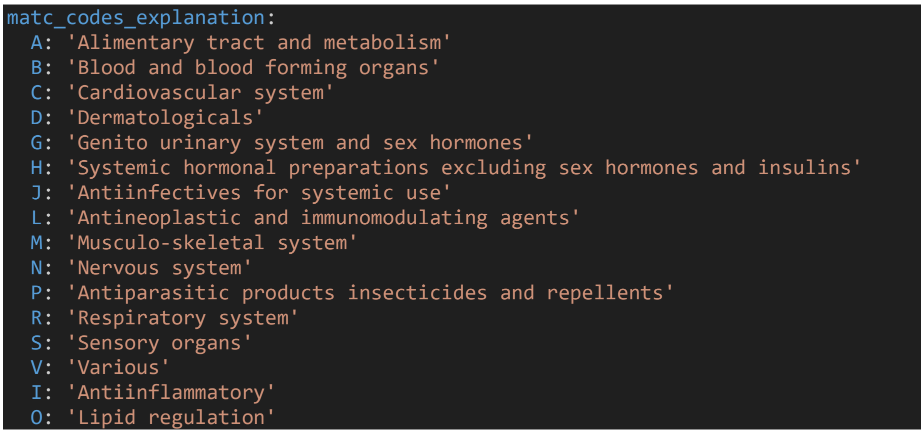
The following specific modifications were made in the Gitter dataset:

- 'RuleFive' Category Addition: the 'RuleFive' category was incorporated using the dedicated function designed for this purpose.
- Conversion of Label to MATC Code: the label was transformed into the corresponding MATC code by mapping through the provided dictionary:



```
matc_gitter_conversion:
  hematologic: "B"
  cardio: "C"
  antiinfective: "J"
  cns: "N"
  antineoplastic: "L"
  reproductivecontrol: "G"
  dermatologic: "D"
  antiinflammatory: "I"
  respiratorysystem: "R"
  gastrointestinal: "A"
  lipidregulating: "O"
  urological: "G"
```

- Addition of MATC Code Explanation: an explanation for the MATC code was appended by mapping through the provided dictionary:



```
matc_codes_explanation:
  A: 'Alimentary tract and metabolism'
  B: 'Blood and blood forming organs'
  C: 'Cardiovascular system'
  D: 'Dermatologicals'
  G: 'Genito urinary system and sex hormones'
  H: 'Systemic hormonal preparations excluding sex hormones and insulins'
  J: 'Antiinfectives for systemic use'
  L: 'Antineoplastic and immunomodulating agents'
  M: 'Musculo-skeletal system'
  N: 'Nervous system'
  P: 'Antiparasitic products insecticides and repellents'
  R: 'Respiratory system'
  S: 'Sensory organs'
  V: 'Various'
  I: 'Antiinflammatory'
  O: 'Lipid regulation'
```

PubChem dataset:

The PubChem dataset underwent two key transformations:

- Conversion of ATC Code to MATC Code: the ATC code was simplified to its first letter, effectively transforming it into the corresponding MATC code.
- Inclusion of MATC Code Explanation: as for the Gitter dataset, a mapping was used to append the MATC code explanation for each molecule.

Drugbank dataset:

Preprocessing of the Drugbank dataset involved several steps:

- Switching InChiKey to CID Number as Identifier: the InChiKey was replaced with the CID number as the primary identifier for each molecule. However, obtaining CID numbers from the SMILES using PubChemPy proved to be slow. To speed up the process, this conversion was selectively performed only for molecules absent in the PubChem dataset. Therefore, entries lacking an ATC code or SMILES were removed. Additionally, ATC codes already present in the processed PubChem dataset were also excluded.
- Transformation of ATC Code to MATC Code: as done in the PubChem dataset.
- Append MATC Code Explanation: As for the Gitter dataset, the MATC code explanation was added using the provided mapping for a comprehensive dataset.

Joining datasets

The integration of datasets was carefully executed in sequential steps to prevent redundancy and duplication of data. Given that some molecules could share identifiers but have present some difference in any of their properties, eliminating duplicates after joining posed a challenge. To circumvent this, a cautious approach was adopted that ensured that molecules sharing CID numbers, but potentially differing properties, were handled appropriately. Specifically:

- CID numbers from the Drugbank dataset were excluded if they were already present in the PubChem dataset, minimizing the possibility of duplicate entries arising from varying ATC codes for the same molecule across different databases, or differing in some of the physicochemical properties (notably, the Log P varies greatly from database to database) while sharing CID number and ATC code.
- The merging between Drugbank and PubChem dataset was carried on using the 'concat' function from the Pandas library, which effectively concatenated the datasets while adhering to the defined principles to maintain data integrity and coherence throughout the integration process.

The resulting dataset from the Drugbank and PubChem merge was then integrated with the Gitter dataset, following the same principle of avoiding duplicate CID numbers. Thanks to the combination of all three sources, we obtained a dataset comprising more than 10.000 unique labeled molecules.

Molecular Computations

The second section of programming focuses on generating a dataset that presents the following properties:

1. The CID number of the molecule.
2. Each fingerprints listed above for every molecule.
3. The encoded MATC code for each molecule in numerical format.

The rationale behind creating this dataset, referred to as the ‘input table’, is to utilize it in the subsequent section to identify the most effective fingerprints for a 1D CNN model with this particular labels.

The steps involved in creating this table are as follows:

- **Adding a column containing the RDKit molecule object** of each molecule, obtained from its SMILES. This is a straightforward task thanks to the function ‘AddMoleculeColumToFrame’ included in the ‘PandasTools’ package integrated into the RDKit toolkit. This function enables the creation of an entire column with a single function.
- **Developing functions to retrieve the fingerprints**, either based on the CID number or the RDKit molecule object. Functions to obtain the fingerprints had to be custom-built in a specific manner, utilizing functions from various packages within the RDKit toolkit or PubChemPy. These functions were mapped either to the RDKit molecule object or the CID number to generate the corresponding columns. Notably, fetching PubChem fingerprints using PubChemPy is an excruciatingly slow process.
- **Encoding the label to make it suitable for a CNN model**: the label column was encoded using the ‘LabelEncoder’ class from the Scikit-Learn ‘preprocessing’ package.

Finally, all unnecessary columns were eliminated, and the table was saved as a pickle. This choice was made to preserve the fingerprint information, as fingerprints are stored in the form of NumPy arrays.

Building a CNN model

The third programming block is focused on constructing a Convolutional Neural Network (CNN) model that categorizes fingerprints into specific medical indications. To achieve this, a two-step approach was employed:

Fingerprint Selection: in this step, the most effective fingerprint type was identified among the retrieved options. A selection prediction model was established, emphasizing speed over optimization. Different sets of fingerprints were introduced to this model, and the resulting accuracies were recorded. The fingerprint type yielding the highest accuracy was chosen for subsequent steps.

Definitive Model Construction: to build the final model, the hyperparameters were selected using the 'RandomSearch' class from the 'keras_tuner' package. Subsequently, the model underwent training with a focus on optimizing the weights. Finally, predictions from the model using the test data were analyzed, and the model was saved for integration into an application.

Data Preparation

The arrays containing various fingerprints needed to be preprocessed to suit both the fingerprint-selection CNN model and the definitive model.

Train/Test Data Splitting

Firstly, they were divided into 'train' and 'test' subsets. To achieve this, two functions were developed. The first function, named 'train_test_split_column,' takes a specific column of a dataframe as X and the label column as y. It utilizes these inputs for the 'train_test_split' function from 'sklearn.model_selection,' producing a tuple that holds the splits for both X and y. The second function, 'train_test_split_columns,' employs the aforementioned function to iterate through a list of columns, which can be specified in the Kedro catalog. It returns tuples of train/test splits as values in a dictionary, with the fingerprint name serving as the key.

Reshaping

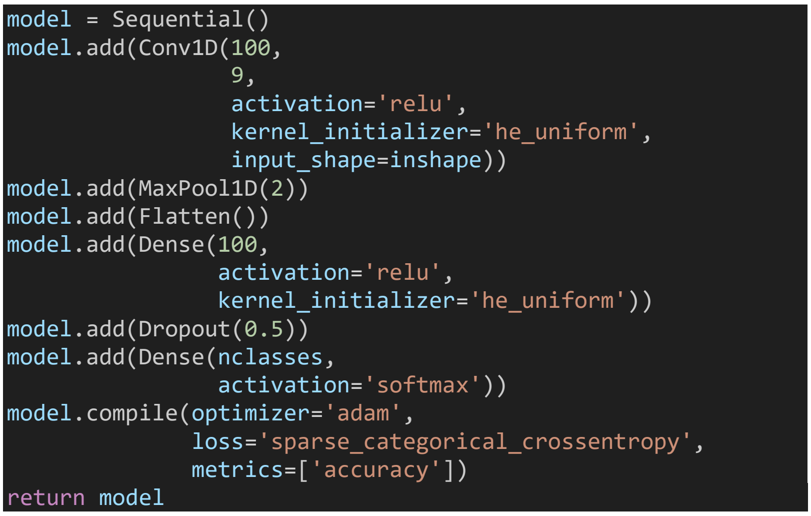
Given that the input data consists of arrays within arrays, they required reshaping before being fed into the model to analyze them one array at a time. For this purpose, a function named 'reshape_input' was created. This function performs the necessary transformation on the tuple containing the train/test splits and returns another tuple with the essential information for fitting a CNN model: the reshaped arrays for X_train and X_test, the internal shape of each array in tuple format, and the total number of classes available as an integer.

Selection of the Optimal Fingerprints

The strategy to identify the best fingerprints involved fitting a CNN model with each fingerprint present in the dataframe (or a selection defined in Kedro's catalog). Three core functions were involved in this process:

'build_selection_model':

This function encompasses the code for constructing a 1D CNN using the 'Sequential' class from 'sklearn.keras.models.' The model consists of a single convolutional layer with suboptimal standard hyperparameters:



```
model = Sequential()
model.add(Conv1D(100,
                 9,
                 activation='relu',
                 kernel_initializer='he_uniform',
                 input_shape=inshape))
model.add(MaxPool1D(2))
model.add(Flatten())
model.add(Dense(100,
                activation='relu',
                kernel_initializer='he_uniform'))
model.add(Dropout(0.5))
model.add(Dense(nclasses,
                activation='softmax'))
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
return model
```

'build_array_dic':

Utilizing the information obtained from the dictionary returned by 'train_test_split_columns', this function utilizes 'reshape_input' to prepare the data for fitting with the model. It then employs 'build_selection_model' to create a model with appropriate values for input shape and number of classes. The function returns a dictionary with reshaped X_train and X_test, y_train and y_test, and the model, using fingerprint names as keys.

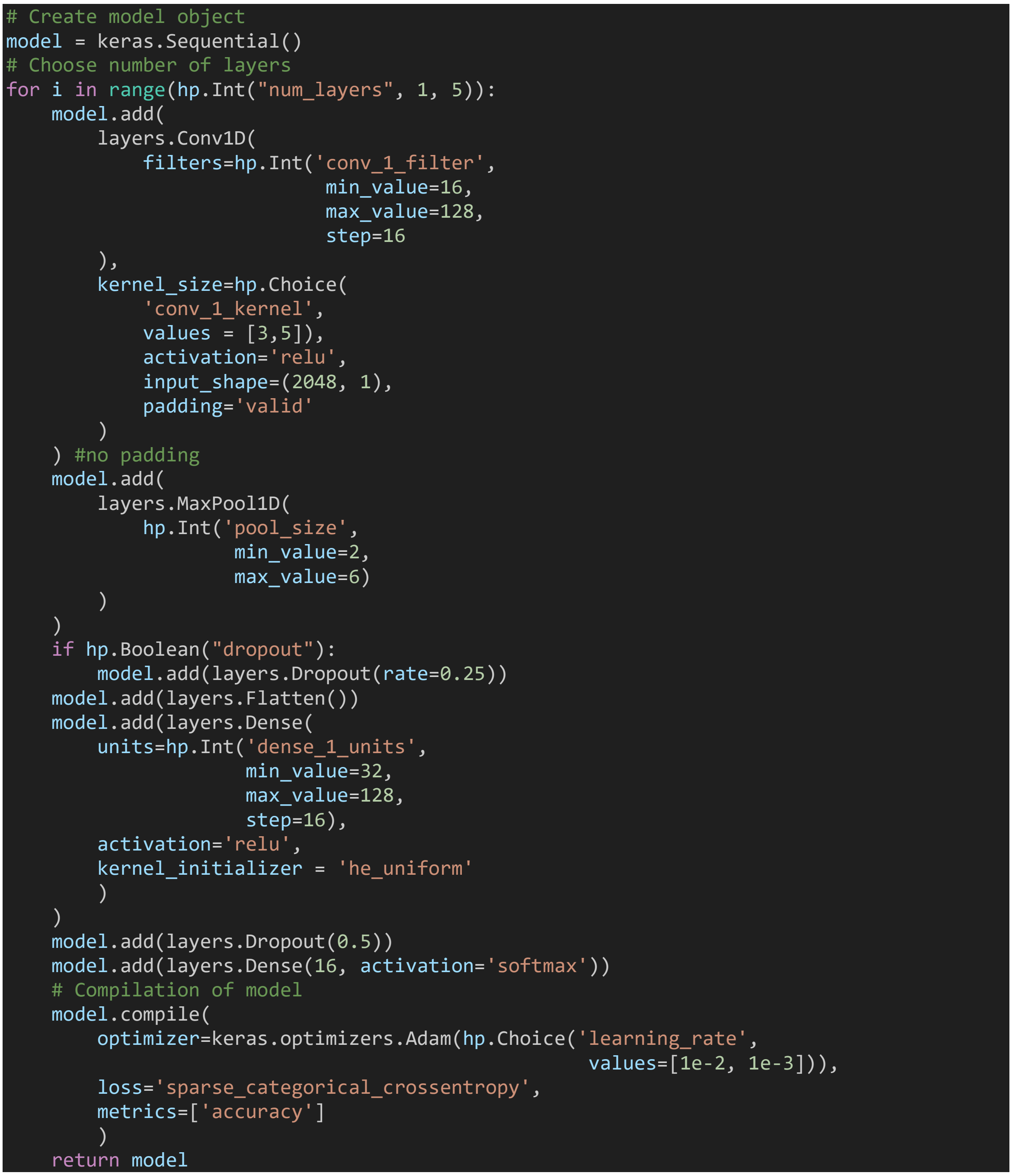
'fit_selection_model':

This function leverages the dictionary generated by 'build_array_dic' to fit and evaluate the CNN model for each tuple within the dictionary. For each tuple, both the input data and the respective model available are utilized. The accuracy values obtained for each fingerprint during evaluation are stored in a dictionary.

Building the definitive model

1. Hiperparameter tuning.

This programming segment involves selecting the fingerprints that yielded the highest accuracy using the previous model and using them to fit a refined convolutional model. To obtain this refinement, the package ‘keras_tuner’ was employed. The ‘RandomSearch’ class within this package allows the selection of a range of hyperparameters to test for optimal performance. The best combination of hyperparameters, including layer count, depth, application of dropouts, etc., is automatically chosen based on the performance. The performance indicator can be selected in Kedro's catalog between accuracy or loss on either the training or validation data. The maximum number of tuning iterations can also be specified.



```
# Create model object
model = keras.Sequential()
# Choose number of layers
for i in range(hp.Int("num_layers", 1, 5)):
    model.add(
        layers.Conv1D(
            filters=hp.Int('conv_1_filter',
                            min_value=16,
                            max_value=128,
                            step=16
        ),
        kernel_size=hp.Choice(
            'conv_1_kernel',
            values = [3,5]),
            activation='relu',
            input_shape=(2048, 1),
            padding='valid'
        )
    ) #no padding
    model.add(
        layers.MaxPool1D(
            hp.Int('pool_size',
                    min_value=2,
                    max_value=6)
        )
    )
    if hp.Boolean("dropout"):
        model.add(layers.Dropout(rate=0.25))
    model.add(layers.Flatten())
    model.add(layers.Dense(
        units=hp.Int('dense_1_units',
                     min_value=32,
                     max_value=128,
                     step=16),
        activation='relu',
        kernel_initializer = 'he_uniform'
        )
    )
    model.add(layers.Dropout(0.5))
    model.add(layers.Dense(16, activation='softmax'))
    # Compilation of model
    model.compile(
        optimizer=keras.optimizers.Adam(hp.Choice('learning_rate',
                                                   values=[1e-2, 1e-3])),
        loss='sparse_categorical_crossentropy',
        metrics=['accuracy']
        )
    return model
```

2. Model training

After the tuning of the hyperparameters, the model obtained was trained to optimize the weights of the convolutional net. The training was set to perform 200 epochs, while monitoring the validation loss, so it stops early in case signs of overfitting start to show (i.e., we see an increase in the validation loss, while the training loss decreases). The training saves a series of checkpoints in a folder called "temp". These checkpoints may be useful if the training of the model spikes the validation loss in the last epochs before the early stop. They file name of these checkpoints is the validation loss value, so a model with a lower loss can be easily chosen.

3. Getting the predictions

Following hyperparameter tuning, the obtained model was trained to optimize the weights of the convolutional net. The training was configured for 200 epochs, while closely monitoring the validation loss. If signs of overfitting emerged, such as an increase in validation loss while training loss continued to decrease, the training was stopped early.

To evaluate the model's performance on the test dataset, the 'predict' function from the ‘Sequential’ class was utilized. This function returned arrays of probabilities, making a direct comparison with the test dataset labels not feasible. In these arrays, the index value corresponded to the label value. To determine the predicted label, the NumPy function 'argmax' was used to return the index value containing the highest probability in each array.

Subsequently, the list of predicted labels was compared to the actual labels using the 'classification_report' function from ‘sklearn.metrics’. This allowed for the computation of accuracies for each label as well as the overall accuracy for the model.

Graphical representation of the Kedro app architecture

The interconnections between each of the coding blocks discussed earlier can be easily observed in the graphic shown in **Fig.3**, easily accessible through Kedro's 'viz' functionality.

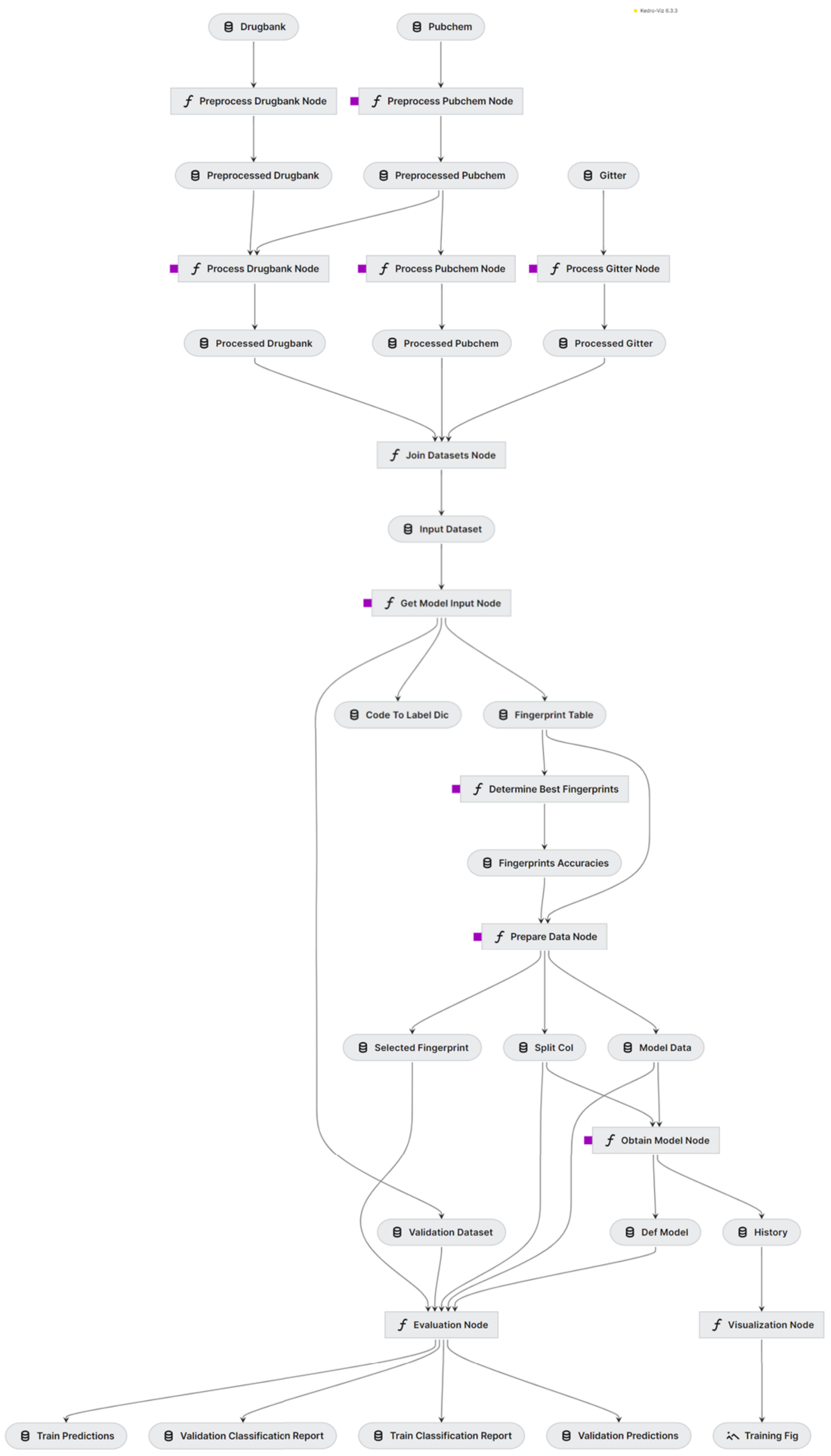


Fig.3. Drug Predictor's Kedro application's architecture.

Frontend applications

The model derived from the Kedro application serves as the foundation for the development of an applications capable of predicting the therapeutic indication of any given molecule. This application, named "Drug Predictor", provides the MATC classification for a molecule and indicates the associated probability for that category. Users have the flexibility to input the molecule in either the form of a SMILES representation or a CID number.

In addition, another application called "Visualization" allows to evaluate the performance of the model developed through the Kedro application.

Programming of Drug Predictor

Drug Predictor is a Python package comprising three modules. It can be graphicly displayed through Streamlit. The basic module organization is as follows:

Module 'compute_fp': comprises the class 'Compute_FP', that contains functions that calculate different types of fingerprints from an RDKit molecule object or CID number. It also contains the function 'relate_fp_functions' that organizes these functions in a dictionary with the name of the fingerprints as keys. This function takes as input the name of the fingerprints with which the CNN model has been trained and the molecule chosen by the user, then returns the correct fingerprints for that molecule.

Module 'calculations': it contains two classes:

- Class 'Display': creates al the graphical interface for the application using Streamlit functions.
- Class 'Calcs': deals with all the process of transforming the user's input (either an integer or array of integers corresponding with CID numbers or a strings or array of strings representing SMILES) into the molecule's fingerprints and reshaping them to fit into the CNN model. This class also loads the model, makes the predictions for those fingerprints and displays them, also saving them if necessary.

Module 'drug_predictor': simply instantiates the previous classes and asks for the user's inputs in the correct order for all to run smoothly.

Programming of the Visualization application

The Visualization application is structured as a Python package comprising three modules, utilizing Streamlit for its user interface. Here's a concise overview of these modules:

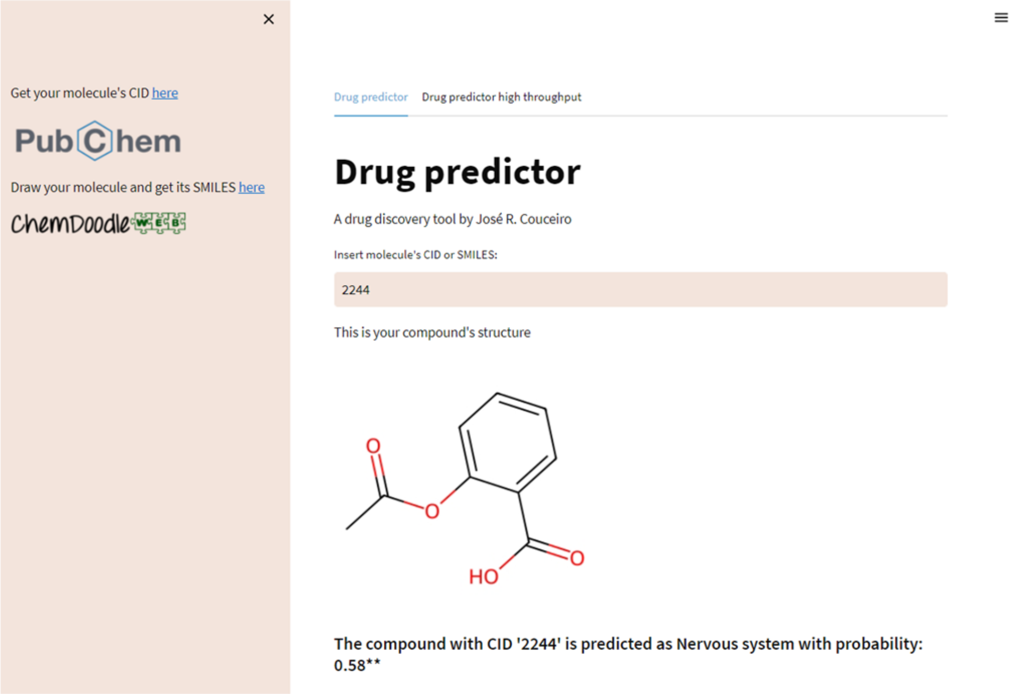
1. **Module 'draw_graphs':** This module houses a single class, 'Graphs,' which assembles two Matplotlib figures: the confusion matrix and the graphical representation of the classification report. It requires the predicted and true labels from Kedro's 'build_model' pipeline and the classification report from Kedro's 'evaluation' pipeline as inputs.
2. **Module 'motor':** This module is responsible for loading the necessary data (class 'Loads'), feeding it to appropriate functions and calculating the average accuracy score (class 'Organizer'), and orchestrating the display using Streamlit functions (class 'Display').
3. **Module 'visualization':** This module serves to initiate the application.

Interacting with the application interfaces:

Drug Predictor

The Drug Predictor interface is organized into two tabs and a sidebar. The sidebar offers links to chemical resources for retrieving molecule CID numbers (PubChem) and SMILES (ChemDoodle). Within the main screen, there are two tabs: 'Drug Predictor' and 'Drug Predictor High Throughput.'

1. In the 'Drug Predictor' tab, users can input individual CID numbers or SMILES strings into a search bar. The search yields the predicted classification of the molecule, along with the associated probability of this prediction. Additionally, a graphical representation of the molecule is presented (**Fig.4**).



Get your molecule's CID here

Draw your molecule and get its SMILES here

Drug predictor

A drug discovery tool by José R. Couceiro

Insert molecule's CID or SMILES:

2244

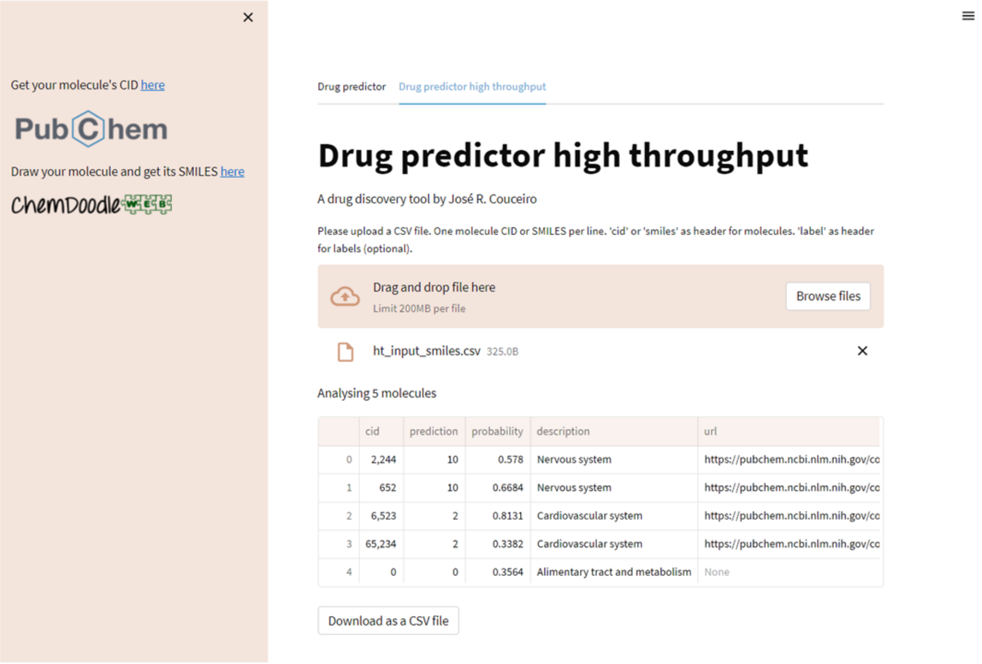
This is your compound's structure

The compound with CID '2244' is predicted as Nervous system with probability: 0.58**

Fig.4. Drug Predictor interface

2. The 'Drug Predictor High Throughput' tab enables users to input arrays of molecules for simultaneous analysis. These arrays can be provided in a CSV file, with one molecule per line and a header labeled either 'cid' or 'smiles' based on the type of data being presented. The column containing this information can be unique or among others; the application seamlessly processes the input as long as the headers are accurately labeled, and the CID and SMILES formats are correct.

In the 'Drug Predictor High Throughput' tab, the datasets that are inputted can either be labeled or unlabeled. In case they are labeled, the header of the target column should be identified as "label". The performance of the model can then be consulted in the Visualization app (**Fig.5**).



Get your molecule's CID here

Draw your molecule and get its SMILES here

Drug predictor high throughput

A drug discovery tool by José R. Couceiro

Please upload a CSV file. One molecule CID or SMILES per line. 'cid' or 'smiles' as header for molecules. 'label' as header for labels (optional).

ht_input_smiles.csv 325.0B

Analysing 5 molecules

	cid	prediction	probability	description	url
0	2,244	10	0.578	Nervous system	https://pubchem.ncbi.nlm.nih.gov/co
1	652	10	0.6684	Nervous system	https://pubchem.ncbi.nlm.nih.gov/co
2	6,523	2	0.8131	Cardiovascular system	https://pubchem.ncbi.nlm.nih.gov/co
3	65,234	2	0.3382	Cardiovascular system	https://pubchem.ncbi.nlm.nih.gov/co
4	0	0	0.3564	Alimentary tract and metabolism	None

Fig.5. Drug Predictor High Throughput Interface

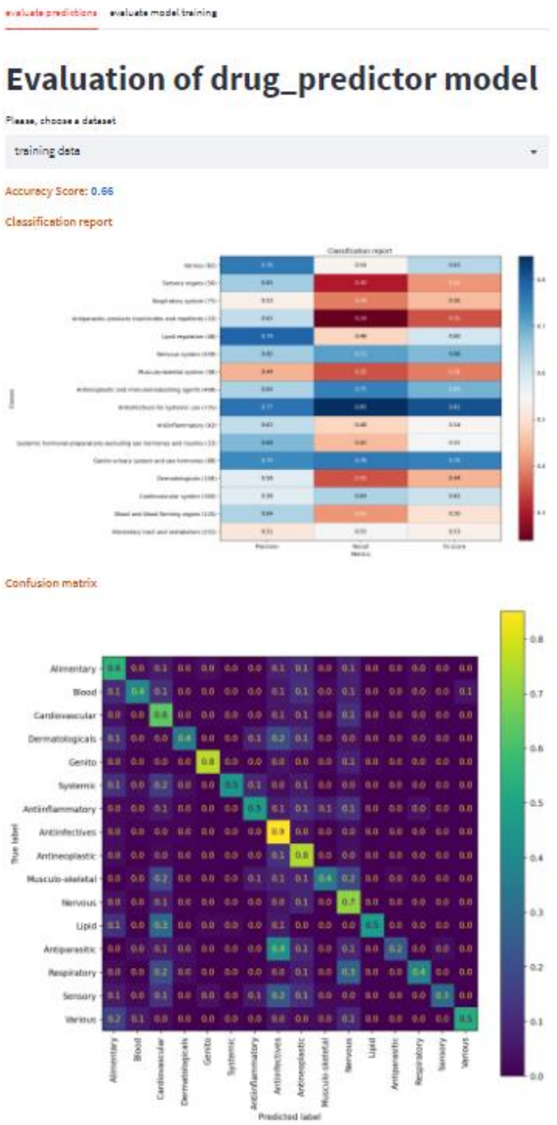
In the event that an incorrect CID or SMILES format is detected, the application will prompt an error message, urging the user to correct the format and resubmit the data for analysis. This ensures that the input conforms to the required format for accurate processing and analysis.

Visualization app

The Visualization app offers the ability to assess the model's performance against different datasets and visualize its training. The main screen includes a selection bar with three options: the training dataset, the validation dataset, and a dataset processed through Drug Predictor High Throughput. Upon selecting an option, the relevant information is displayed below the bar, including:

1. The global accuracy value.
2. A graphical representation of the classification report, enabling assessment of false negatives and positives for each label.
3. A graphical representation of the confusion matrix, facilitating the detection of biases in performance, such as a label being incorrectly assigned to a disproportionate number of molecules.

If there has been no search in Drug Predictor High Throughput and the option Drug Predictor High Throughput is selected in the selection bar, the app will display an error message, urging the user to perform a search.



Evaluation of drug_predictor model

Please, choose a dataset

training data

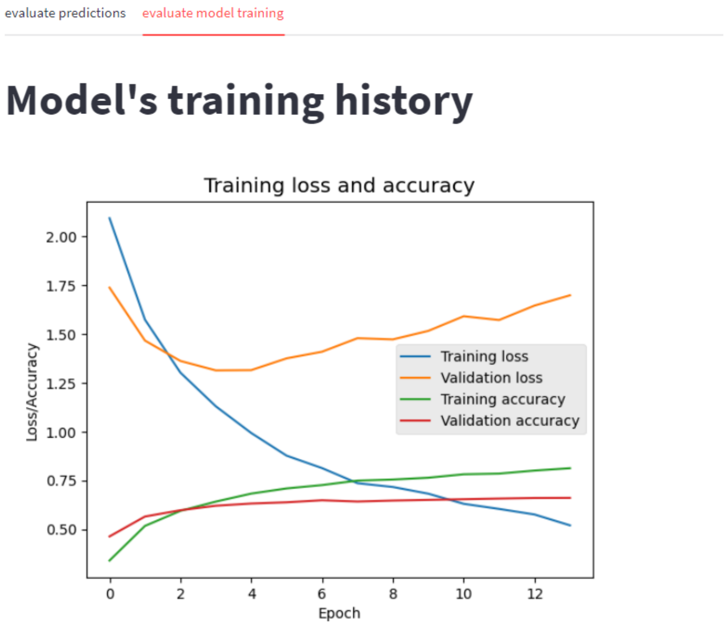
Accuracy Score: 0.66

Classification report

Confusion matrix

Fig.6. Visualization app interface ('evaluate predictions' tab)

Additionally, the 'evaluate model training' tab provides a graphical representation of the model training history, illustrating trends in loss and accuracy both in validation and training. This allows users to gain insights into the model's training process.



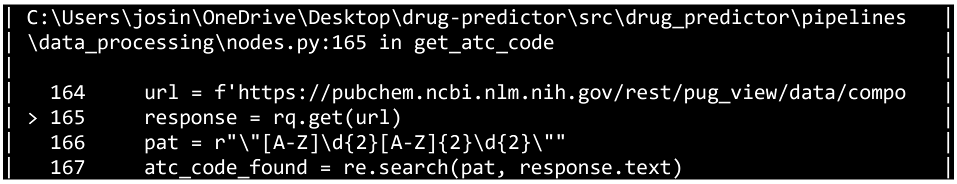
Model's training history

Fig.7. Visualization app’s interface (‘evaluate model training’ tab)

Known errors

PubChem related errors

Some errors have been reported to occur on occasions after running the Kedro application, particularly related to PubChem queries. These errors are associated with the use of the Python 'requests' package within the Kedro app for consultations to PubChem, primarily to obtain ATC codes. One common error that may occur is as follows:

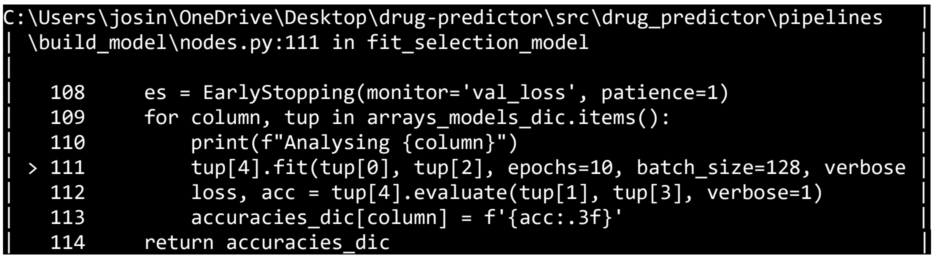


```
│ C:\Users\josin\OneDrive\Desktop\drug-predictor\src\drug_predictor\pipelines │
│ \data_processing\nodes.py:165 in get_atc_code                               │
│                                                                             │
│   164     url = f'https://pubchem.ncbi.nlm.nih.gov/rest/pug_view/data/compo │
│ > 165     response = rq.get(url)                                            │
│   166     pat = r"\"[A-Z]\d{2}[A-Z]{2}\d{2}\""                              │
│   167     atc_code_found = re.search(pat, response.text)                    │
```

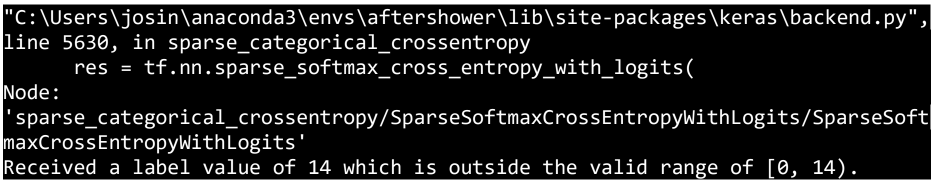
Currently, there is no definitive solution for the occasional errors related to PubChem queries, primarily due to the unavailability of ATC codes via API. Given that these errors are not consistent and only occur sporadically, the suggested approach is to relaunch the run.

Error related to the number of classes

On occasions, when launching the Kedro app with a small dataset, a discrepancy may arise regarding the number of labels that the model is set to receive and the real number of labels in the input dataset.



```
C:\Users\josin\OneDrive\Desktop\drug-predictor\src\drug_predictor\pipelines │
│ \build_model\nodes.py:111 in fit_selection_model                          │
│                                                                           │
│   108     es = EarlyStopping(monitor='val_loss', patience=1)              │
│   109     for column, tup in arrays_models_dic.items():                   │
│   110         print(f"Analysing {column}")                                │
│ > 111         tup[4].fit(tup[0], tup[2], epochs=10, batch_size=128, verbose │
│   112         loss, acc = tup[4].evaluate(tup[1], tup[3], verbose=1)      │
│   113         accuracies_dic[column] = f'{acc:.3f}'                       │
│   114     return accuracies_dic                                           │
```



```
"C:\Users\josin\anaconda3\envs\aftershower\lib\site-packages\keras\backend.py",
line 5630, in sparse_categorical_crossentropy
      res = tf.nn.sparse_softmax_cross_entropy_with_logits(
Node:
'sparse_categorical_crossentropy/SparseSoftmaxCrossEntropyWithLogits/SparseSoft
maxCrossEntropyWithLogits'
Received a label value of 14 which is outside the valid range of [0, 14).
```

A solution to address this issue is being developed. However, considering that Drug Predictor is intended for use with large datasets where the likelihood of encountering this error is minimal, it is not of significant concern. The recommended action in the event of this error is to relaunch the run. As dataset sizes increase, the occurrence of this error becomes more improbable.

CONCLUSIONS

Main conclusions

Following the training of the model with a 10,000-molecule database, an overall accuracy of 66% was achieved both on the training set and on a validation set of 200 molecules. This accuracy is inferior to that of previous reports that utilized a similar approach but with different prediction models, like random forests generators and 2D neural networks[28]. Moreover, when tested with new drugs that have not yet been marketed but have successfully completed phase 3 clinical assays, the accuracy dropped to 60%. The reasons for this low accuracy are outlined below.

Analysis of the confusion matrix revealed notable biases of the model towards certain labels, particularly Anti-infective, Antineoplastic, Cardiovascular, and Nervous System drugs. These labels were consistently favored by the model, leading to a decrease in recall for other categories. Conversely, categories such as Blood, Musculo-skeletal, Respiratory, and Dermatological exhibited significantly low recall rates, primarily due to the model's struggle in accurately categorizing them. This misclassification can be partly attributed to the ambiguous and inconsistent nature of the label codes being utilized. This discrepancy in categorization resulted in an overrepresentation of categories that signify the functionality of the molecule, such as Anti-infective or Antineoplastic, within the model's predictions. On the other hand, categories organized by organ of action, like Blood or Musculo-skeletal, are less frequently selected by the model. Interestingly, this indicates a positive aspect of the model's performance, as it has learned to associate function rather than just the system of application based on molecular structure. Nevertheless, this favorable performance is possibly due to the overrepresentation of functionality-based categories in the input dataset, as depicted in **Fig.8**.

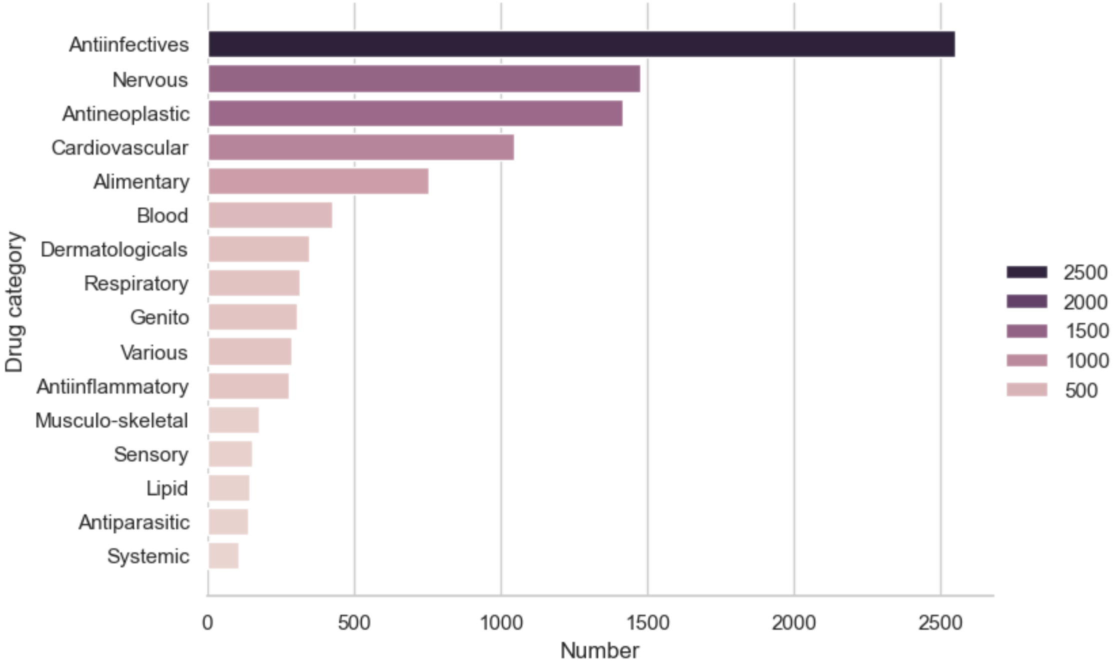


Fig.8. Distribution of classes in the 10.000-molecule dataset used to train the model.

[28] Meyer, J. G., Liu, S., Miller, I. J., Coon, J. J., & Gitter, A. (2019). Learning Drug Functions from Chemical Structures with Convolutional Neural Networks and Random Forests. Journal of Chemical Information and Modeling. https://doi.org/10.1021/acs.jcim.9b00236

When analyzing the confusion matrix, some anticipated confusions are observed, particularly between categories like Anti-infective and Antiparasitic. Conversely, contrary to initial expectations, the model demonstrated good discrimination between categories such as Anti-inflammatory and Nervous System (**Fig.9,** 16 labels model).

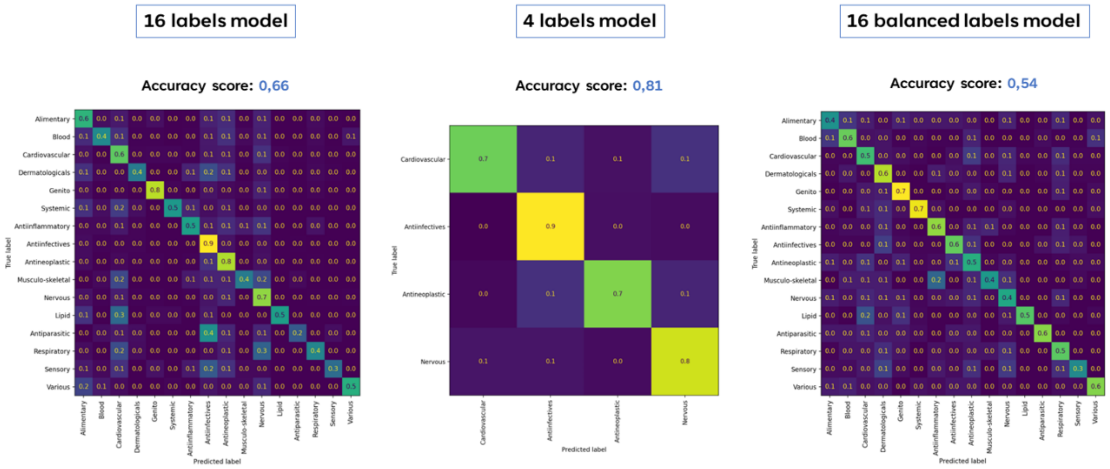


Fig.9. Accuracy score and confusion matrix for models with different number of labels

In summary, the model effectively predicts the medical indications of drugs, though it does not surpass the performance of other existing models. This outcome suggests potential advantages in utilizing 1D convolutional neural networks for drug discovery, especially in terms of simpler programming compared to image classification methodologies. However, given the inferior performance with respect to simpler machine learning models, there is a clear need to enhance the performance of this model to fully leverage its benefits. It's important to recognize that the use of convolutional neural networks does come with certain limitations, and further refinement is essential to optimize their utility in this domain.

Limitations of the model

Number of molecules and classes

CNN models typically perform optimally when trained on large datasets, often in the hundreds of thousands or millions, as seen in cases like ImageNet[29]. However, in this project, the challenge is the limited number of molecules available compared to the diverse range of drug classes. On one hand, increasing the number of molecules is difficult since the number of labeled drugs is fixed. On the other hand, reducing the number of labels would diminish the model's usefulness. Grouping ATC codes into broader categories could be considered, but this may not be feasible given the intricacies of how these categories are assigned. The labeling system using ATC codes poses challenges due to its complexity and lack of straightforward grouping that are discussed below.

To study the impact of the number of classes in the model's performance, the model was trained with a limited dataset consisting only of the molecules belonging to the 4 majority

[29] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., & Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision, 115(3), 211–252. https://doi.org/10.1007/s11263-015-0816-y

classes: Anti-infective, Nervous, Antineoplastic and Cardiovascular. This comprises around 6500 molecules out of the approximately 10000 that are present in the whole dataset. The result is an increase on accuracy from 0.66 to 0.81 and an overall improvement in recall (**Fig.9,** 4 label model). This result confirms that the overabundance of labels in the dataset are affecting greatly the model's performance.

Another factor with a potential detrimental effect on the model's performance is the unbalancing of the data labels in the dataset (**Fig.8**). To study this, a more balanced dataset was constructed reducing the size of the five more abundant classes to the median number of molecules per label in the dataset. This obviously came with the drawback of a lower number of total molecules in the dataset. The performance of the model trained with this dataset dropped to 0.54 with poor recall (**Fig.9,** 16 label balanced model), which reaffirms the hypothesis that the excessive number of labels in a dataset with a limited number of molecules is the cause of the poor performance of the model.

While limited training data is a challenge, transfer learning can be employed to adapt pretrained networks from a related problem with a larger dataset. This technique has shown success in various domains, allowing the model to leverage knowledge gained from a larger dataset and apply it to the drug prediction task, even with a limited dataset[30] [31]. For example, transfer learning has been effectively utilized in tasks such as classifying medical ultrasound images with fewer than 6,000 samples[32], oceanfront image classification with just 2,000 images[33], and even cellular image classification with less than 1,000 images[34]. What's notable is that these pretrained networks were originally trained on entirely different objects (e.g., classifying everyday objects), showcasing the wide range of applications and domains where transfer learning can be leveraged to address data limitations and boost model performance.

Considering this, an effective strategy to enhance Drug Predictor's model could involve training on an extensive dataset to predict Log P (partition coefficient) of molecules. This process involves identifying areas of high and low polarity within the molecule[35] [36]. The knowledge gained from this training can be employed to refine the detection of drugability. This approach could be extended to other physicochemical properties as the number of H bond acceptors and donors.

[30] Brown, A. L., & Kane, M. J. (1988). Preschool children can learn to transfer: Learning to learn and learning from example. Cognitive Psychology, 20(4), 493–523. https://doi.org/10.1016/0010-0285(88)90014-X

[31] Ching, T., Himmelstein, D. S., Beaulieu-Jones, B. K., Kalinin, A. A., Do, B. T., Way, G. P., Ferrero, E., Agapow, P.-M., Zietz, M., Hoffman, M. M., Xie, W., Rosen, G. L., Lengerich, B. J., Israeli, J., Lanchantin, J., Woloszynek, S., Carpenter, A. E., Shrikumar, A., Xu, J., … Greene, C. S. (2018). Opportunities and obstacles for deep learning in biology and medicine. Journal of The Royal Society Interface, 15(141), 20170387. https://doi.org/10.1098/rsif.2017.0387

[32] Cheng, P. M., & Malhi, H. S. (2017). Transfer Learning with Convolutional Neural Networks for Classification of Abdominal Ultrasound Images. Journal of Digital Imaging, 30(2), 234–243. https://doi.org/10.1007/s10278-016-9929-2

[33] Lima, E., Sun, X., Dong, J., Wang, H., Yang, Y., & Liu, L. (2017). Learning and Transferring Convolutional Neural Network Knowledge to Ocean Front Recognition. IEEE Geoscience and Remote Sensing Letters, 14(3), 354–358. https://doi.org/10.1109/LGRS.2016.2643000

[34] Nguyen, L. D., Lin, D., Lin, Z., & Cao, J. (2018). Deep CNNs for microscopic image classification by exploiting transfer learning and feature concatenation. 2018 IEEE International Symposium on Circuits and Systems (ISCAS), 1–5. https://doi.org/10.1109/ISCAS.2018.8351550

[35] Chen, Q., Zhang, Y., Gao, P., & Zhang, J. (2023). An interpretable graph representation learning model for accurate predictions of drugs aqueous solubility. Artificial Intelligence Chemistry, 1(2), 100010. https://doi.org/10.1016/j.aichem.2023.100010

[36] Gao, P., Liu, Z., Tan, Y., Zhang, J., Xu, L., Wang, Y., & Jeong, S. Y. (2022). Accurate predictions of drugs aqueous solubility via deep learning tools. Journal of Molecular Structure, 1249, 131562. https://doi.org/10.1016/j.molstruc.2021.131562

Labeling problems

In this study, the ATC label was selected as the classification model's label due to its widespread use and universal acceptance. However, it presents several challenges when employed for this purpose:

Ambiguity

The ATC labeling system introduces ambiguity as molecules can have multiple labels. Although our model only considered one ATC code for each molecule, it's not uncommon for a molecule to possess several ATC codes. This is especially true for established medicines with a long history in the market, increasing the likelihood of repurposing. For instance, aspirin falls into the N (Nervous system) group due to its analgesic effects, the A (Alimentary tract and metabolism) group because it is used as an anesthetic in oral surgery, and the B (Blood and blood-forming organs) group for its anticoagulating properties. Another illustrative example of a repurposed drug is sildenafil, famously marketed by Pfizer as Viagra. While widely recognized for its use in treating erectile dysfunction, sildenafil also finds application in the treatment of pulmonary arterial hypertension. Intriguingly, despite the shared mechanism of action in both treatments (specifically, the inhibition of phosphodiesterase 5 in circulatory system cells) the ATC code system classifies sildenafil under G (Genito-urinary system and sex hormones). However, it could just as reasonably be categorized as B, considering its effects on the circulatory system. This highlights how the current classification system may not always align with the actual mechanism of action of a drug.

Incoherence

While most ATC groups classified drugs based on the body system they act upon, antineoplastic drugs do not follow this pattern. This inconsistency can introduce ambiguity in the prediction model, since many antineoplastic drugs target specifically cellular markers only present in the tissues affected by the tumor which they are effective against. This incongruity can potentially cause confusion in classification models, blurring the distinction between antineoplastic drugs and those targeting specific organs.

Moreover, the labeling of drugs in a model aiming to decipher their function should ideally be based on functionality rather than anatomy. Coming back to the aspirin example, its function as an anesthetic is classified as a drug affecting the alimentary tract due to the established medical consensus on this particular use. Drugs that act as antagonist of neuronal muscarinic receptors are also perfect examples of this inconsistency. These drugs share a similar inhibitory action on muscarinic receptors but still they are classified in various ATC categories (**Table 3**).

CID	Name	Selectivity	ATC abbr.
5572	Trihexyphenidyl	M1	Nervous
3042	Dicyclomine	M1	Alimentary
4848	Pirenzepine	M1	Alimentary
154417	Hyoscyamine	M2	Alimentary
4942	Propiverine	M2	Genito-Urinary
444031	Darifenacin	M3	Genito-Urinary
6918558	Fesoterodine	M3	Genito-Urinary
11693	Glycopyrrolate	M3	Respiratory
24199	Hexocyclium	M3	Alimentary
154059	Solifenacin	M3	Genito-Urinary
5593	Tropicamide	M4	Sensory
4634	Oxybutynin	M5	Genito-Urinary
443879	Tolterodine	Nonselective	Genito-Urinary
3000322	Scopolamine	Nonselective	Alimentary
174174	Atropine	Nonselective	Alimentary
10429215	Homatropine	Nonselective	Alimentary
9819382	Desfesoterodine	Nonselective	Genito-Urinary
2905	Cyclopentolate	Nonselective	Sensory
657308	Ipratropium	Nonselective	Respiratory
5284631	Trospium	Nonselective	Genito-Urinary

Table 3. Some drugs targeting muscarinic receptors and their ATC classification.

A functional classification based on mechanism of action would provide a more coherent representation, emphasizing the drug's mechanism rather than the administration route or the tissue of application. Ultimately, the mechanism of action should be the primary consideration in relating a drug's structure to its classification. After all, it is the mechanism of action what relates to the molecular structure, not the tissue of application or the administration route. The ATC code, therefore, with all its usefulness, does not seem the more appropriate labeling system for the training of a machine learning algorithm.

Classification failures and repurposing

There are two possible explanations for the mislabeling of drugs by a machine learning model: (1) either the model lacks adequate knowledge to precisely predict the correct drug class, or (2) the model has gained new perspectives on the drugs and their respective classifications. Although the latter scenario is more intriguing, the former is the more cautious and likely interpretation. Nevertheless, situations where the model makes mistakes in its predictions can present valuable opportunities for drug repurposing. Besides, these inaccuracies might provide insights into drug mechanisms.

Limitations of the fingerprints

While a fingerprint can describe the three-dimensional structure of a molecule, certain areas of the fingerprint might be predominantly filled with zeros due to variations in molecular sizes. Complementarily, certain parts of a molecule may be irrelevant to its functionality, as functionality often relies on specific functional groups or specific contact regions within the molecule. It is for this reason, that it could be a valuable approach to employ principal component analysis or other method of feature selection to reduce irrelevant data within the fingerprint.

From a different perspective, there's ongoing research exploring the substitution of fingerprints with alternative mathematical representations of molecules. A recent advancement in this field is the introduction of a novel molecular representation called topological distance-based electron interaction (TDEi) tensor[37]. This representation draws inspiration from the quantum mechanical model of a molecule, defining a molecule in terms of electrons and protons. An advantage of the TDEi tensor is its 3D array shape, allowing for the straightforward application of CNN architectures developed in computer vision to analyze the TDEi tensor.

[37] Shin, H. K. (2021). Topological Distance-Based Electron Interaction Tensor to Apply a Convolutional Neural Network on Drug-like Compounds. ACS Omega, 6(51), 35757–35768. https://doi.org/10.1021/acsomega.1c05693