In Silico Prediction Tool for Drug-likeness of Compounds based on Ligand Based Screening

Ani R$^1$, Anand P S$^1$, Sreenath B$^1$, Deepa O S$^2$

$^1$Department of Computer Science and Applications, Amrita Vishwa Vidyapeetham, Amritapuri, Kollam-690525, India
$^2$Department of Mathematics, Amrita Vishwa Vidyapeetham, Coimbatore-641112, Tamil Nadu, India

**ABSTRACT**

Drug Likeness prediction is a time-consuming and tedious process. An in-vitro method the drug development takes a long time to come to market. The failure rate is also another one to think about in this method. There are many in-silico methods currently available and developing to help the drug discovery and development process. Many online tools are available for predicting and classifying a drug after analyzing the drug-likeness properties of compounds. But most tools have their advantages and disadvantages. In this study, a tool is developed to predict the drug-likeness of compounds given as input to this software. This may help the chemists in analyzing a compound before actually preparing a compound for the drug discovery process. The tool includes both descriptor-based calculation and fingerprint-based calculation of the particular compounds. The descriptor-calculation also includes a set of rules and filters like Lipinski’s rule, Ghose filter, Veber filter and BBB likeness. The previous studies proved that the fingerprint-based prediction is more accurate than descriptor-based prediction. So, in the current study, the drug-likeness prediction tool incorporated the molecular descriptors and fingerprint-based calculations based on five different fingerprint types. The current study incorporated five different machine learning algorithms for prediction of drug-likeness and selected the algorithm, which has a high accuracy rate. When a chemist inputs a particular compound in SMILES format, the drug-likeness prediction tool predicts whether the given candidate compound is drug or non-drug.

**INTRODUCTION**

Nowadays we can find out different types of software and tools for predicting the drug-likeness on the web. But the problem with these tools are they are typically based on one method, even if there are other methods are available for drug-likeness prediction. Generally, they focus on only descriptor-based calculation or fingerprint-based calculation or docking. In our further studies, we found out the importance of a tool which incorporate all these methodologies. If we create such a tool, the accuracy rate is very high in the prediction of drug-likeness, or we can derive much better conclusion for the chemist who is eagerly waiting for the results. The fact is that there doesn’t exist any other tools that
incorporate all these methods. This increases the importance of our project. DruLiTo, Swiss ADME and Drug mint are the well-known drug-likeness prediction tools that are existing on the web into the problems of these tools:

DruLiTo is an open-source virtual screening tool for the drug-likeness prediction. The problem with this tool is, the tool gets freezes when processing some compounds. DruLiTo only contains the rules and filters for descriptor-based prediction. Another tool called SwissADME has the same problem which is the tool only focusing on descriptor-based calculation. Various studies and papers prove the accuracy of fingerprint-based prediction. So, for more accurate values, we need to combine these two methodologies. Drug Mint is a web server, which integrates these two methodologies, but it has a limited number of fingerprint types.

**Table 1: Lipinski Rule**

| Lipinski Rule       |          |
|---------------------|----------|
| logP                | <= 5     |
| Molecular Weight    | <= 500   |
| H-Bond Donor        | <= 5     |
| H-Bond Acceptor     | <= 10    |

The fact is that there doesn’t exist any other tools that incorporate all these methods effectively. This increases the importance of our Drug Likeness Prediction Tool. The tool incorporates different filters and rules for descriptor prediction. Because by using any single filter or rule, we can’t be able to make a better prediction. For example, when we use Lipinski’s rule alone, the prediction of the drug-likeness is not that much accurate. When using only this rule for the prediction, we got 50% — 80% wrong information in the output. By combining these filters and tools, we can reduce the wrong information and derive much better results. When coming to the fingerprint-based prediction, we use five different types of fingerprints with five different algorithms. So, the tool can figure out which is the most accurate algorithm for the five fingerprints and present the best-predicted output to the chemist.

**QSAR**

QSAR is a technique that tries to predict the activity, reactivity, and properties of an unknown set of molecules such as the binding affinity or hypothetical molecules or the toxic potential of existing with the help of effective molecular descriptors. The idea behind the QSAR modeling is that the candidate molecules which have the same molecular features will trigger the same biological responses. This area of research is trying to establish a relation between structural and electronics characteristics among candidate molecules. We are well-known with the ADMET properties. This is the mandatory features of a drug compound. The ADMET includes adsorption, distribution, metabolism, excretion, and toxicity. Nowadays through QSAR modeling, we can also predict the candidate molecule is a drug or non-drug (Lo et al., 2018).

**Table 2: Ghose Filter**

| Ghose Filter         |          |
|----------------------|----------|
| Atom Count           | 20 to 70 |
| Molecular Weight     | 160 to 480 |
| Molecular Refractivity| 40 to 130 |
| logP                 | -0.4 to 5.6 |

The initial phases of the QSAR technique are focused on the odd features of a molecule. Because of that we can estimate the biological responses on a particular molecule. This initial approach is called as 1D-QSAR. The researchers in this field extended their studies by binding more than one molecule. It leads to the finding of the 2D-QSAR technique (Kwon et al., 2019).

The much better version of QSAR, known as 3D-QSAR technique maps the particular compound interactivity with the biological and chemical features into 3-D vector area. The main drawback is with 3-D QSAR is we are incapable to anticipate the exact location of the corresponding molecule. It is because the molecule is plotted into a 3-D vector space, that includes other replacement molecules. It is very hard to locate the target molecule without the structural information. By using alignment descriptors we can solve this problem (Eckert and Bajorath, 2007).

**Table 3: CMC-50 Likeness**

| CMC-50 Likeness |          |
|-----------------|----------|
| Atom Count      | 30 to 55 |
| Molecular Weight| 230 to 390 |
| Molecular Refractivity | 70 to 110 |
| logP            | 1.3 to 4.1 |

The 4-D QSAR technique is developed to address the alignment issue in the 3-D QSAR and 4-D QSAR is also an extension of 3-D QSAR technique. This technique solve alignment problem by constituting each molecules in non-identical conformation, orientation and protonation. Then the fundamental QSAR algorithm will locate the particular molecule.
Descriptors

We discussed how we screen the candidate molecules in the drug discovery process. Now we are going to discuss descriptors which identify or categorizes the given compound into drug or non-drug. QSAR (quantitative structure-activity relations) models started to play a major role in drug discovery because of the cost of the methods like high throughput screening. This model requires good molecular descriptors which provides information about the molecular features for the target candidate molecule. Molecular descriptors are different in the basis of its underlying algorithm uses for the calculation and the type of molecular representation. Geometrical descriptors, topological indices, physicochemical and constitutional descriptors are some of the well-known types of descriptors.

Table 4: Veber Rule

| Veber Rule |           |
|------------|-----------|
| Rotatable Bond | <= 10 |
| TPSA       | <= 140   |

Topological Descriptors

These descriptors are the 2D descriptors that mainly concerned with the internal alignment of molecular compounds. They considered as structure explicit descriptors because these descriptors are derived from the topological representation of compounds. In numerical form, it encodes the features like shape, presence of hetero atoms, multiple bonds, size and branching. It generally represents the connectivity of atoms or bonds. Because of including these kinds of features and properties it has a major role in biological activities, pharmacokinetic properties and physicochemical properties of the respective compounds. Numerical graph calculations are very necessary for the calculation of the topological descriptors because graphs contain a non-numeric form of compound molecular substructure. The commonly used topological descriptors are connectivity indices, Balaban j index, Zagreb indices, wiener index and kier shape. These descriptors categorize the compound molecules based on the shape, size, branching and flexibility.

Geometrical Descriptors

Geometric descriptors are calculated from the atoms 3D coordinate in a specific compound molecule. When comparing with topological descriptors, this kind of descriptors provides more information and discrimination when coming to the comparing compounds having the same structures. For geometric optimization a geometric overhead is needed to use in this kind of descriptors it will lead to exploiting the new pieces of information about flexible molecules having several molecular conformations. It increases complexity also. For this specific reason, these descriptors need alignment rules for comparing the candidate molecules.

Table 5: MDDR Likeness

| MDDR Likeness \( \geq 3 \) |           |
|---------------------------|-----------|
| No of Rings |           |
| No of Rigid Bonds | >= 18 |
| No of Rotatable Bonds | >= 6 |

Physicochemical Descriptors

From the 2D structure, we can analyze different physical and chemical properties. These properties influence drug activities in the body. The proper features of the drug might increase market demand. So, by evaluating the chemical and physical properties we can also assist the drug discovery process by identifying the compound selected. This part indicates that we need to pay attention in physicochemical properties like solubility, permeability that decides the optimal potency (Khan, 2016; Brüstle et al., 2002).

Descriptor Rules

There are several descriptor prediction rules available. For our Drug Likeness Prediction tool we took six descriptor rules. The conditions for a compound to satisfy Lipinski rule and Ghose filter are shown in Tables 1 and 2 respectively. Tables 3 and 4 shows the requirements of a compound to satisfy the CMC-50 rule and Veber rule respectively. The other two descriptor rules MDDR and BBB likeness conditions that should be satisfied by a compound are given in Tables 5 and 6.

Table 6: BBB Likeness

| BBB Likeness |           |
|--------------|-----------|
| Molecular Weight | <= 400 |
| H-Bonds | <= 8 |
| No of Acids | 0 |

Fingerprints

So, we discussed the descriptors and their calculations that are used for the descriptor-based calcula-
tion. Now we are going to have an in-depth look into the fingerprint prediction part of our tool.

The most common problem when trying to measure the similarity between two molecules is the complexity in its molecular representation. To measure the similarity between two molecules in a computationally more straightforward manner, we have to minimize the complexity in representing molecules. The commonly used easier and simple representation of molecules is molecular fingerprints. The molecular fingerprints convert the molecule into a sequence of bits; by this simple representation, we can easily compare the similarity between two molecules.

### Table 7: Data Sources

| Database          | Type     |
|-------------------|----------|
| ChemBL            | Public   |
| DrugBank          | Public   |
| NCI Set           | Public   |
| PubChem           | Public   |
| ZINC              | Public   |
| ChemSpider        | Public   |
| Superdrug         | Public   |
| CoCoCo            | Public   |
| TCM               | Public   |
| ACD               | Commercial |
| Life Chemicals    | Commercial |
| IBS Database      | Commercial |
| Chembase          | Commercial |

### Substructure Key Based Fingerprints

As the name denotes sub-structure key-based fingerprints make the bit string according to the presence of the keys. The fingerprint checks the substructure keys with the candidate compounds and generates the bit string depends on the presence of corresponding features. It is helpful when the list of structural keys is given but not in the case of absence of the structural keys. MACCS is considered as one of the commonly used fingerprint types. It is small in length. It is commonly used because it includes most of the features for drug discovery. It is available in both 166-bit and 960-bit format. It includes structural keys, which are used in the SMARTS pattern (Reker and Schneider, 2015; Cereto-Massagué et al., 2015).

### Topological Fingerprints

They evaluate all the components of the molecules that merge in each stage of bond creation. It continues till a certain number of bonds and then applying a hash value to every analyzed component for generating the fingerprints. Due to its working manner, we can convert any candidate molecule into a fingerprint. With the help of the hashing mechanism, we can adjust the length of the bit string. These features enabled topological fingerprints faster in substructure searching and filtering of candidate molecules. One disadvantage of this fingerprint type is a single bit cannot be tracked back to a specific feature. This may lead to bit collision. Atom pairs and Topological torsions are two types of fingerprints that come under topological fingerprints. AtomPairs contains 2 versions of hashed and normal ones. Hashed fingerprint have 2048 bits, and ordinary has 16000+bits. Topological Torsion consist of four different types of atoms are included in each fragment of the bit string conversion range based on the path (Lavecchia, 2015; Willett, 2006).

### Circular Fingerprints

The extension of topological fingerprints is circular fingerprints. Instead of looking for the components and path up to individual bonds, this fingerprint type evaluates the environment of the particular atom with a given radius circle. For this reason, this is not meant for substructure searching or substructure queries. This type of fingerprint is mainly used for structure similarity searching. ECFP and FCFP are well-known types in circular based fingerprints. ECFP (Extended Connectivity Fingerprint) is mainly based on the Morgan algorithm. There are two types of ECFPs. One is ECFP4 and ECFP6. The main difference is the difference in the radius. One uses four as the radius while the other uses six as the range. FCFP (Functional Class Fingerprints) have a variation from ECFP. Because instead of indexing the environment of the particular atom, it’s indexing the atom’s role in the compound (Kumar and Zhang, 2015; Duan et al., 2010).

### Table 8: AtomPairs Analysis

| Algorithm    | Accuracy |
|--------------|----------|
| Naïve-Bayes  | 0.74     |
| KNN          | 0.83     |
| Random Forest| 0.88     |
| MLP          | 0.87     |
| CNN          | 0.87     |

### MATERIALS AND METHODS

#### Naive Bayes

Naive Bayes algorithm is a classification model mainly based on the well-known Bayes Theorem.
The Naive Bayes classifier also presumes the existence of a specific feature in class or category is independent to the existence of any other feature. Along with simplicity, this classification model is useful for size able data sets and easy to build. Because of these factors, Naive Bayes outperforms even highly advanced classification techniques.

Above, the posterior probability $P(c/x)$ can be found by dividing the product of the prior probability of category $P(c)$ and the likelihood which is the probability of predictor given category $P(x/c)$ by the prior probability of the predictor $P(x)$. The Naive Bayes algorithm works as follows: First of all, we need to convert the input data set into a frequency table. From the frequency table, the algorithm creates a likelihood table by finding out the probabilities. By substituting the probabilities into the Bayes equation, the algorithm can find out the posterior probability for the classes. The outcome of the prediction is the category which has a higher posterior probability.

| Table 9: MACCS-166 Analysis |
|-----------------------------|
| **Algorithm** | **Accuracy** |
| Naïve-Bayes | 0.74 |
| KNN | 0.90 |
| Random Forest | 0.91 |
| MLP | 0.90 |
| CNN | 0.89 |

Advantages of the Naive Bayes classifier are: When comparing to other classification methods like logistic regression, Naive Bayes is fast and easy to implement. Naive Bayes uses less training data, and also the algorithm is expandable in nature or the algorithm linearly expandable with the count of data points and predictors. The Naive Bayes classifier is efficient to deal with the discrete and continuous data, and also the technique can make probabilistic predictions more accurately. This well-known classification algorithm is also used for the multi class and binary classification problems.

| Table 10: RDK Fingerprint Analysis |
|-----------------------------------|
| **Algorithm** | **Accuracy** |
| Naïve-Bayes | 0.79 |
| KNN | 0.86 |
| Random Forest | 0.90 |
| MLP | 0.90 |
| CNN | 0.89 |

Disadvantages of the Naive Bayes classifier are: The Naive Bayes classifier is enormously depending on its feature independence, and this is the most significant disadvantage of the classifier too. Because it is tough to have a group of features which are entirely independent of each other in the real-life scenarios, if a categorical variable has a class and not being noticed in the training data then the Naive Bayes classifier will set a zero probability to the class. It will be impotent to make a prediction. This problem is generally known as “Zero frequency” in the Naive Bayes classification (Sun, 2006; Vijayarani and Muthulakshmi, 2013).

K Nearest Neighbours

K Nearest Neighbours (KNN) is a simple, supervised ML algorithm which is generally suggested for the classification as well as the regression problems. But the industries are mainly depending on this algorithm in the classification predictive problems. KNN is defined as an idle algorithm because the algorithm does not have a dedicated training part and uses all the input data for training in the time of classification. The algorithm does not assume anything about the underlying data. Because of this, the algorithm is also known as a non-parametric algorithm.

Working of a KNN algorithm: KNN algorithm predicts the new data point values based on the feature similarity, which further means that the values of the new data points are how similarly matches the points in the training data set. The working of the algorithm is as follows: We must feed the training as well as test data for implementing the KNN algorithm. Then, we need to assign the value of $K$, i.e. the nearest data points. $K$ can be any integer.

| Table 11: Morgan-Circular Analysis |
|-----------------------------------|
| **Algorithm** | **Accuracy** |
| Naïve-Bayes | 0.84 |
| KNN | 0.82 |
| Random Forest | 0.91 |
| MLP | 0.87 |
| CNN | 0.88 |

For each data point in the test data, the algorithm does the following: With the help of the Euclidean method, the algorithm calculates the distance between training data and test data. Sort them in ascending manner based on the Euclidean distance. Next, the algorithm will select the top $K$ rows from the sorted array after that KNN provides a category to the test points based on the most recurrent class of these rows.

Advantages of the KNN algorithm are: Along with the simplicity, the KNN algorithm is beneficial for non-linear data classification because the algorithm
doesn't assume anything about data. We can use the algorithm for regression as well as classification problems with a result having high precision.

Disadvantages of the KNN algorithm are: The prediction of KNN is slow in the case of big N. It is very sensitive to the irrelevant features. As we mentioned earlier the KNN algorithm stores all the training data. Because of this the algorithm is computationally a bit expensive and requires high memory storage when comparing to other models. (Jia et al., 2020; Shen et al., 2003).

**Random Forest**

The Random forest algorithm is an ensemble learning method which is generally suggested for classification and regression problems. We already know that a forest is made up of a large number of trees and the number of trees increases the robustness of the forest. Likewise, the algorithm generates decision trees based on data samples and then predicts form each of the decision trees.

| Algorithm    | Accuracy |
|--------------|----------|
| Naïve-Bayes  | 0.80     |
| KNN          | 0.80     |
| Random Forest| 0.89     |
| MLP          | 0.88     |
| CNN          | 0.87     |

Advantages of Random forest algorithm are: Complexity is considered as one of the most significant disadvantages of the random forests. When comparing to the regular decision trees, the creation of the random forest is time-consuming and required more computational resources. In comparison, with other algorithms, the random forest prediction process is very time-consuming, and for an extensive collection of decision trees, the prediction is less intuitive (Ani et al., 2018, 2016).

**Multi-layer perceptron**

The Multi-layer perceptron or MLP is a type of algorithm in feed forward artificial neural networks. MLP has a single hidden layer sometimes known as a vanilla artificial neural network. MLP contains at least three layers named as the input layer, hidden layer and an output layer. The multi-layer perceptron considers every node as a neuron with an activation function, except for the input nodes. For training purposes, MLP uses the algorithm called back propagation. The noticeable differences from linear perceptron are the multi-layer perceptron consist of non-linear activation function and multiple layers. The MLP can also distinguish the data which is not linearly separable. How does a multi-layer perceptron work: A fully connected input layer and output layer are the critical components of the multi-layer perceptron. Multiple hidden layers are there in between the layers as mentioned above of an MLP. The multi-layer perceptron includes fully connected input and output layers as same as the perceptrons. The difference is MLP contains multiple hidden layers in between input and output layers (Yosipof et al., 2018).

Let's look into the algorithm of multi-layer perceptron: The inputs are forwarded through the multi-layer perceptron by considering the value of the dot product of assigned weights with inputs. What lies in between the input layer and the hidden layer. It's almost the same with working of the perceptron. In the hidden layer, the algorithm assigns a value to the
dot product, but we don’t push forward this value. We already mentioned that the MLP uses the activation functions at each of their layers except the input layers. Sigmoid functions and Rectified linear function are examples of such activation functions. Through this activation functions, MLPs pushes the calculated output at the presentation layer. So, in the next step, the algorithm forwards the computed result at the hidden layer through the activation function and pass it to the next layer in multi-layer perceptron. It also carries forward the dot product of corresponding weights. The algorithm will redo the steps to the final output layer. The result will be used for decision-making.

Advantages of the MLP: As you can understand, the multi-layer perceptron builds the basis for all the neural networks. And the multi-layer perceptron increases the computational power when dealing with the regression and classification problems. By multi-layer perceptron, we can also understand complex, and large models and also the computers are free from the xor problems (Radhika et al., 2019; Kavitha et al., 2017).

Convolutional Neural Networks

Convolutional Neural Networks or CNN is a kind of neural networks, which is generally used for image or visual data identification and classifications. The algorithm is efficiently used for areas like commercial recommended applications and natural language processing. There are different types of CNN’s like 1D, 2D and 3D. Regardless of its different types, the algorithm is having the same traits and working based on the same approach. The difference between CNN’s lies in the type of input data. For example, 1D CNN is used for natural language processing and 2D for image classification. The inputs to these CNN’s are generally a sentence and pixels in image respectively. Working of the filters is also distinguishing the CNN into these three types. When coming to the architecture of the convolutional neural networks, shared weights, and sparse connectivity is two essential features. When comparing to multi-layer perceptron, CNN has a different architecture or connectivity between the neurons. In the connectivity pattern, we can set the width of the dataset. Assume that the width is three, then the neurons in the m layer will connect to three neurons in the below layer and it will go on till the last layer. The neurons in the same layer will share common boundaries, and we can categorize them. This category which shares the common boundaries are called filters. In the m-1 layer, we have five neurons and three different filters. Through these filters, we can generate better results to an input dataset. We use a non-linear function with the filters. For our tool, we have applied the sigmoid function for the non-linearities. The most significant advantage of this architectural pattern is we can find out a connection between neurons in the neighbouring layers (Chen et al., 2018).

Shared weights are another feature of the CNN which means that the same parameters are shared among each filter. We already mentioned about m and m-1 layers. From the figure, it is clear that each neuron in the m-1 layer shares standard weights with the neurons in the above layer m. By using filters like, this will undoubtedly help in the identification of features nevertheless of their location in the input dataset. Shared weights also reduce the number of parameters. It will increase the efficiency of the algorithm and its fast processing. CNN also uses ideas like max-pooling and average pooling to reduce the over-fitting of the input data set (Wallach et al., 2015; Li et al., 2017).

Database

Dataset sources from which data are taken for the training and testing purposes are given in Table 7.

Dataset

Training and test data set contains a total of 11449 compounds out of which 6449 drug compounds were taken from Chembl database, and 5000 non-drug compounds were taken from the NCI database (Roy and Kadam, 2007).

Tool Operation

The tool front-end is built on anvil platform, and the back-end works on Jupiter notebook based on python 3. The anvil app is a cloud-based web-app development platform for the easy creation of a web app. The tool takes input in SMILES format and options are given to the user for drug-likeness prediction based on two methods, i.e. De- script Prediction and Fingerprint based prediction. The descriptor prediction has six descriptor rules used to predict. It also allows the user to choose which type
of fingerprint to be used in the fingerprint prediction technique. The tool allows five fingerprint techniques to be used. They include Maccs166,RDK, AtomPairs, Topological Torsion, Morgan Fingerprints. The tool uses the random forest algorithm for the prediction as analyzed earlier. The UI of our tool would look like as in Figure 1.

The output of descriptor shows pass or fail results and fingerprint output shows whether the compound can become a drug or not. The images of a tool predicting drug and non-drug are shown in Figures 2 and 3.

RESULTS AND DISCUSSION

The result set contains the accuracy of various fingerprints on five different algorithms. The analysis of basic MACCS-166 bit fingerprint is shown in Table 9. Here we have taken RDK Fingerprint given by the RD Kit Packages for analysis and results are given in Table 10. Tables 8 and 12 contains the result set of two topological fingerprints AtomPairs and Topological Torsion, respectively. The only circular fingerprint taken for analysis is Morgan Fingerprint, and the result set is shown in Table 11.

CONCLUSIONS

From the analysis, we came to know that the Random Forest algorithm gave better results with all the fingerprints used for prediction. It out performed other algorithms in similarity-based drug-likeness prediction. So we implemented a random forest algorithm into our tool for the drug-likeness prediction of five fingerprint types. Our tool also shows the result set based on descriptor prediction, which has a total of six rules. The tool incorporating both the fingerprint and descriptor prediction gives a better result compared to other available tools.

Funding Support

The authors declare that they have no funding support for this study.

Conflict of Interest

The authors declare that they have no conflict of interest for this study.

REFERENCES

Ani, R., Augustine, A., Akhil, N. C., Deepa, O. S. 2016. Random forest ensemble classifier to predict the coronary heart disease using risk factors. In Proceedings of the International Conference on Soft Computing Systems, pages 701–710. Springer.
Ani, R., Manohar, R., Anil, G., Deepa, O. S. 2018. Virtual Screening of Drug Likeness using Tree Based Ensemble Classifier. Biomedical and Pharmacology Journal, 11(3):1513–1519.
Brüstle, M., Beck, B., Schindler, T., King, W., Mitchell, T., Clark, T. 2002. Descriptors, Physical Properties, and Drug-Likeness. Journal of Medicinal Chemistry, 45(16):3345–3355.
Cereto-Massagué, A., Ojeda, M. J., Valls, C., Mulero, M., Garcia-Vallvé, S., Pujadas, G. 2015. Molecular fingerprint similarity search in virtual screening. Methods, 71:58–63.
Chen, H., Engkvist, O., Wang, Y., Olivecrona, M., Blaschke, T. 2018. The rise of deep learning in drug discovery.
Duan, J., Dixon, S. L., Lowrie, J. F., Sherman, W. 2010. Analysis and comparison of 2D fingerprints: Insights into database screening performance using eight fingerprint methods. Journal of Molecular Graphics and Modelling, 29(2):157–170.
Eckert, H., Bajorath, J. 2007. Molecular similarity analysis in virtual screening: foundations, limitations and novel approaches. Drug Discovery Today, 12(5-6):225–233.
Jia, C. Y., Li, J. Y., Hao, G. F., Yang, G. F. 2020. A drug-likeness toolbox facilitates ADMET study in drug discovery. Drug discovery today, 25:248–258.
Kavitha, K. R., Gopinath, A., Gopi, M. 2017. Applying improved svm classifier for leukemia cancer classification using FCBF. 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pages 61–66.
Khan, A. U. 2016. Descriptors and their selection methods in QSAR analysis: paradigm for drug design. Drug discovery today, 21(8):1291–1302.
Kumar, A., Zhang, K. Y. 2015. Hierarchical virtual screening approaches in small molecule drug discovery. Methods, 71:26–37.
Kwon, S., Bae, H., Jo, J., Yoon, S. 2019. Comprehensive ensemble in QSAR prediction for drug discovery.
BMC Bioinformatics, 20(1):521–521.

Lavecchia, A. 2015. Machine-learning approaches in drug discovery: methods and applications.

Li, J., He, P., Zhu, J., Lyu, M. R. 2017. Software defect prediction via convolutional neural network. 2017 IEEE International Conference on Software Quality, Reliability and Security (QRS), pages 318–328.

Lill, M. A. 2007. Multi-dimensional QSAR in drug discovery. Drug Discovery Today, 12(23-24):1013–1017.

Lo, Y.-C., Rensi, S. E., Torng, W., Altman, R. B. 2018. Machine learning in chemoinformatics and drug discovery.

Myint, K. Z., Xie, X. Q. 2010. Recent advances in fragment-based QSAR and multi-dimensional QSAR methods. International journal of molecular sciences, 11(10):3846–3866.

Radhika, P. R., Nair, R. A., Veena, G. 2019. A Comparative Study of Lung Cancer Detection using Machine Learning Algorithms. 2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), pages 1–4.

Reker, D., Schneider, G. 2015. Active-learning strategies in computer-assisted drug discovery. Drug Discovery Today, 20(4):458–465.

Roy, N., Kadam, R. U. 2007. Recent trends in drug-likeness prediction: A comprehensive review of <i>In silico</i> methods. Indian Journal of Pharmaceutical Sciences, 69(5):609–609.

Shen, M., Xiao, Y., Golbraikh, A., Gombar, V. K., Tropsha, A. 2003. Development and Validation of k-Nearest-Neighbor QSAR Models of Metabolic Stability of Drug Candidates. Journal of Medicinal Chemistry, 46(14):3013–3020.

Sun, H. 2006. An accurate and interpretable Bayesian classification model for prediction of hERG liability. ChemMedChem: Chemistry Enabling Drug Discovery, 1(3):315–322.

Vijayarani, S., Muthulakshmi, M. 2013. Comparative analysis of bayes and lazy classification algorithms. International Journal of Advanced Research in Computer and Communication Engineering, 2(8):3118–3124.

Wallach, I., Dzamba, M., Heifets, A. 2015. AtomNet: a deep convolutional neural network for bioactivity prediction in structure-based drug discovery. arXiv preprint.

Willett, P. 2006. Similarity-based virtual screening using 2D fingerprints.

Yosipof, A., Guedes, R. C., García-Sosa, A. T. 2018. Data Mining and Machine Learning Models for Predicting Drug Likeness and Their Disease or Organ Category. Frontiers in Chemistry, 6.