Artificial Intelligence in the Life Sciences 2 (2022) 100049

Contents lists available at ScienceDirect

Artificial Intelligence in the Life Sciences

journal homepage: www.elsevier.com/locate/ailsci

LIDeB Tools: A Latin American resource of freely available, open-source cheminformatics apps

Denis N. Prada Gori, Lucas N. Alberca, Santiago Rodriguez, Juan I. Alice, Manuel A. Llanos, Carolina L. Bellera, Alan Talevi*

Laboratory of Bioactive Compounds Research and Development (LIDeB), Faculty of Exact Sciences, University of La Plata (UNLP), La Plata, Buenos Aires, Argentina

A R T I C L E I N F O

Keywords:
Cheminformatics
Open source
Web applications
Clustering
Decoy generation
Decoys
Druggability prediction
Druggability

A B S T R A C T

Cheminformatics is the chemical field that deals with the storage, retrieval, analysis and manipulation of an increasing volume of available chemical data, and it plays a fundamental role in the fields of drug discovery, biology, chemistry, and biochemistry. Open source and freely available cheminformatics tools not only contribute to the generation of public knowledge, but also to reduce the technological gap between high- and low- to middle-income countries. Here, we describe a series of in-house cheminformatics applications developed by our academic drug discovery team, which are freely available on our website (https://lideb.biol.unlp.edu.ar/) as Web Apps and stand-alone versions. These apps include tools for clustering small molecules, decoy generation, druggability assessment, classificatory model evaluation, and data standardization and visualization.

1. Introduction

Cheminformatics describes the use of information technology to handle chemical information [1]. The field itself has been integrated with the chemical sciences for many decades. However, the term was coined relatively recently, when the increasing amount of chemical data generated within the drug discovery field (e.g., due to the implementation of combinatorial chemistry and high-throughput screening platforms) made the use of chemical information technologies increasingly mandatory [2]. Although cheminformatics has a wide scope, core tasks within the cheminformatics field include the management of chemical databases and datasets, storage and retrieval of chemical information, structure-property relationships, in silico screening, and the design of combinatorial and focused libraries.

As in other areas related to informatics, many relevant cheminformatics software were developed under the open-source philosophy. The open-source paradigm began in the informatics field and is intrinsically related to the notions of collaborative research and public knowledge. It was later partially adopted in other fields, for example, in the pharmaceutical sector [3,4], when it was realized how efficient the collaborative model might become in relation to traditional close-doors and market-driven philosophies. The open-source approximation has special relevance to bridge technological and scientific gaps between low- and high-income countries and to address neglected needs from the poorest regions of the globe. This is well reflected in our organization: LIDeB (Laboratory of Bioactive Compounds Research and Development) is a small academic research center dependent on the University of La Plata (UNLP, Argentina) with a focus on drug discovery projects, and a particular interest in the field of infectious tropical diseases. As part of these, we have developed publicly accessible cheminformatics Web Apps deployed through the Streamlit framework and standalone source codes. Here, we provide an overview of these resources, which are freely accessible on our website at https://lideb.biol.unlp.edu.ar/. These include clustering, decoy generation, druggability prediction apps, and other secondary resources related to chemical data standardization and visualization.

Several data science and machine learning platforms have been developed thus far, enabling users with limited coding ability to create pipelines for data exploration, analysis, and mining. Some well-known examples include KNIME, Pipeline Pilot, and Alteryx. Although they offer free and open-source distributions, most advanced features or third-party applications are only available under commercial licenses, which sometimes makes them inaccessible to small research units in developing countries.

All applications within the LIDeB Tools suite were deployed as Web Apps on the Streamlit platform, so scientists can use them through a user-friendly interface using computational resources allocated to the cloud. Alternatively, their standalone distributions are written in plain Python, under an Object-Oriented Programming (OOP) paradigm and using open-source libraries, offering a higher level of customization and code reusability. Moreover, they can easily be plugged into existing chemoinformatic pipelines.

* Corresponding author.
E-mail address: alantalevi@gmail.com (A. Talevi).

https://doi.org/10.1016/j.ailsci.2022.100049
Received 23 September 2022; Received in revised form 24 November 2022; Accepted 1 December 2022
Available online 2 December 2022

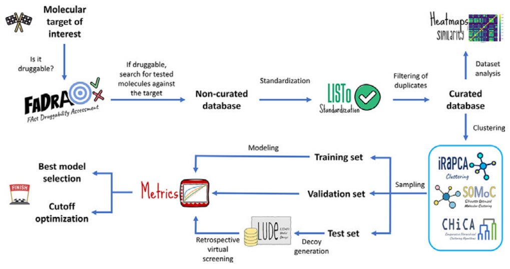


Fig. 1. A scheme of a drug discovery machine learning campaign, from assessment of target druggability to retrospective virtual screening for protocol validation. Possible steps where our reported open-source tools can be incorporated are shown.

A general workflow of a machine learning-based drug discovery project is shown in Fig. 1, which indicates the part of the workflow in which each of our tools might be integrated.

2. Clustering tools

Clustering of small molecules implies sorting a collection of chemical structures into smaller groups (clusters) such that the molecules within a cluster display a high degree of similarity (within-cluster similarity) compared to their similarity to elements allocated to other groups (between-cluster similarity). Ideally, clustering should provide compact clusters far away from each other within the relevant chemical space [5]. A clustering strategy requires a sensible decision regarding the clustering algorithm and parameter settings to obtain a good approximation of an optimal solution. Clusters found using a clustering procedure offer a hypothesis of the possible groups within the data because the ground truth (if existent) is generally unknown [6].

LIDeB Tools include two in-house clustering approximations, the iterative Random subspace Principal Component Analysis (iRaPCA) approach and Silhouette-Optimized Molecular Clustering (SOMoC), along with an implementation of some classical agglomerative clustering algorithms, Comparative Hierarchical Clustering Algorithms (CHiCA), which allows an interactive exploration of the output dendrograms for an informed choice of the most suitable clustering method and the optimal level to cut the tree, depending on the user's needs.

2.1. iRaPCA

iRaPCA clustering is based on an iterative combination of random subspace (feature bagging), dimensionality reduction by Principal Component Analysis (PCA) [7] and the K-means algorithm [8]. The iRaPCA algorithm can be described as a sequential combination of four steps: *input and encoding, dimensionality reduction, actual clustering* and, optionally, *iteration*. To adapt iRaPCA to a particular user's needs, the algorithm includes customizable hyperparameters.

In the *input step*, the molecules to be clustered should be provided as a .csv file, where each line corresponds to a molecule in SMILES format. After optional standardization employing MolVS [9], Mordred [10] is used to compute 1613 molecular descriptors for each molecule. Alternatively, users may provide their own pool of molecular descriptors as tab-delimited .txt file.

Subsequently, the *dimensionality reduction* step starts with the removal of descriptors with low information content, and 100 random subsets of 200 descriptors are created using a random subspace/feature bagging approximation [11], followed by the removal of highly correlated descriptors within each subset. Importantly, the user may define a higher number of random subsets in the standalone version of the script. Subsequently, PCA is conducted on each subset; by default, only the first two principal components are considered for the next steps of the procedure. In the final clustering step, the K-means algorithm is applied to the PC space by systematically varying the number of clusters (K) for each subset and evaluating them using the silhouette score [12]. The silhouette coefficient or score is a measure of how similar an element is to the other elements of its own cluster, compared to its distance to the elements allocated to other clusters. Assume the data have been into k clusters. The silhouette value $s(i)$ of a given data point i can be calculated as $b(i)\text{-}a(i)/\max\{(a(i),b(i)\}$, where $a(i)$ is the mean distance between i and all other data points in the same cluster and b(i) is the smallest mean distance of i to all points in any other cluster. $s(i)$ ranges from −1 to 1, with a value close to 1 indicating that the cluster to which i belongs is compact and distant from its closest cluster. The mean $s(i)$ over all points of a cluster reflects how tightly grouped all points in the cluster are. The silhouette score is the maximum value of the mean s(i) over all data of the entire dataset.

The results of the systematic variation of k per step by iRaPCA are presented in a silhouette vs. K plot for the 100 subsets (an example is shown in Fig. 2). The subset and K pair that delivers the highest silhouette score is typically selected. Other Cluster Validity Indexes (CVIs) are also provided: the Dunn Index [13], Davies–Bouldin (DB) Index [14] and Caliński-Harabasz (CH) Index [15].

Finally, the *iteration step* starts with the calculation, for each cluster, of the ratio of the number of molecules in a cluster over the total molecules in the clustered dataset, followed by a new round of clustering for those clusters that exceed a user-selected cutoff value of this ratio. Once there are no more clusters that exceed this ratio, or the maximum number of rounds defined by the user has been executed, the clustering ends and several output files are generated: *Cluster_assignations.csv*, which compiles all the molecule's codes and their cluster membership; *Cluster_distribution.csv*, which specifies the obtained clusters and their sizes; *Validations.csv*, which displays the values of the CVIs for the clustered dataset plus the CVIs obtained when randomly allocating the dataset compounds to an identical number of clusters; and *Settings.csv*, which summarizes all the settings used in the run.

As a benchmark exercise the clustering performance of this algorithm was tested across 29 diverse datasets extracted from the literature [16], comparing it with classic clustering techniques: Butina [17], and agglomerative clustering based on single-, complete-, average-, and weighted-linkage [18] (Tanimoto coefficient and Morgan Fingerprints of length = 1024 and radius = 2, an ECFP4-like representation, were used to characterize the molecules). In the case of Butina, the number of neighbors for each molecule in the datasets was calculated for different Tanimoto distance cutoff levels ranging from 0.5 to 0.90, with a step size of 0.01. The algorithm, as implemented in RDKit [19], was applied to each similarity level of the ranking generated by sorting the molecules according to the number of neighbors. For each Tanimoto level, the silhouette score was calculated and the optimal Tanimoto level, defined as the one that provides the highest silhouette score, was selected. In the case of the agglomerative clustering approaches, the single-, complete-, average-, and weighted-linkage algorithms, as implemented in SciPy [20], were applied over the Tanimoto distance matrix to obtain the corresponding dendrogram and the distance cutoff value providing the highest silhouette score was selected. In all the cases, the previously mentioned set of CVIs was computed for comparison. The silhouette scores obtained for the 29 datasets using the five clustering methodologies are presented in Fig. 3. The silhouette values for iRaPCA correspond to the first round of the iterative procedure. It can be observed that the silhouette scores for iRaPCA oscillated between 0.85 - 0.98 (that is, very close to the ideal value of 1), much higher than those of the other methods under comparison.

2.2. SOMoC

SOMoC clustering is based on a combination of molecular fingerprinting, dimensionality reduction using the Uniform Manifold Approximation and Projection (UMAP) algorithm [21,22], and clustering with the Gaussian Mixture Model (GMM) [23,24]. The general workflow of the algorithm is similar to that of iRaPCA in the initial steps of *input and*

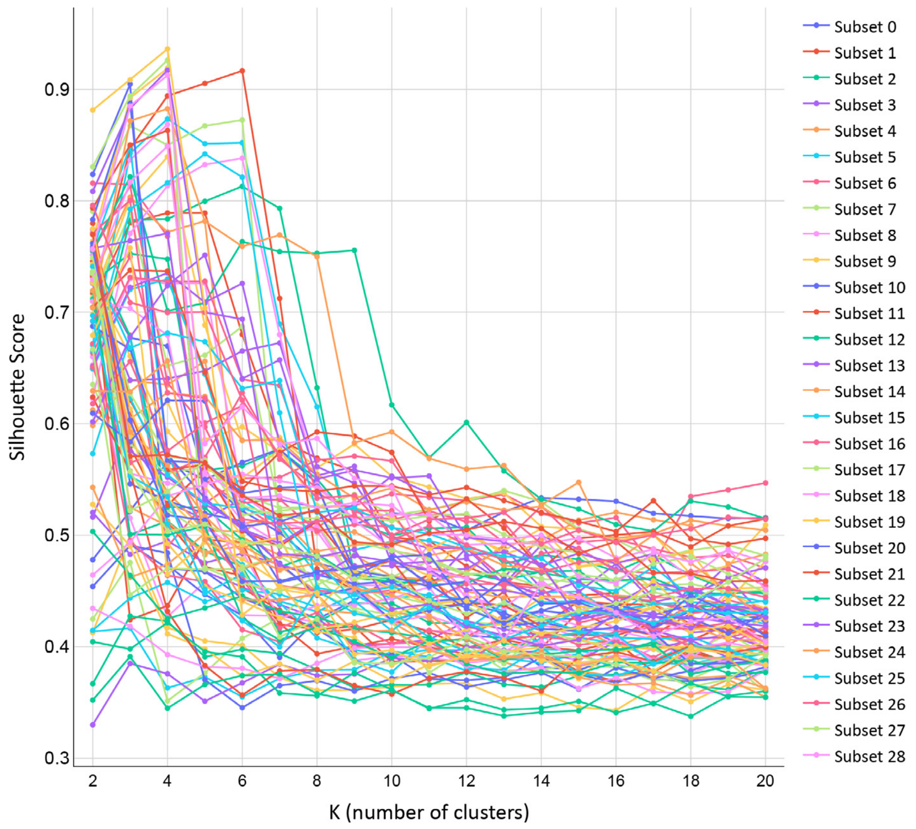


Fig. 2. Example of silhouette score vs. K plot generated by iRaPCA, for the default 100 randomly chosen descriptor subsets.

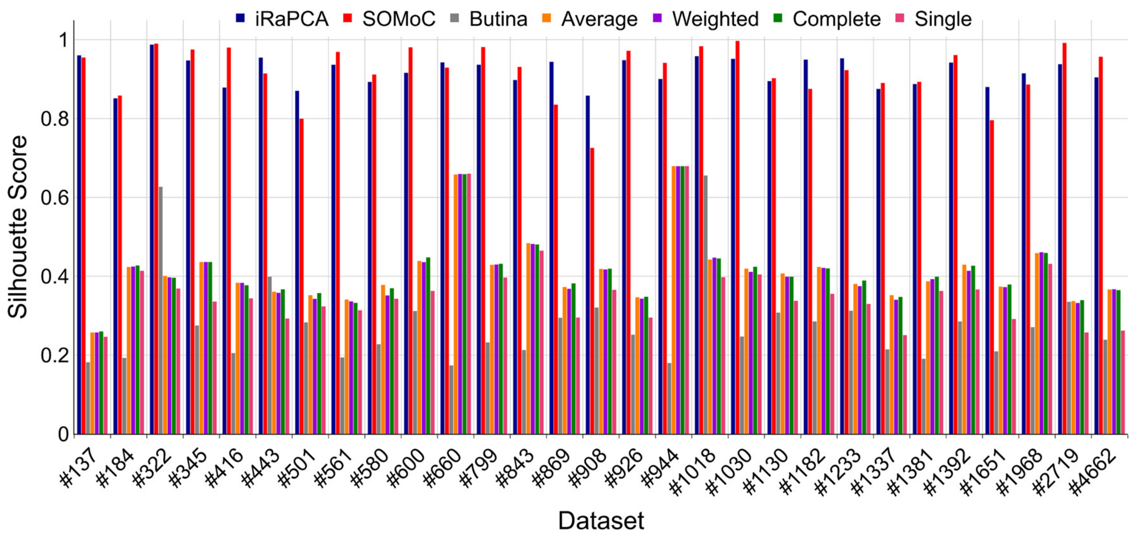


Fig. 3. Performance comparison of the iRaPCA and SoMOC algorithms versus Butina and several agglomerative hierarchical clustering algorithms across 29 benchmark datasets. These were ordered and labeled in the figure according to their size (i.e., by the number of compounds in each dataset).

encoding, dimensionality reduction, and *clustering*. Its hyperparameters, as in the case of iRaPCA, can also be customized.

In the *input step*, the molecules are provided again in SMILES notation as a .csv file, but instead of calculating molecular descriptors, the molecular representations are encoded into EState1 molecular fingerprints, calculated by RDKit, with a fixed-length fingerprint containing 79 features [25]. The UMAP algorithm, a nonlinear dimensionality reduction technique based on Riemannian geometry and algebraic topology, is then applied to reduce the hyperdimensional space of the fingerprints, retaining the most informative features of the process. The size of the embedded space is determined based on the size of the dataset. In the clustering step, the GMM algorithm is applied to the molecules in the embedded UMAP space to search for the number of K clusters that maximizes the silhouette score. An elbow plot of the silhouette score vs. K is presented for visual estimation of the optimal K. The same additional CVIs previously described for iRaPCA are also calculated and the same output files are generated.

To validate SOMoC as a clustering tool, a benchmarking exercise similar to the one used for iRaPCA was performed across the same datasets, using Butina and agglomerative clustering algorithms as comparators. The SOMoC algorithm displayed Silhouette scores ranging between 0.72 - 0.99 across the 29 datasets. The results are presented in Fig. 3.

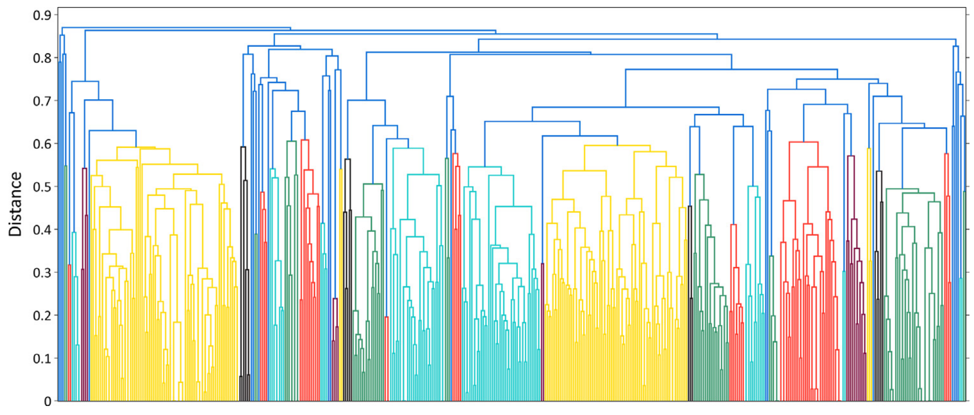


Fig. 4. Example of a dendrogram generated by CHiCA.

2.3. CHiCA

CHiCA implements diverse classical hierarchical agglomerative clustering approaches. Despite exhibiting modest performance in comparison to iRaPCA and SOMoC, as shown in Fig. 3, CHiCA allows the user, by using interactive graphs, an intuitive and visual choice of the number of clusters to consider. Molecules are characterized by Morgan fingerprints with user-defined radio and length (by default, 2 and 1024, respectively) [26]. It is possible to include special, fuzzy features in the fingerprints, such as hydrogen-bond donors, acceptors, aromatic rings, halogen atoms, and acidic or basic groups. As in our other clustering apps, the input molecules are provided in SMILES notation through .csv files, and a distance matrix is obtained. CHiCA allows the measurement of the pairwise distance between molecules using different metrics (e.g., Jaccard, Euclidean, Cosine) by employing SciPy [20]. The SciPy's clustering package is then used to perform hierarchical clustering. Different agglomerative methods are available for this purpose. By default, single-linkage is used.

The clusters are plotted in an interactive dendrogram (Fig. 4) that displays the composition of each cluster by drawing the links between a non-singleton cluster and its children. The top of a link indicates a cluster and its two legs specify which clusters have been merged, while the length of the legs reflects the distance between the child clusters. This dendrogram allows the user to visually select a distance cutoff for clustering molecules based on their distribution in the generated clusters and the number of outliers.

Additionally, an interactive scatter plot of distance vs silhouette score vs the number of molecules is shown (Fig. 5). This plot allows the user to select an optimal cutoff for clustering based on this score or the number of outliers, number of small clusters (defined as clusters with more than one molecule and less than 5% of the total molecules by) or number of dense clusters (defined as clusters with more than 5% of the total molecules by default).

Once the optimal cutoff has been identified, CHiCA is re-run with this distance value, generating four output files: *Clustering_assignations.csv* which specifies which molecules have been assigned to which cluster according to the chosen cutoff value; *Cluster_distribution.csv*, which specifies the number of obtained clusters and their sizes; *Validations.csv* which contains the values of the CVIs; and *Settings.csv*, which contains the selected parameters in the run.

The molecular structures of the group centroid for the most populated cluster obtained using iRaPCA, SOMoC and CHiCA for the human cannabinoid receptor 1 dataset (Hum_can_rec) are shown in Fig. 6.

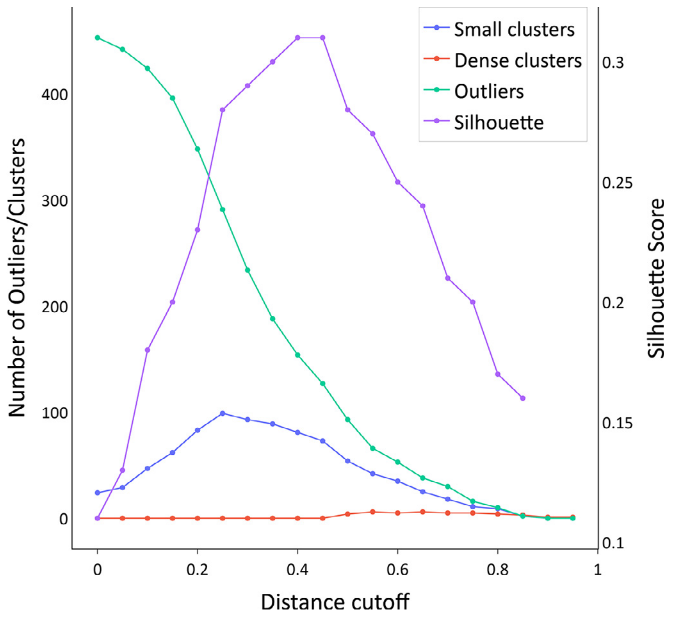


Fig. 5. Example Distance cutoff vs. Silhouette score vs. Number of Molecules plot.

3. Decoy generation

Any virtual (or, for the case, wet) screen is confronted with an intrinsic class imbalance: the number of active compounds in a chemical library is significantly exceeded by the number of inactive compounds. Therefore, validation of the virtual screening protocol should realistically reflect this scenario. In other words, the ability of the virtual screening method to retrieve relatively few known active compounds from a large chemical library predominantly composed of inactive compounds should be evaluated. This is often performed through retrospective screening experiments in which a comparatively small number of known active compounds are dispersed among a large number of (proven or putative) inactive compounds. Putative inactive compounds are also known as decoys, and in essence they are similar to the active compounds in terms of an array of physicochemical properties, such as molecular weight (MW), logP (see belong) but topologically dissimilar. As the molecular topology is key to complementarity with the pharmacological target, it can be assumed that decoys that are topologically distinct from active compounds will likely be inactive compounds. If the virtual screening protocol works properly, it is expected that the known active compounds will be preferentially ranked above decoys. Truchon and Bayly showed that, for accurate estimation of enrichment metrics, the proportion of active compounds in the library that is subjected to retrospective screening should not exceed 0.05 [27]. Furthermore, the "saturation effect" is avoided or at least ameliorated when such a proportion is observed. When inactive compounds are scarce, known active compounds can be complemented by putative inactive compounds [28,29]. That is, unverified inactive molecules that are cho-

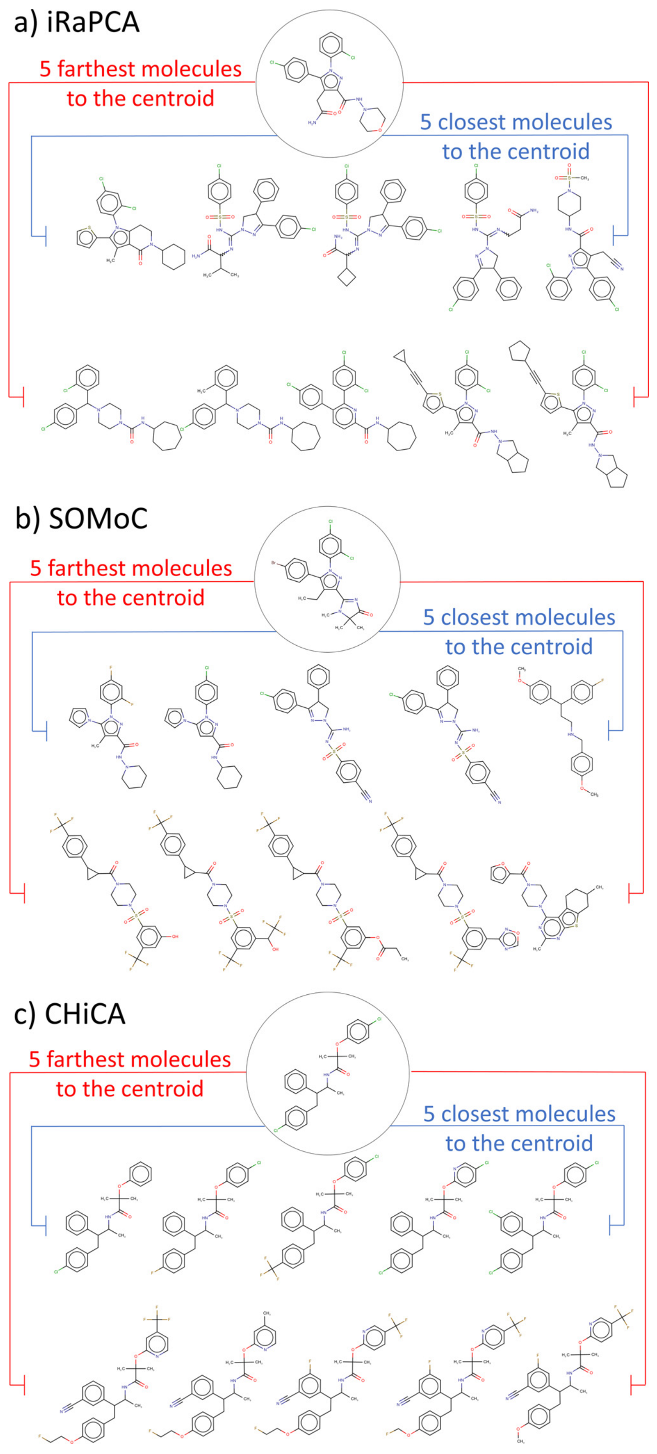

a) iRaPCA
5 farthest molecules to the centroid
5 closest molecules to the centroid
b) SOMoC
5 farthest molecules to the centroid
5 closest molecules to the centroid
c) CHiCA
5 farthest molecules to the centroid
5 closest molecules to the centroid

Fig. 6. The figure shows the centroids and the closest and farthest neighbors from the centroids (within the most populated cluster) obtained for the "Hum_can_rec" dataset using iRaPCA, SOMoC and CHiCA (weighted linkage) methods.

sen on some theoretical basis to assure a rather small probability of mislabeling; the Directory of Useful Decoys Enhanced (DUD-E) [30], with a recently release improved version [31] is possibly the most popular online decoy-generation tool to validate structure-based screening protocols, although some other decoy generators have also been reported, including methods that rely on hard machine learning approaches [32–34]. Although undoubtedly useful, DUD-E was at times limited by the capacity of its online server to return putative decoys in a short time, sometimes demanding several days to complete a task (this was particularly true when the number of active queries increased to hundreds and has been greatly expedited in DUDE-Z). This led us to develop a similar approximation available as a standalone script, which we called LIDeB's Useful Decoys (LUDe).

3.1. LIDeB's useful decoys

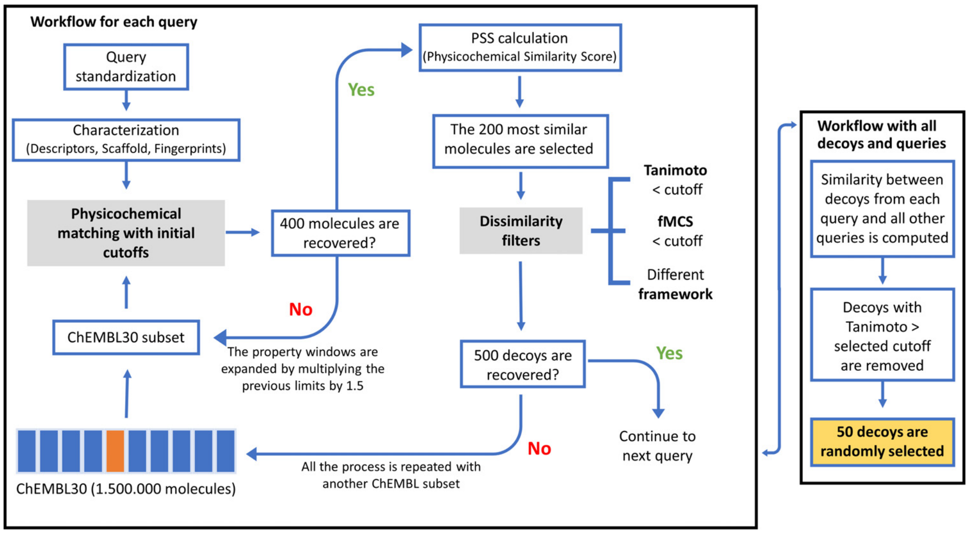
LUDe retrieves decoys from a curated ChEMBL30 database (curation details are provided in Supplementary Information). These decoys are paired with some general physicochemical properties of the query active compounds, which ensures that the generated decoys are challenging (i.e., non-trivial) so that enrichment bias [35,36] is avoided. However, as described later, different filters are subsequently applied to ensure that the provided decoys are topologically distinct from (and share no scaffolds with) any of the active compounds; thus the false-negative bias is also reduced [36]. The general LUDe workflow is illustrated in Fig. 7. Our in-house script requires a .txt file as input with molecular representations of the query compounds (known active compounds) in SMILES notation. The queries are then standardized using MolVS [10], returning the largest organic covalent unit in the molecule with all atoms replaced with the most abundant isotope for that element and their charge removed. Additionally, the ionization state at pH 7.4 is calculated with the appropriate charges using Openbabel [37]. Once the molecules have been standardized, the following physicochemical descriptors are calculated using the rdkit.Chem package: molecular weight (MW), log P (LogP), number of rotatable bonds (nRotB), number of H-bond acceptors (nHAcc), number of H-bond donors (nHDon) and formal charge (Charge). Morgan fingerprints (with tunable lengths and radii, by default radius=2 and length=1024) and the Murcko Scaffold are next obtained for each compound, using the rdkit.Chem package for this purpose.

Afterwards, decoys are sought across different subsets of the curated ChEMBL database containing 150 K molecules per subset. A physicochemical similarity search with established cut-off values is carried out in one randomly selected ChEMBL subset. By default, the retrieved decoys will present features in the following range in comparison to their query: MW±20 Daltons, LogP±0.5 log units, nRotB±1 bond, nHAcc±1 bonds, nHDon±1 bonds, and Charge±1 units, although the tolerance windows can be narrowed or expanded by the user. If fewer than 400 molecules are recovered in this randomly chosen ChEMBL subset, the property limits are extended by a factor of 1.5, up to five times. For instance, in the case of MW the window is extended, round by round, to ±30 Daltons, ±45 Daltons, etc.

Physicochemical similarities between the queries and the generated preliminary list of decoys are estimated. For this, the difference between each known active compound and each decoy in terms of the six pairing properties is calculated and summarized with a physicochemical similarity score (PSS) ranging from 0 (lowest similarity) to 1 (highest similarity) [38]. The compounds are then ordered with descending PSS values, and the top 200 are selected.

Three successive topological similarity filters are applied to these retrieved decoys to selectmolecules that are topologically less similar to the query compound. First, the Tanimoto similarity coefficient between the query compound and each potential decoy is calculated. By default, only decoys with a maximum similarity coefficient of 0.2 are kept. The maximum common substructure (MCS) between the query compound and each surviving decoy is then determined, and the ratio between the number of atoms in the MCS and the total number of atoms in the query compound (fMCS) is calculated [39]. Only decoys with fMCS below a user-defined value (by default, 0.5) are retained. The Murcko scaffold [40] of the query compound is compared with those of the potential decoys, keeping only those that present scaffolds that are different from the query.

The previous steps are repeated with another ChEMBL subset until a total of 500 potential decoys for each query compound have been recovered or all subsets have been exhausted, whichever happens first.



Workflow for each query
Query standardization
Characterization (Descriptors, Scaffold, Fingerprints)
Physicochemical matching with initial cutoffs
400 molecules are recovered?
Yes
No
The property windows are expanded by multiplying the previous limits by 1.5
ChEMBL30 subset
ChEMBL30 (1.500.000 molecules)
PSS calculation (Physicochemical Similarity Score)
The 200 most similar molecules are selected
Dissimilarity filters
Tanimoto < cutoff
fMCS < cutoff
Different framework
500 decoys are recovered?
Yes
Continue to next query
No
All the process is repeated with another ChEMBL subset
Workflow with all decoys and queries
Similarity between decoys from each query and all other queries is computed
Decoys with Tanimoto > selected cutoff are removed
50 decoys are randomly selected

Fig. 7. General workflow for LUDe.

Table 1
Number of decoys generated in each step of the LUDe workflow for five active compounds in the FABP4 dataset of DUD-E.

Query Molecules	Selected by matching physicochemical properties	Pass the Tc filter per active	Pass the fMCS filter	Pass the Framework filter	Non- duplicated decoys per active:	Pass the cross-check TC filter against all queries submitted
Query 1	800	582	582	582	582	217
Query 2	1000	641	640	640	625	286
Query 3	1000	624	623	623	586	285
Query 4	1000	755	754	754	523	185
Query 5	800	642	622	622	515	182

Tc: Tanimoto coefficient.

After performing the workflow for all query molecules, the molecular fingerprints of the resulting list of decoys are compared to those of all query compounds, and only decoys with a Tanimoto similarity below 0.2 (by default) to any query are retained. This ensures that the decoys are not only different from the query they were derived from but also from any of the known active compounds used to generate them. Up to 50 decoys per query compound are retrieved by default, although this number can also be customized. The following output files are generated: Generated_decoys.csv, which contains all the decoys in SMILES notation; Decoys_analysis.csv, which provides a table containing a summary of the number of molecules that passed each successive filter; and Decoys_setting.csv, containing a summary of all the settings used in the run. As an example, Table 1 presents the number of decoys obtained when using the active compounds included in the Fatty Acid Binding Protein 4 (FABP4) subset from DUD-E as queries for LUDe. This dataset contains 47 FABP4 inhibitors. Only five of these have been included in the table, for illustrative purposes. For instance, LUDe found 800 molecules that were physicochemically similar to query 1. Among these, 582 passed the three filters that ensure topological dissimilarity from the query. Of these, only 217 molecules were dissimilar to the remaining queries. By default, fifty decoys were then randomly selected for query 1, and the same analysis was repeated for every other query. It is important to emphasize that because 2D similarity methods are used to select decoys for each query (excluding decoys with high 2D similarity to any query), LUDe decoys should not be used to assess the performance of in silico screens based on 2D similarity approximations/molecular fingerprints. This is a general warning for any method that uses a 2D similarity criterion to exclude potential decoys from the list.

The performance of LUDe was compared with that of DUD-E across 102 target subsets of active compounds and their corresponding DUD-E-generated decoys, which are available on the DUD-E website (http://dude.docking.org/subsets). For each set of active compounds, a new set of decoys was generated using LUDe. The total number of decoys obtained for each subset is included in the Supplementary Information (Table S1).

The quality of the generated decoy molecules, in terms of the physicochemical matching of decoys and the risk of allowing latent active compounds in the decoy set (LADS), was determined by the deviation from the optimal embedding score (DOE score) and the assessment of the Doppelganger score, respectively, as recommended by Vogel et al. [38]. The DOE score was obtained by analyzing the spatial distances of the molecules in the chemical space defined by the six normalized physicochemical properties used to match queries and decoys. Briefly, the distances from each active compound to all remaining active compounds and decoys were calculated in the normalized multidimensional space and molecules were sorted in ascending order according to these distances. The real class of each molecule (labeled 1 for active compounds and 0 for decoys) and the obtained distances were later used to build a Receiving Operating Characteristic (ROC) curve for each active compound, and the average absolute value of the difference between the area of these ROC curves and the area of a randomly sorted list of compounds (0.5) was defined as the DOE score:

$$DOE\ score = \frac{1}{m}\sum_{a=1}^{m} ABC_a^{DOE}$$

where m is the number of queries (active compounds). Optimal embedding of active compounds into decoys corresponds to a value of zero, whereas a value of 0.5 denotes no embedding at all.

Fig. 8A shows that the DOEs obtained with LUDe decoys are above those obtained with DUD-E decoys for 65% of the 102 targets. Moreover, 86% of the subsets of decoys generated by our script achieved a DOE score below 0.2, whereas this proportion represented 74% of the targets for DUD-E decoys.

In contrast, the Doppelganger score captures the structural similarity between actives and their most structurally related decoys. To assess

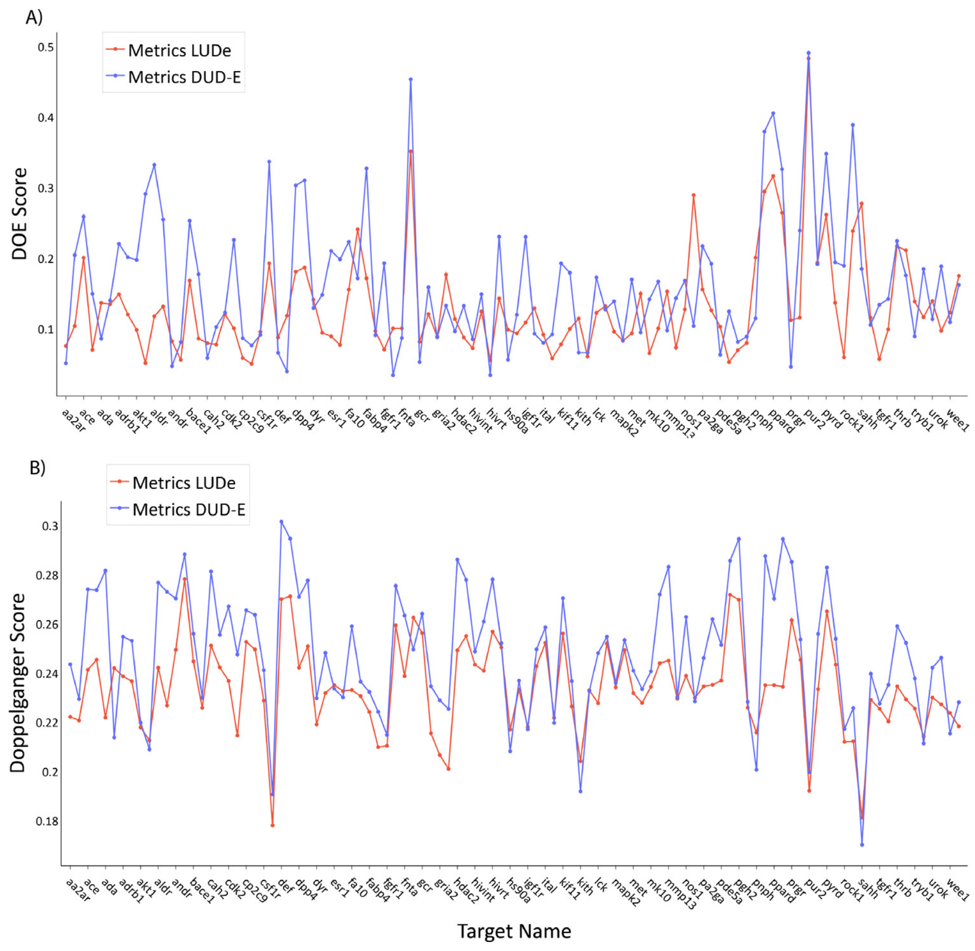


Fig. 8. A) DOE scores of the original DUD-E set (blue) compared with the LUDe generated decoys (red) across 102 targets. The targets with even indices were not labeled on the x-axis owing to space limitations. B) Mean doppelganger scores of the original DUD-E set (blue) compared with the LUDe generated decoys (red) across 102 pharmacological targets. The targets with even indices are not labeled on the x-axis owing to space constraints.

this score, FCFP6 fingerprints were generated using RDKit [19] for every known active compound and decoy, and for each of the 102 pharmacological targets; the similarity between each decoy and each active compound (query) was assessed using the Tanimoto coefficient. The Doppelganger score of a decoy is defined as the maximum value of the Tanimoto coefficient obtained in this way, across all active compounds used as queries. For each target, we report the mean Doppelganger score over all decoys and the maximum structural similarity between an active and a decoy. Fig. 8B shows the mean Doppelganger score over all decoys for each of the 102 targets. The Doppelganger score was lower for 85% of the targets for LUDe decoys than for DUD-E decoys, with an average Doppelganger score across the targets of 0.23 for LUDe decoys and 0.25 for DUD-E decoys. This possibly reflects the effect of the additional filters implemented in LUDe to ensure topological dissimilarity between the decoys and queries (fMCS, Mucko scaffold). The maximum average Doppelganger score per target was very similar for LUDe and DUD-E (0.36 and 0.37, respectively). This suggests that the chance of false-negative decoys tends to be reduced for LUDe.

4. Druggability prediction tool

Druggability refers to the ability of a given protein to bind with a high affinity to small drug-like molecules [41]. Assessing the druggability of potential pharmacological targets prior to initiating a target-focused drug discovery campaign is crucial. *Fast Druggability Assessment* (FaDrA) is a druggability prediction web application based on four linear classifiers that can discriminate druggable from non-druggable targets from complete proteomes within a few minutes, with acceptable accuracy, based only on the protein sequence.

To develop these algorithms, a protein dataset was compiled from ChEMBL [42], DisProt [43] and DCD [44]. Proteins were labeled "druggable" if at least two of the following conditions were met: more than 100 reported small-molecule modulators with IC50 < 20 μM; more than 5% hit rate in reported high-throughput screening campaigns; at least five publications reporting modulators per year, indexed in Scopus. Otherwise, the proteins were labeled as "non-druggable". Using these criteria, 222 protein sequences were retrieved, of which 111 were considered druggable and 111 were considered non-druggable.

Seventy percent of the proteins in each class was sampled for the training set of the classifiers. The remaining proteins were used as a test set to validate the generated models externally. To realize this sampling representatively, we used a clustering strategy that combined dimensionality reduction of Zernike descriptors [45] by Principal Component Analysis (PCA) followed by the application of the k-means clustering algorithm.

The dataset proteins were then characterized using 147 CTD descriptors available in PyBioMed [46], which are independent of the secondary, tertiary, or quaternary structure of the proteins, as they only require the sequence of the protein as input. The pool of 147 descriptors for the training set was subjected to a random subspace approach generating 1000 subsets of 20 descriptors each. Then, using a Forward Stepwise strategy we generated 1000 linear classifiers with no more than five descriptors each. The robustness of the models and chance of spurious correlations were assessed using Leave-Group-Out (LGO) cross-validation and randomization tests. LGO cross-validation was performed using randomly stratified subsets composed of eight druggable and eight non-druggable proteins that were removed from the training set in each cross-validation round, and the model was regenerated using the remaining proteins as training examples. The resulting model was used

Table 2
Results of the randomization test and the LGO cross-validation for the four best models.

Model	Accuracy (s.d.)[1]	Accuracy Randomization test (s.d.)[1]	Accuracy LGO (s.d.)[1]
260	0.820 (0.066)	0.499 (0.150)	0.723 (0.110)
361	0.808 (0.069)	0.500 (0.134)	0.712 (0.113)
424	0.814 (0.068)	0.503 (0.129)	0.758 (0.110)
763	0.801 (0.069)	0.499 (0.128)	0.744 (0.104)

[1] s.d. indicates Standard Deviation.

Table 3
Performance of the best models in the external validation.

Model	Accuracy (s.d.)[1]	Recall (s.d.)[1]	Precision (s.d.)[1]
260	0.803 (0.066)	0.939 (0.059)	0.744 (0.070)
361	0.802 (0.070)	0.757 (0.108)	0.839 (0.085)
424	0.818 (0.069)	0.818 (0.095)	0.825 (0.082)
763	0.803 (0.068)	0.908 (0.072)	0.756 (0.074)

[1] s.d. stands for Standard Deviation.

to predict class labels of the removed proteins. The procedure was repeated 1000 times, with each of the training set examples removed at least once.

In contrast, in the randomization test the class label was randomized across the proteins comprising the training set. The training set with the randomized variable was used to train new models, from the descriptor selection step. This procedure was repeated 1000 times for each descriptor subset. Randomized models are expected to have poor accuracy compared with real models.

The results of both tests are presented in Table 2. The results suggest some degree of overfitting for each classifier, but very low chances of spurious correlation (as seen as the mean accuracy in the randomization test, practically identical to the expected accuracy for random classification, i.e., 0.5 for balanced sets).

The equations of the four best models and the definitions of the descriptors included in them have been incorporated in the Supplementary Information. The predictive power of the selected models was further examined by calculating the accuracy, recall and precision over the 66 protein sequences that compose the test set (Table 3).

Finally, the predictions of the four models were combined in a meta-classifier to reduce the number of false positives, that is, proteins that are predicted to be druggable but are actually non-druggable. The meta-classifier considers the predictions of the four individual models and the assessment of the applicability domain (AD) using the leverage approach and 3d/n as the cut-off value, where d is the number of descriptors in the corresponding model and n is the number of compounds in the training set. Finally, meta-classifier prediction is performed using the decision tree shown in Fig. 9.

Table 4 compares the performance of each individual classifier, in terms of accuracy, with that of the meta-classifier. It can be observed that the meta-classifier obtains the best accuracy within the non-druggable class; it is also the scheme that leads to less “non-conclusive” results.

FaDrA allows druggability prediction of complete proteomes within a few minutes. The app receives a “.fasta” file as input. This file may include a simple sequence, a few protein sequences, or a complete proteome. After reading the file, the necessary descriptors for the models are calculated using PyBioMed and the four models are applied to each protein. In addition, the applicability domain is determined. A pie chart is displayed (Fig. 10) showing the percentage of “druggable” and “non-druggable” proteins, and the percentage of “non-conclusive” results. Finally, a .CSV file is generated that summarizes the predictions for the four individual models and the meta-classifier, as well as whether each protein sequence belongs to the ADs of the models. For example, the complete proteome of *Escherichia coli* (Genome assembly ASM584v2) was run in FaDrA. 4298 proteins were loaded of which 1978 (46%) were predicted as “non-druggable”, 2084 (48.5%) were predicted as “druggable”, and 236 (5.5%) generated non-conclusive results. Table 5 shows the predictions for the first ten proteins in the E. coli proteome.

5. Other tools

In addition to the previously discussed Web Apps, LIDeB Tools also include three other secondary tools that employ available packages and algorithms with minor changes and are implemented with a user-friendly interface. These are Heatmap-Similarity, LIDeB’s Standardization Tool (LISTo), and Metrics.

The Heatmap-Similarity Web App builds a heatmap of molecular similarities using RDKit. These plots of intermolecular similarity allow for fast visual inspection of the molecular diversity of chemical datasets. The inputs for the app are two .txt files (which can be identical or different), where each line corresponds to a molecule in SMILES. The algorithm constructs a similarity matrix between two loaded sets of compounds, computed as a Tanimoto similarity coefficient using Morgan fingerprints with a user-defined radius (from 1 to 3, 2 by default) and bit length (1024 by default). The matrix is plotted as a heatmap with a color scale indicating similarity, and the resulting plots can be downloaded as .png files.

LISTo is a standardization Web App that helps to automatically standardize collections of chemical structures that may present different, non-homogeneous molecular representations. This simple standardization is useful for ensuring a homogeneous format of molecules before calculating any conformation-independent molecular descriptor. In this process each molecule, submitted in a .txt file in SMILES notation, passes through a series of standardization steps that are included in MolVS. By default, LISTo retains only the parent fragment of the molecule (the largest organic covalent unit in the molecule), removes all stereochemical information from tetrahedral centers and double bonds, mantains the uncharged version of the fragment parent, replaces atoms with the most abundant isotope, disconnects metals from organic atoms, applies a series of transformations to correct common drawing errors, and removes hydrogen atoms. The Web App also returns the canonical tautomer, which is a unique representation, selected with a scoring function from all possible tautomers that could be generated from a molecule; importantly, the canonical tautomer is not necessarily the most energetically favorable [47]. A log file with additional information regarding potential problems for specific standardization actions can be downloaded. In addition, the standardized molecules can be visualized in the Web App through the interactive chemical viewer mols2grid (https://github.com/cbouy/mols2grid).

Finally, the Metrics Web App evaluates the performance of classificatory models by calculating different metrics implemented in

Table 4
Accuracy by category for each selected model and the meta-classifier in the test set.

	Model 260	Model 361	Model 424	Model 763	Meta-classifier
Druggable accuracy	0.848	0.800	0.879	0.933	0.727
Non-druggable accuracy	0.593	0.700	0.643	0.667	0.900
Number of “Non-conclusives”	9	6	5	8	3

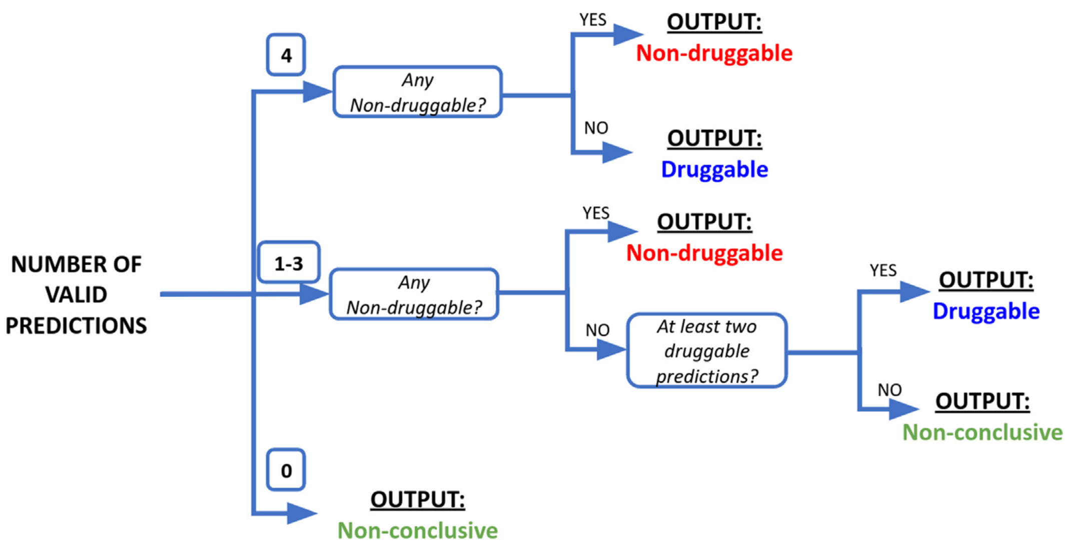


Fig. 9. Decision tree for meta-classifier druggability assessment. For each protein, the predictions of four individual classifiers are provided along with their respective AD predictions. Depending on how many predictions are considered valid (i.e., inside the AD), the class assignment follows different criteria.

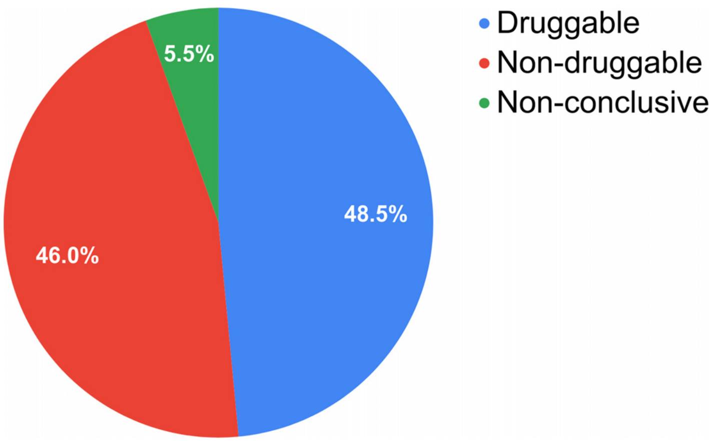


Fig. 10. Pie chart for FaDrA prediction of the *E. coli* proteome.

Table 5
Output table for the prediction of the first 10 proteins in *E.coli* proteome.

Protein ID	PREDICT 1	AD 1	PREDICT 2	AD 2	PREDICT 3	AD 3	PREDICT 4	AD 4	CLASSIFICATION
NP_414542.1	Druggable	NO	Non-druggable	NO	Non-druggable	NO	Druggable	NO	Non-conclusive
NP_414543.1	Druggable	YES	Druggable	YES	Druggable	YES	Druggable	YES	Druggable
NP_414544.1	Druggable	YES	Druggable	YES	Druggable	YES	Druggable	YES	Druggable
NP_414545.1	Druggable	YES	Druggable	YES	Druggable	YES	Druggable	YES	Druggable
NP_414546.1	Non-druggable	NO	Non-druggable	NO	Druggable	NO	Druggable	NO	Non-conclusive
NP_414547.1	Non-druggable	YES	Druggable	YES	Druggable	YES	Druggable	YES	Non-druggable
NP_414548.1	Druggable	YES	Druggable	YES	Druggable	YES	Druggable	YES	Druggable
NP_414549.1	Druggable	YES	Druggable	YES	Druggable	YES	Druggable	YES	Druggable
NP_414550.1	Druggable	YES	Druggable	YES	Druggable	YES	Non-druggable	YES	Non-druggable
NP_414551.1	Druggable	YES	Druggable	YES	Druggable	YES	Druggable	YES	Druggable

scikit-learn [48]. The input is a .txt file with two columns, named “class” and “score”, where each line corresponds to a molecule; the “class” column should indicate the active and the inactive compounds (class 1 and 0 respectively) and the “score” column the score which was given by the classification model that was employed. The output includes different performance metrics (accuracy, balanced accuracy, F-measure, precision, recall and Matthews correlation coefficient) and a confusion matrix for the user-selected score threshold to split active and inactive compounds (by default is set to 0.5). Moreover, this threshold can be modified, and in this case the results will be updated immediately. Moreover, the ROC and Precision-Recall curves are plotted (Fig. 11a) and their areas under the curve are presented in a table along with the BEDROC for a specific alpha (20 by default) and the EF for a specific top-ranked fraction of the entire set (0.01 by default). The values of alpha and the fraction can be modified by the user and the results will be immediately updated in the table. These last metrics are presented as a mean value with a standard deviation, sampling without replacement (by default, 100 iterations) 85% of the entire dataset (by default). Additionally, a Positive Predictive Value surface (PPV) rotatable surface is generated displaying the interplay between PPV, the Se/Sp ratio (sensitivity/specificity ratio associated with a given score threshold value) across a theoretical range of Ya (yield of actives) (Fig. 11b). PPV helps estimate the probability that a predicted hit will confirm its activity when subjected to experimental testing. By visual inspection of this

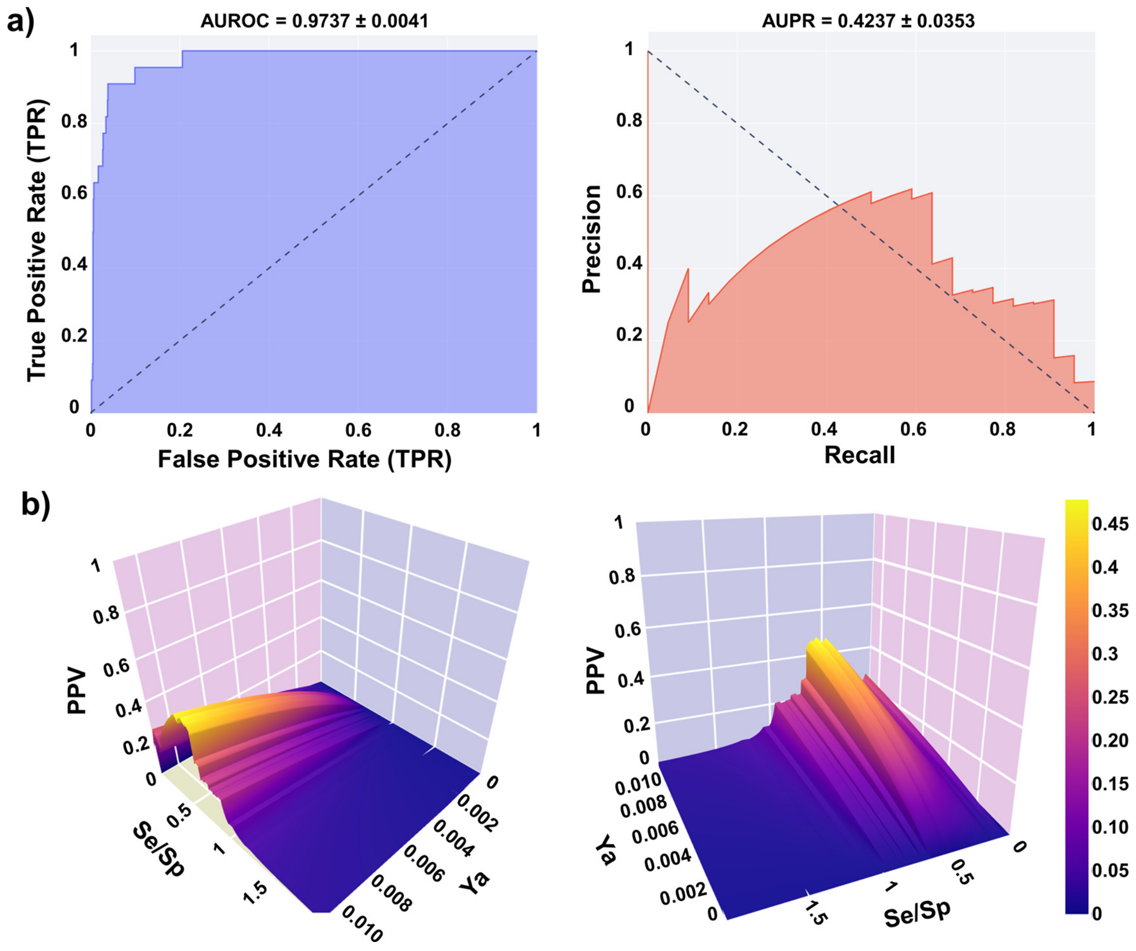


Fig. 11. a) Example of ROC and PR curves generated by Metrics Web App; b) Examples of PPV surfaces generated by Metrics Web App.

3D plot, or by analyzing the table with the Se, Sp, and PPV values for different score cutoff values, it is possible to choose an adequate score threshold, associated with the desired PPV range, to select predicted hits in prospective virtual screening experiments [49]. Importantly, the Se/Sp ratios observed in the ROC curve for each cut-off value are used to build the graph, assuming that such ratios will remain approximately the same across different datasets.

6. Conclusions

We presented and discussed an array of cheminformatics tools developed in our academic drug discovery laboratory. These open-source resources are freely available as online Web Apps (https://lideb.biol.unlp.edu.ar/) and as standalone versions (https://github.com/LIDeB/). They were developed using publicly available open-source resources and internal programming.

The open source/open-software model contributes to the development of public knowledge and bridges the technological gap between low- to middle-income countries and high-income countries. Although such a gap exists in practically all fields of science and technology, it is possibly narrower within the informatic community, because of the persistent efforts of global developers to make valuable resources freely available to the public.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Have shared the link to my code.

Acknowledgments

The authors are funded by UNLP (Incentivos UNLP and Subsidio para Jóvenes Investigadores UNLP 2019 y 2021) and Agencia Nacional de Promoción Científica y Tecnológica (ANPCyT), via grants PICT 2017-0643, PICT 2019-1075, PICT 2019-00984 and PICT-CATI-2021-00073. A.T., L.N.A. and C.L.B. are members of the Research Career from the Argentinian National Research Council (CONICET, Argentina). The authors gratefully acknowledge the support of Streamlit.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.ailsci.2022.100049.

References

[1] Xu J, Hagler A. Chemoinformatics and drug discovery. Molecules 2002;7(8):566–600. doi:10.3390/70800566.
[2] Leonis G, Melagraki G, Afantitis A. Open source chemoinformatics software including KNIME analytics platform. In: Leszczynski J, editor. Handbook of computational chemistry. Dordrecht: Springer Netherlands; 2016. p. 1–30.
[3] Bhardwaj A, Scaria V, Raghava GPS, Lynn AM, Chandra N, Banerjee S, Raghunandanan MV, Pandey V, Taneja B, Yadav J, Dash D, Bhattacharya J, Misra A, Kumar A, Ramachandran S, Thomas Z, Brahmachari SK. Open source drug discovery– A new paradigm of collaborative research in tuberculosis drug development. Tuberculosis 2011;91(5):479–86. doi:10.1016/j.tube.2011.06.004.

[4] Årdal C, Røttingen J-A. Open source drug discovery in practice: a case study. PLoS Negl Trop Dis 2012;6(9):e1827. doi:10.1371/journal.pntd.0001827.

[5] Tan P-N, Steinbach M, Kumar V. Introduction to data mining. 1st editor. USA: Addison-Wesley Longman Inc; 2005.

[6] Rivera-Borroto OM, Marrero-Ponce Y, García-de la Vega JM, Grau-Ábalo RC. Comparison of combinatorial clustering methods on pharmacological data sets represented by machine learning-selected real molecular descriptors. J Chem Inf Model 2011;51:3036–49. doi:10.1021/ci2000083.

[7] Jolliffe I. Principal component analysis. 2nd editor. New York: Springer-Verlag; 2002.

[8] Lloyd S. Least squares quantization in PCM. IEEE Trans Inf Theory 1982;28:129–37. doi:10.1109/TIT.1982.1056489.

[9] M. Swain, MolVS:molecule Validation and Standardization, https://molvs.readthedocs.io/en/latest, 2019 (accessed July 2022).

[10] Moriwaki H, Tian Y-S, Kawashita N, Takagi T. Mordred: a molecular descriptor calculator. J Cheminformatics 2018;10:4. doi:10.1186/s13321-018-0258-y.

[11] Sutton C, Sindelar M, McCallum A. Feature bagging: preventing weight undertraining in structured discriminative learning. CIIR Tech Rep IR-402 2005:1–7.

[12] Rousseeuw PJ. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J Comput Appl Math 1987;20:53–65. doi:10.1016/0377-0427(87)90125-7.

[13] Dunn JC. Well-Separated Clusters and Optimal Fuzzy Partitions. J Cybern 1974;4:95–104. doi:10.1080/01969727408546059.

[14] Davies DL, Bouldin DW. A Cluster Separation Measure. IEEE Trans Pattern Anal Mach Intell 1979;1(2):224–7. doi:10.1109/TPAMI.1979.4766909.

[15] Calinski T, Harabasz J. A dendrite method for cluster analysis. Commun Statist 1974;3:1–27. doi:10.1080/03610927408827101.

[16] Cortes-Ciriano I. Benchmarking the predictive power of ligand efficiency indices in QSAR. J Chem Inf Model 2016;56:1576–87. doi:10.1021/acs.jcim.6b00136.

[17] Butina D. Unsupervised data base clustering based on daylight's fingerprint and Tanimoto similarity: a fast and automated way to cluster small and large data sets. J Chem Inf Comput Sci 1999;39:747–50. doi:10.1021/ci9803381.

[18] Greenacre M, Primicerio P. Multivariate analysis of ecological data. BBVA Foundation; 2013.

[19] Rational Discovery LLC, RDKit: open-Source Cheminformatics and Machine Learning Software, Open-Source Cheminformatics and Machine Learning, http://www.rdkit.org, 2006, (accessed June 2022).

[20] Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat Methods 2020;17:261–72. doi:10.1038/s41592-019-0686-2.

[21] McInnes L, Healy J, Melville J. UMAP: uniform manifold approximation and projection. J Open Source Software 2018;3(29):861. doi:10.21105/joss.00861.

[22] Allaoui M, Kherfi ML, Cheriet A. Considerably improving clustering algorithms using UMAP dimensionality reduction technique: a comparative study. In: El Moataz A, Mammass D, Mansouri A, Nouboud F, editors. Image Signal Process. Cham, Germany: Springer International Publishing; 2020. p. 317–25. ISBN: 978-3-030-51935-3.

[23] Reynolds D. Gaussian Mixture Models. In: Li SZ, Jain A, editors. Encyclopedia of biometrics. Boston, MA: Springer US; 2009. p. 659–63. doi:10.1007/978-0-387-73003-5_196.

[24] Evangelidis GD, Kounades-Bastian D, Horaud R, Psarakis EZ. A generative model for the joint registration of multiple point sets. In: Comput. Vis. – ECCV 2014. Lecture Notes in Computer Science, 8695. Cham, Germany: Springer International Publishing; 2014. p. 109–22. doi:10.1007/978-3-319-10584-0_8.

[25] Hall LH, Kier LB. The Electrotopological State: an Atom Index for QSAR. J Chem Inf Comput Sci 1995;35:1039–45. doi:10.1021/ci00028a014.

[26] Morgan HL. The generation of a unique machine description for chemical structures—a technique developed at chemical abstracts service. J Chem Doc 1965;5:107–13. doi:10.1021/c160017a018.

[27] Truchon JF, Bayly CI. Evaluating virtual screening methods: good and bad metrics for the "early recognition" problem. J Chem Inf Model 2007;47(2):488–508. doi:10.1021/ci600426e.

[28] Li D, Jiang K, Teng D, Wu Z, Li W, Tang Y, Wang R, Liu G. Discovery of New Estrogen-related receptor α agonists via a combination strategy based on shape screening and ensemble docking. J Chem Inf Model 2022;62(3):486–97. doi:10.1021/acs.jcim.1c00662.

[29] Kaplan AL, Strachan RT, Braz JM, Craik V, Slocum S, Mangano T, Amabo V, O'Donnell H, Lak P, Basbaum AI, Roth BL, Shoichet BK. Structure-based design of a chemical probe set for the 5-HT5A serotonin receptor. J Med Chem 2022;65(5):4201–17. doi:10.1021/acs.jmedchem.1c02031.

[30] Mysinger MM, Carchia M, Irwin JJ, Shoichet BK. Directory of useful decoys, enhanced (DUD-E): better ligands and decoys for better benchmarking. J Med Chem 2012;55(14):6582–94. doi:10.1021/jm300687e.

[31] Stein RM, Yang Y, Balius TE, O'Meara MJ, Lyu J, Young J, Tang K, Shoichet BK, Irwin JJ. Property-unmatched decoys in docking benchmarks. J Chem Inf Model 2021;61(2):699–714. doi:10.1021/acs.jcim.0c00598.

[32] Imrie F, Bradley AR, Deane CM. Generating property-matched decoy molecules using deep learning. Bioinformatics 2021;37(15):2134–41. doi:10.1093/bioinformatics/btab080.

[33] Wang L, Pang X, Li Y, Zhang Z, Tan W. RADER: a RApid DEcoy Retriever to facilitate decoy based assessment of virtual screening. Bioinformatics 2017;33(8):1235–7. doi:10.1093/bioinformatics/btw783.

[34] Cereto-Massagué A, Guasch L, Valls C, Mulero M, Pujadas G, Garcia-Vallvé S. DecoyFinder: an easy-to-use python GUI application for building target-specific decoy sets. Bioinformatics 2012;28(12):1661–2. doi:10.1093/bioinformatics/bts249.

[35] Nicholls A. What do we know and when do we know it? J Comput Aided Mol Des 2008;22:239–55. doi:10.1007/s10822-008-9170-2.

[36] Irwin JJ. Community benchmarks for virtual screening. J Comput Aided Mol Des 2008;22:193–9. doi:10.1007/s10822-008-9189-4.

[37] O'Boyle NM, Banck M, James CA, Morley C, Vandermeersch T, Hutchison GR. Open babel: an open chemical toolbox. J Cheminform 2011;3:33. doi:10.1186/1758-2946-3-33.

[38] Vogel SM, Bauer MR, Boeckler FM. DEKOIS: demanding evaluation kits for objective in silico screening - a versatile tool for benchmarking docking programs and scoring functions. J Chem Inf Model 2011;51(10):2650–65. doi:10.1021/ci2001549.

[39] Yang Y, Chen H, Nilsson I, Muresan S, Engkvist O. Investigation of the relationship between topology and selectivity for druglike molecules. J Med Chem 2010;53(21):7709–14. doi:10.1021/jm1008456.

[40] Bemis GW, Murcko MA. The properties of known drugs. 1. Molecular frameworks. J Med Chem 1996;39:2887–93. doi:10.1021/jm9602928.

[41] Hopkins AL, Groom CR. Target analysis: a priori assessment of druggability. Small molecule — protein interactions. ernst schering research foundation workshop. Waldmann H, Koppitz M, editors, Berlin, Heidelberg: Springer; 2003. vol 42. doi:10.1007/978-3-662-05314-0_2.

[42] Gaulton A, Hersey A, Nowotka M, Bento AP, Chambers J, Mendez D, Mutowo P, Atkinson F, Bellis LJ, Cibrián-Uhalte E, Davies M, Dedman N, Karlsson A, Magariños MP, Overington JP, Papadatos G, Smit I, Leach AR. The ChEMBL database in 2017. Nucleic Acids Res 2017;45(D1):D945–54. doi:10.1093/nar/gkw1074.

[43] Hatos A, Hajdu-Soltész B, Monzon AM, Palopoli N, Álvarez L, Aykac-Fas B, Bassot C, Benítez GI, Bevilacqua M, Chasapi A, Chemes L, Davey NE, Davidović R, Dunker AK, Elofsson A, Gobeill J, Foutel N, Sudha G, Guharoy M, Horvath T, Piovesan D. DisProt: intrinsic protein disorder annotation in 2020. Nucleic Acids Res 2020;48(D1):D269–76. doi:10.1093/nar/gkz975.

[44] Schmidtke P, Barril X. Understanding and predicting druggability. A high-throughput method for detection of drug binding sites. J Med Chem 2010;53(15):5858–67. doi:10.1021/jm100574m.

[45] Sael L, Li B, La D, Fang Y, Ramani K, Rustamov R, Kihara D. Fast protein tertiary structure retrieval based on global surface shape similarity. Proteins 2008;72:1259–73. doi:10.1002/prot.22030.

[46] Dong J, Yao ZJ, Zhang L, Luo F, Lin Q, Lu AP, Chen AF, Cao DSJ. PyBioMed: a python library for various molecular representations of chemicals, proteins and DNAs and their interactions. Cheminform 2018;10:16. doi:10.1186/s13321-018-0270-2.

[47] Sitzmann M, Ihlenfeldt WD, Nicklaus MC. Tautomerism in large databases. J Comput Aided Mol Des 2010;24:521–51. doi:10.1007/s10822-010-9346-4.

[48] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay É. Scikit-learn: machine Learning in Python. J Mach Learn Res 2011;12:2825–30. doi:10.5555/1953048.2078195.

[49] Alberca LN, Chuguransky SR, Álvarez CL, Talevi A, Salas-Sarduy E. In silico guided drug repurposing: discovery of new competitive and non-competitive inhibitors of Falcipain-2. Front Chem 2019;7:534. doi:10.3389/fchem.2019.00534.