AIMSim: An Accessible Cheminformatics Platform for Similarity Operations on Chemicals Datasets

Himaghna Bhattacharjee[a,b,c], Jackson Burns[a,c], Dionisios G. Vlachos*[a,b]

[a]Department of Chemical and Biomolecular Engineering, University of Delaware, 150 Academy St., Newark, DE 19711

[b]Catalysis Center for Energy Innovation, RAPID Manufacturing Institute, and Delaware Energy Institute, 221 Academy St., Newark, DE 19716

[c]Equal contribution

*Corresponding author: vlachos@udel.edu

Abstract

The recent advances in deep learning, generative modeling, and statistical learning have ushered in a renewed interest in traditional cheminformatics tools and methods. Quantifying molecular similarity is essential in molecular generative modeling, exploratory molecular synthesis campaigns, and drug-discovery applications to assess how new molecules differ from existing ones. Most tools target advanced users and lack general implementations accessible to the larger community. In this work, we introduce *Artificial Intelligence Molecular Similarity (AIMSim)*, an accessible cheminformatics platform for performing similarity operations on collections of molecules (molecular datasets). *AIMSim* provides a unified platform to perform similarity-based tasks on molecular datasets, such as diversity quantification, outlier and novelty analysis, clustering, and inter-molecular comparisons. *AIMSim* implements all major binary similarity metrics and molecular fingerprints and is provided as a Python package that includes support for command-line use as well as a fully functional Graphical User Interface for code-free utilization.

Keywords

Cheminformatics, molecular fingerprints, similarity, data visualization, open-source software

Introduction

Quantifying molecular similarity in datasets of molecules is a subject of immense importance in machine learning [1-5], synthetic chemistry [6], and cheminformatics [7-11] at large. Decades of research in developing suitable approaches have yielded various descriptors for molecules, particularly fingerprints, and dozens of metrics for comparisons. Molecular representations and similarity metrics are covered at length below with case studies in the fields mentioned above.

Tools to implement molecular descriptors and similarity measures exist but are fragmented, difficult to use, often accessible only from the command line, and require substantial coding by the end-user. For example, *RDKit* [12], the ubiquitous Python cheminformatics package, makes many tools available through the Boost library, requiring the end-user to work with C data types at intermediate stages. *mordred,* a cheminformatics tool with more than 1,000 molecular descriptors in a single package, has a separate command line or scripting interface [13]. *ccbmlib* [14] implements several common molecular fingerprints but is quite limited regarding implemented descriptors and broader functionalities. *chemfp* performs similarity searching remarkably fast on user-specified fingerprints but is command-line only and not fully open-source [15]. Similarity metrics are available throughout numerous Python packages, some in cheminformatics and others in statistics or machine learning packages.

Here we introduce *AIMSim* to make common cheminformatics tools available to end-users with no code required. Morgan [16], Daylight [17], and RDKit's [12] topological fingerprints are implemented. Additionally, we provide support for all molecular descriptors implemented in the *mordred* [13], *PaDEL-Descriptor* [18], and *ccbmlib* [14] software packages, including hashed fingerprints and simple scalar descriptors. We implement 47 similarity measures, including all standard metrics used in cheminformatics, such as the Tanimoto similarity [19, 20]. Finally, we explore applications of *AIMSim* to homogeneous catalysis, organic chemistry, and machine learning with case studies.

Background and Theory

Molecular Fingerprints

Molecular fingerprints are an essential tool in cheminformatics for representing molecules in various fields, from virtual high throughput experimentation [8] to drug discovery [9]. They have been the go-to for decades, known for their ease of use thanks to minimal setup yet broad flexibility [3]. Since their inception with the Daylight fingerprint in the mid-20th century [17], many molecular fingerprints have been created and refined into substructure-based and atom-pair fingerprints. This division results primarily from the performance of the former on large molecules. Atom-pair fingerprints are far more effective on large molecules, such as peptides, but ineffective on small molecules common in pharmaceutical development. Capecchi et al. [9] have recently defined a universal fingerprint; yet, many workflows depend on a specific fingerprint, and systems are built around the existing technology. For this reason, *AIMSim* allows the user to select from any of the commonly known molecular fingerprints through a single interface. For further specialized use cases, *AIMSim* also implements all molecular descriptors available through the *mordred* [13], *PaDEL-Descriptor* [18], and *ccbmlib* [14] software packages. These are considered "experimental descriptors" since their implementation and maintenance are not part of *AIMSim*.

Similarity and Distance Metrics

Similarity measures have been used widely to quantify the structural similarity of molecules [21-23]. The importance stems from the Similar Property Principle [24], which states that structurally similar molecules are likely to have similar properties or quantities of interest (QoI). As a result, a wide range of similarity measures is used for virtual screening, diversity quantification, and clustering [7, 25-33]. Formally, a similarity measure between two vectors is a function:

$$R^n \times R^n \rightarrow R$$

Additionally, we constrain the similarity measures in the range [0, 1] by a linear transformation, if necessary. 0 denotes minimum similarity and 1 identity.

Distances

Unlike similarity, distance quantifies dissimilarity. A distance metric is helpful for clustering data, diversity quantification, etc. All *AIMSim* similarities are scaled to [0, 1] and are converted to a distance metric using the linear transformation (the only exceptions being cosine and dice distances that are converted to their analogous metric distances. The formulae can be found in the SI, Table S1):

$$Distance = 1 - Similarity.\ Eq.3$$

For a true distance metric, a mapping $d(\boldsymbol{x}, \boldsymbol{y})$:

$$R^n \times R^n \rightarrow R.\ Eq.4$$

must satisfy the following criteria:

1. $d(\boldsymbol{x}, \boldsymbol{y}) = 0 \Leftrightarrow \boldsymbol{x} = \boldsymbol{y}$ (Identity of indiscernibles)
2. $d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x})$ (Symmetry)
3. $d(\boldsymbol{x}, \boldsymbol{z}) \leq d(\boldsymbol{x}, \boldsymbol{y}) + d(\boldsymbol{y}, \boldsymbol{z})$ (Triangle inequality)

Generally, distances derived from asymmetric similarity measures do not satisfy criteria 1 and 2 [30] and are non-metric or semi-metric. Clustering can only be carried out using metric similarity measures. This is enforced in *AIMSim*. The non-metric cosine similarity measure is an exception, which is converted to a metric angular distance using a linear transformation (*vide infra*).

Choosing the Appropriate Metrics

Different features may be generated for a molecule based on the descriptors or fingerprinting algorithms and weighting schemes. Pairing the correct fingerprint or molecular descriptor with a similarity measure is essential. For a given QoI, some featurization and similarity measures are more appropriate [34-36]. We propose a simple algorithm to accomplish this for a QoI.

Step 1: Select an arbitrary featurization scheme (fingerprint).

Step 2: Featurize the molecule set using the selected scheme.

Step 3: Choose an arbitrary similarity measure.

Step 4: Select each molecule's nearest and furthest neighbors in the set using the similarity measure.

Step 5: Measure the correlation between a molecule's QoI and its nearest neighbor's QoI.

Step 6: Measure the correlation between a molecule's QoI and its further neighbor's QoI.

Step 7: Define a score that maximizes the value in Step 5 and minimizes the value in Step 6.

Step 8: Iterate Steps 1 – 7 to select the featurization scheme and similarity measure to maximize the result of Step 7.

The chemical intuition behind this algorithm is straightforward. It is desirable to choose a featurization scheme and a similarity measure such that a pair of "similar" molecules have "similar" (correlated) QoIs.

Mathematically, a featurization scheme and similarity measure implicitly define a feature space to embed molecules. Thus, we convert chemical entities into mathematical objects. The QoI should ideally be a function of the molecule's coordinates and a smooth and continuous function to make it amenable to machine learning methods or optimization. The algorithm proposed above empirically enforces the smoothness criteria:

$$f(x + \epsilon) \to f(x) \quad as \ \epsilon \to 0. \ Eq. 5$$

Here, $f()$ is the QoI function, x is the feature space embedding of a molecule, and $x + \epsilon$ becomes the feature space embedding of its neighbor as $\epsilon \to 0$. *AIMSim* selects the most appropriate fingerprint and similarity measure. This, and other functionalities, are discussed in the next section.

Overview of *AIMSim* Software Design

AIMSim is written in the widely used Python programming language. The implementation is modularized and written strictly in an Object-Oriented fashion. The software design is divided orthogonally into backend and frontend segments. The backend includes all the computational functionalities and data structures of the software. The functionalities themselves are arranged into abstractions called Tasks, which are discussed later. The backend also exposes several API (Application Programming Interface) methods to utilize the functionalities of *AIMSim* programmatically in the same way as a Python package.

The primary focus of *AIMSim* is making cheminformatics tools accessible. Therefore, the software has been designed as a stand-alone application with code-free design in mind. This is achieved by the frontend of the software that provides an intuitive Graphical User Interface (GUI) for launching any of the complex functionalities of *AIMSim* without a single line of code. A Task Manager module manages the connection between the front end and backend of *AIMSim* and abstracts away any low-level details from the user. The frontend design elements are described in this section and the backend functionalities in a later section.

Task Manager

All tasks are accessible as classes with modules accessible to perform subtasks and are available on the documentation page. However, for the stand-alone version of *AIMSim* (with GUI), a Task Manager class is provided for scheduling tasks and pre-verifying configuration settings before launching any intensive computations (Figure 1). This prevents the user from wasting time waiting for a set of data to run which would otherwise never complete.

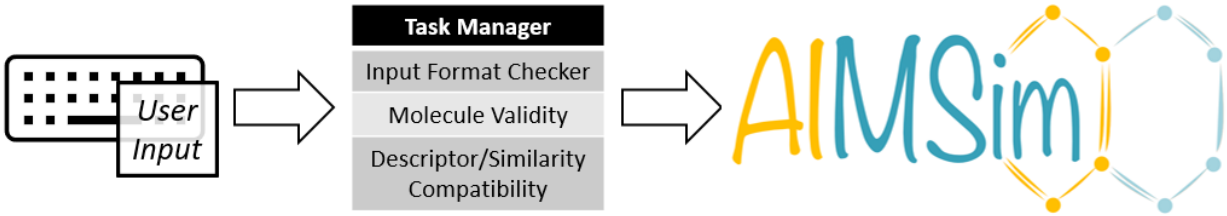

Figure 1. Flowchart of AIMSim's input verification and execution.

Graphical User Interface (GUI)

We provide a full-featured, easily installed, and launched GUI interface with key internal functions (Figure 2). Similarity measures and molecular descriptors are available in dropdown menus with commonly accessed configuration options accessible from toggles. This interface automatically generates configuration files for use by *AIMSim*. It allows the user to execute the files directly from the interface or open them in an external text editor for fine-grain changes and subsequent execution from the command line. The GUI can be run from a local installation of *AIMSim* using a single command to configure all dependencies automatically. The layout of *AIMSim,* including brief descriptions of the functionalities explored in the case studies, is shown in Figure 3.

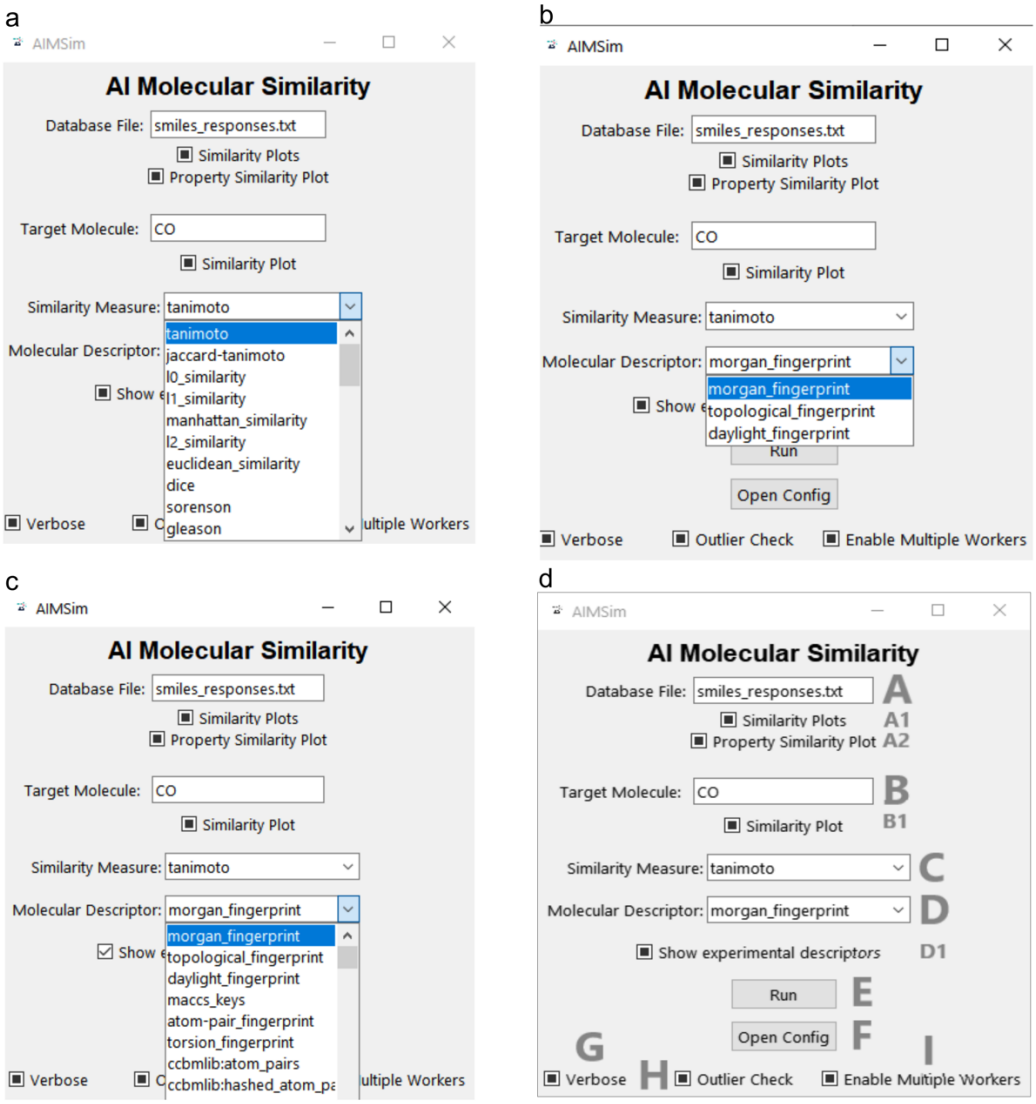

Figure 2. AIMSim Graphical User Interface (GUI). a) AIMSim implements almost 50 common similarity measures selected via a drop-down menu. b) AIMSim implements the Morgan, Daylight, and topological fingerprints. c) The "Show experimental descriptors" checkbox enables the user to select from a range of third-party fingerprints provided in the mordred [13], PaDEL-Descriptor [18], and ccbmlib [14] libraries. d) The complete GUI with all components labeled. A full walkthrough of all components is included in the SI.

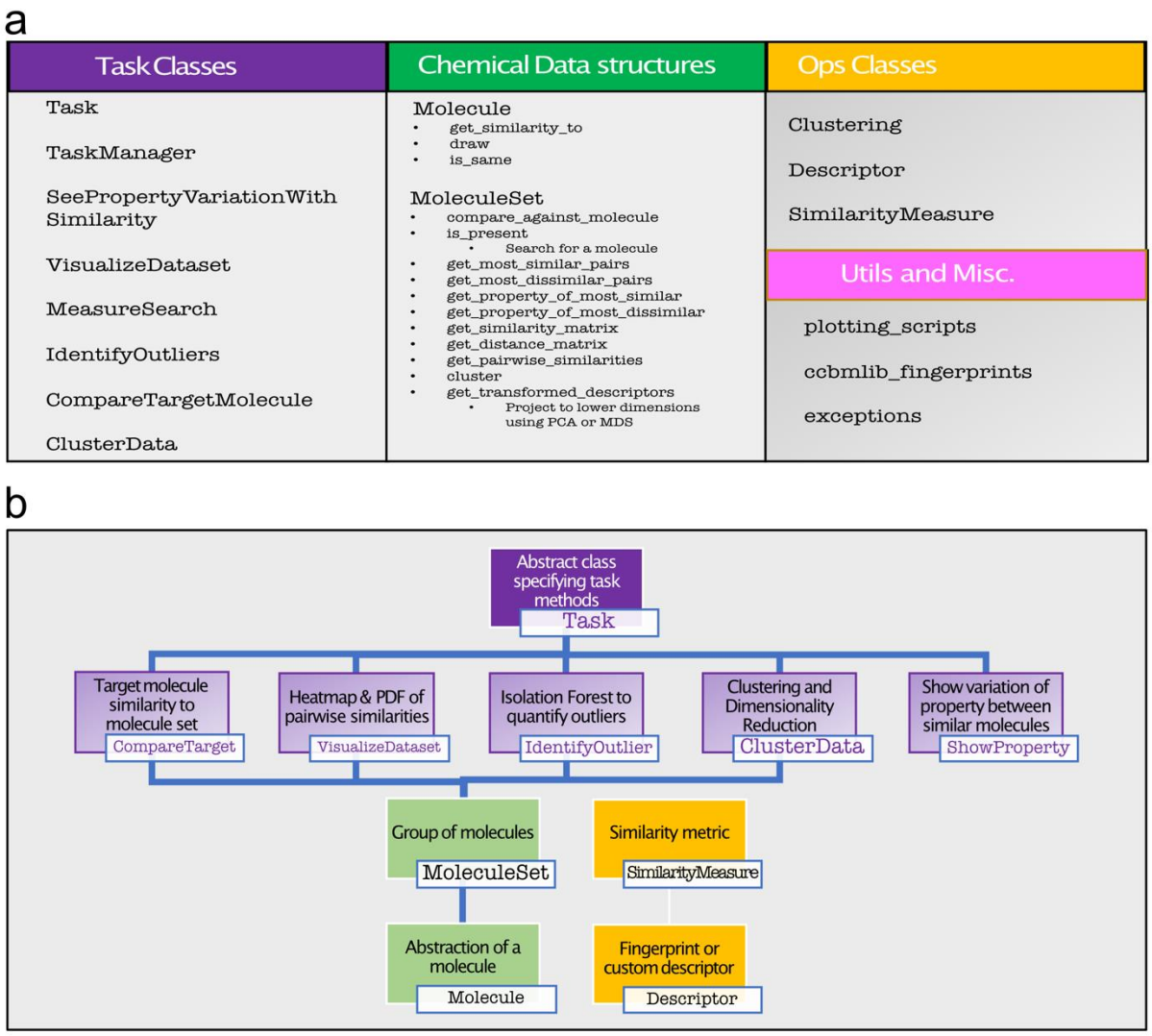

Figure 3. a) Main classes and methods of AIMSim. B) Core structure of AIMSim with brief description of the classes.

Overview of Functionalities

AIMSim implements many functionalities and can generate similarity density distributions and pairwise heatmaps simply using a list of SMILES strings. One can use the forty-plus distance metrics and common molecular fingerprints, including the Morgan [16] and Daylight specifications [17]. Dimensionality reduction can be performed to visualize high dimensional fingerprints in 2 dimensions. At the implementation level, multithreading and configurable output levels facilitate the analysis of large datasets, as discussed at length in the Use Case #3. The rest of this section is an overview of the various capabilities of *AIMSim*.

Accepted Input Formats

AIMSim can ingest many common cheminformatics file types. Protein Data Bank (.pdb), SMILES strings (.txt, .SMILES, or .smi), Excel workbooks (.xlsx), and Comma Separated Values (.csv) are all directly supported. Directories containing a collection of these datatypes can also be parsed directly in *AIMSim* without the need for file aggregation or repeat execution.

Tasks

At the core of *AIMSim*'s functionalities are Task objects to encapsulate the operations run over chemical datasets. They are described below, along with references to case studies for illustration.

Task: Measure Search

Virtual screening or exploratory synthesis campaigns may not have a rich body of heuristics or benchmark studies for new applications. Then, it becomes necessary to pair a fingerprint to a similarity measure (this choice is called "a measure" here for brevity). *AIMSim* implements the algorithm discussed above for measure search via the MeasureSearch Task class. Using a user-specified random subsample of the data (due to the computational load of this task), MeasureSearch scores the measures based on the degree of correlation in the QoI properties between molecules and their nearest and furthest neighbors in the space defined by the measure. The scoring is done using a user-specified strategy to maximize correlation in QoI properties between nearest neighbors (max strategy), minimize absolute correlation in QoI properties between furthest neighbors (min strategy) or place an equal weight combination of the two strategies (max-min strategy). In the min strategy, we reduce the absolute value of the correlation and not the correlation itself (the property of a molecule should be uncorrelated from its furthest neighbor; minimizing the correlation drives the search towards anti-correlation (-1) instead of 0. A bar graph enumerating the score and neighbor correlations for the top n (user-specified) measures can be displayed. Alternatively, the top measure can be programmatically extracted (for module-level usage). This Task is automatically launched when the measure is set to 'determine' in the configuration file (when operating *AIMSim* as a stand-alone application). The user can constrain the optimization to specific fingerprints and similarity measures or to only metric distances (which can be used for clustering). Generally, we find that a combination of Morgan fingerprint and a Tanimoto similarity metric works reasonably well for a lot of use cases and is used for illustrative purposes in this work. It is recommended that measure search is used if this measure choice is not found to be satisfactory.

Task: See Property Variation with Similarity

Upon selecting a suitable measure, using the *AIMSim* Measure Search task or heuristics, one can verify the efficacy by quantifying the correlation with the nearest and furthest neighbors. The correlation with the nearest (furthest) neighbor properties should be close to 1 (0). *AIMSim* creates a parity plot with the Pearson correlation coefficient as a legend, as illustrated in Case Study #1.

Task: Visualize Dataset

At the beginning of machine learning or computational model-building, it is helpful to quantify the diversity in the training set or the outputs of a molecular generative model (Case Study #3) or for substrate scope verification (Case Study # 2). This task generates a heatmap and density plot of the pairwise similarity between molecules in a molecule set for exploratory analysis.

Dimensionality Reduction for Visualizing Molecule Set

After the clustering operation, *AIMSim* embeds the entire molecule set from the high dimensional space of fingerprints (typically ~1024 dimensions corresponding to the number of bits used for generating the fingerprint) to two dimensions. *AIMSim* currently implements multidimensional scaling [37-39], T-distributed Stochastic Neighbor Embedding (t-SNE) [40] and Principal Component Analysis (PCA) [41] for dimensionality reduction. Note that dimensionality reduction can only be done using similarity measures that yield a valid distance (metricity requirement).

Task: Compare Target Molecules to Molecule Set

Comparing a target molecule to a database is vital for locating molecules with similar or different properties and narrowing the candidates (leads or hits) to explore experimentally or computationally. For example, one may need to replace a top-performing but toxic solvent with a green one of similar properties. Conversely, it might be required to select the set of molecules most different from a query molecule. Such a use case is typical in designing a training set for a machine learning model, where one is interested in identifying molecules that enhance the diversity of the training set or making the model more robust and generalizable to unseen data. The task generates a pairwise similarity distribution quantifying the similarity between a target molecule and an entire molecule set, its most similar and dissimilar molecules, and the top "n" most similar and dissimilar molecules. Optionally, *AIMSim* also generates a structural representation of these molecules. An example is shown in Case Study #1.

Task: Cluster Data

Clustering is an unsupervised technique used to group similar molecules. Performing this analysis and visualizing the results yields deeper insight into patterns in the data and is often the first step. This task clusters a set of molecules based on structural similarities.

Clustering Algorithms

AIMSim implements two broad classes of clustering algorithms:
Hierarchical Agglomerative clustering [42]: The data points are clustered by grouping similar points in a hierarchical fashion. This is done by constructing coarse grouping and then subdividing the groupings into smaller sizes until the required number of clusters is obtained. There are several implementations of hierarchical clustering. *AIMSim* implements complete linkage, single linkage, and average linkage algorithms for binary fingerprints and Ward's algorithm for arbitrary vector descriptors using norm-based similarity metrics. *AIMSim* uses Agglomerative Clustering implementation of the scikit-learn package.

K-medoids [43]*:* The k-medoids is a partitioning algorithm that builds clusters of data points by minimizing the distance of each point to the median of their respective clusters. The k-medoids are implemented for arbitrary vector descriptors using norm-based similarity metrics. *AIMSim* uses the k-medoids implementation of scikit-learn-extra package [44, 45].

Clustering algorithms typically require inter-sample distances. Calculating this distance is an expensive O(n^2) operation in terms of dataset size. The similarity matrix is internally converted to a distance matrix using linear operations for the molecule set to avoid this computational cost. This distance matrix is used by the clustering algorithm. *AIMSim* automatically detects the clustering algorithm (non-Euclidean vs. Euclidean) based on the descriptors (binary fingerprints vs. arbitrary vector values). This task generates a json file containing the names of molecules in the different clusters. Additionally, if the

molecule set is initialized with molecular properties (QoI), it generates a plot of the distribution of molecular properties in the different clusters. This plot enables visual comparison of the efficacy of the clustering (ideally, a separation of the distributions in the molecular properties is desired).

Dimensionality Reduction for Visualizing Clusters

AIMSim generates another 2D embedding of the molecule set where the molecules are colored according to the cluster they belong to. This plot enables a visual inspection of the success of the clustering process. Clustering molecules based on structural similarity can be achieved if the similarity measure satisfies the metricity requirement (as discussed in the distance section above) and can yield a valid distance metric.

This task is illustrated in Case Study # 1.

Task: Outlier Detection

This task implements an isolation forest to identify outliers. Every molecule in a dataset is assigned a dissimilarity score: a value of 0 or below implies an outlier. Accessible from the user interface via a simple toggle, this task can provide a "sanity check" to avoid erroneous data and verify molecule additions not already represented in the data. The results can be written to the command line as visual output or saved to a file.

Automated Testing

Since *AIMSim* is an Open-Source project, community participation and contributions are encouraged through the GitHub project page. However, it is necessary to maintain the integrity and correctness of the codebase for reliable utilization by the community. Therefore, we have made available an extensive suite of automated tests. Only changes that successfully pass all these tests are incorporated into the main codebase. This ensures the health of the project. The tests can be found on the GitHub page and additional test suggestions are accepted and encouraged.

Case Study #1: Exploratory Solvent Search

An essential use of *AIMSim* is for catalyst discovery and solvent search. The latter case is illustrated here. The data is taken from Wang *et al.* [46]. The authors screened 2214 organic solvents for reactive extraction of HMF (5-hydroxylmethylfurfural), a platform chemical produced in the acid-catalyzed dehydration of hexoses, in biphasic organic-water systems. The authors obtained the log (water-candidate molecule) partition coefficient of HMF from the ADFCRS-2018 database using the ADF COSMO-RS software package.

It is essential to visualize the "information richness" of the dataset. A diverse set is preferred to maximize the information and avoid wasting time and resources investigating similar molecules of no practical interest. *AIMSim* provides *a priori* diversity identification. For this use case, we utilize the Morgan fingerprint (radius 3)[16] and the Tanimoto similarity measure. Figures 4a and 4b show the correlation in QoI (log(water-candidate molecule) partition coefficient of HMF). The high linear correlation in the responses of nearest neighbors (Pearson coefficient of 0.78, Figure 4a) and the low correlation of furthest neighbors (0.02, Figure 4b) illustrate that this measure works, i.e., molecules grouped as similar have correlated responses.

Figure 4c shows the 2-dimensional embedding of the dataset visualized by *AIMSim* using the MDS algorithm. Figure 4d shows the results of clustering this dataset using *AIMSim*'s hierarchical clustering functionality. Two distinct clusters are identified and can be separating out in the lower dimensional embedding generated by *AIMSim*. While these clusters separate out in low dimensions, note that these clusters are not immediately apparent from a visual inspection of the low dimensional embedding in Figure 4c. This is due to the high dimensional nature of fingerprints and highlights the importance of clustering. Plots generated by *AIMSim* showing number of molecules in each cluster and distribution of QoI for molecules in each of the clusters are shown in Figure S1.

Finally, we run a simulated example of a typical solvent search scenario. The authors of the work note that the phenolic group of the solvents lead to a higher HMF partition coefficient with water. Thus, solvents with phenolic groups should be better for reactive extraction of HMF. Thus, we run a target search of the simplest such solvent (phenol) against the dataset. In general, since the study was used for screening good solvents, we would expect that phenol to have a high degree of similarity with the dataset. In fact, phenol was studied by the authors. Using it as a query molecule is done for illustrative purposes to mimic the situation when a collection of molecules with favorable properties are known and a new molecule that resembles this molecule set is sought. In such a case, a high-throughput *in-silico* search over multiple candidates can be made using *AIMSim*. Only the SMILES string (or some other structural identifier) of the candidates is needed. As a control, we also run a target search using ammonia, a molecule which we expect to be very different from the solvents studied in the work.

Figure 4e shows the results of running a target search using ammonia. The pairwise similarity density is strongly peaked around 0 indicating that the solvents are all very different from ammonia. Figure 4f shows the results of running a target search using phenol. The pairwise similarity distribution now has a heavy tail away from zero indicating that the solvents in general are similar to phenol (with those in the extremes of the tail being very similar to phenol). Thus qualitatively, we can conclude that phenol is similar to the collection of solvent molecules. *AIMSim* also displays the structures of molecules which are most similar and least similar to the query. In Figure 4g and h, the structures for the most and least similar molecule to the target molecule (phenol) as generated by *AIMSim* are shown. The most similar molecule (Figure 4g) to phenol is of course phenol itself (since the authors had included phenol in their dataset) and the least similar molecule (Figure 4h) is a branched saturated fluoroalkane which is clearly very different from a phenolic compound.

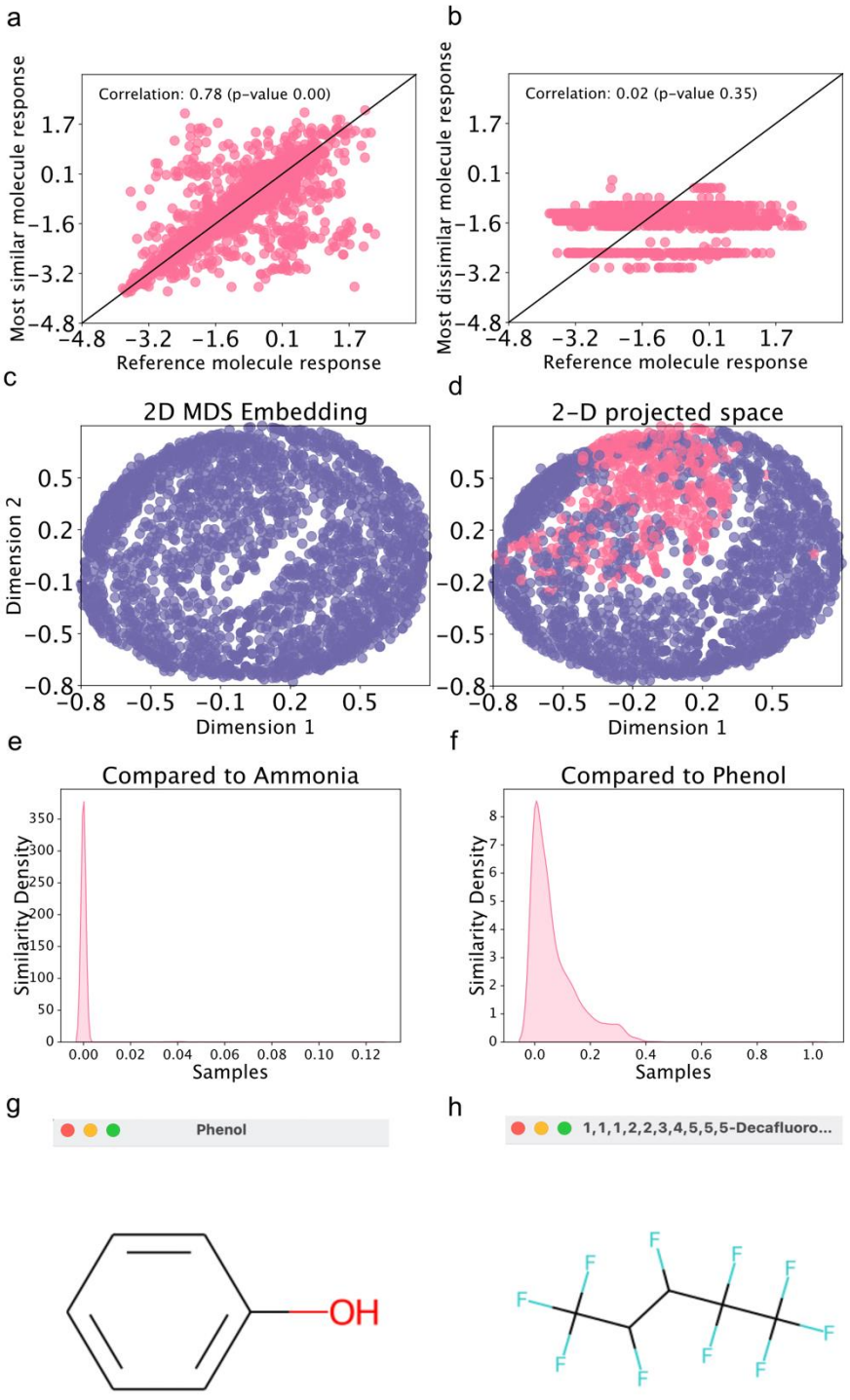

Figure 4. AIMSim analysis of solvents for biphasic extraction of 5-hydroxymethylfurfural. Data from [38]. a-b) Parity between log partition coefficient of (a) nearest neighbor and (b) furthest neighbor solvents using the Morgan fingerprint and Tanimoto similarity measure. c) Low dimensional embedding of the dataset using the MDS algorithm. D) Low dimensional embedding generated after the clustering using the complete linkage agglomerative hierarchical clustering algorithm. Two distinct clusters of molecules are shown. e-f) Target analysis of the dataset using AIMSim using e) N (ammonia) target SMILES. and f) phenol. g-h) Structure of the most similar (g) and most dissimilar (h) molecule to the target.

Case Study #2: Substrate Scope Diversity Verification

When proposing a novel reaction, it is essential to evaluate the transformation's tolerance of diverse functional groups and substrates [6]. This collection of molecules is conventionally referred to as the substrate scope, or more often, simply the scope. Using *AIMSim*, one can evaluate the structural and chemical similarity across an entire scope to ensure that it avoids redundant examples and is sufficiently diverse prior to experimentation to avoid unnecessary and expensive work. Using existing literature data paired with *AIMSim*, one can evaluate if a novel substrate not included in a given scope is similar to any substrates assessed.

Figure 5 is an example of a similarity heatmap and a distribution generated by *AIMSim* for published chemical data. The data is retrieved from Chen and coworkers' copper-catalyzed three-component sulfonamide synthesis [47]. In their work, an aryl- or alkenyl-boronic acid and a substituted amine were simultaneously coupled to a sulfone to yield the sulfonamide, essential for pharmaceuticals and agrochemicals. To evaluate the functional group tolerance and overall applicability of the proposed transformation, they created 104 products, each composed of a unique combination of amine and boronic acid coupling partners. These products are compared using the Rogers-Tanimoto distance and the Morgan fingerprint, the default configuration for *AIMSim*.

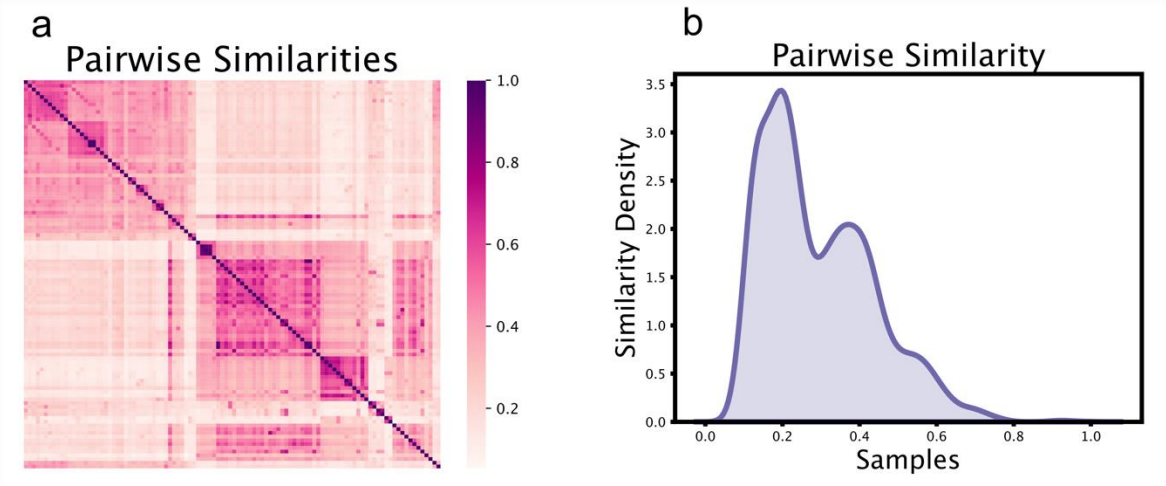

Figure 5. Similarity heatmap (a) and similarity distribution (b) for Chen's three-component sulfonamide synthesis substrate scope.

As shown in the heatmap, the prominent region of high similarity near the diagonal corresponds to substrates presented sequentially in the publication with only minor structural differences, such as a different aryl-methyl substitution pattern. In the bulk of the heatmap, and more obviously in the similarity distribution, most samples have a similarity of approximately 0.2-0.4. This matches expectations, as the substrate scope was constructed by allowing one partner in the coupling to vary at a time.

AIMSim can verify that an additional sample for this dataset would be sufficiently diverse to make it worth investigating. Shown schematically in Figure 6 and Figure 7, *AIMSim* can quickly identify which members of a given dataset already evaluated are most similar, indicating if the new one is unexplored. For the examples, the proposed additions include a variation on the aryl-halide substituent and an increase in the amine ring size, extensions provided in the original scope.

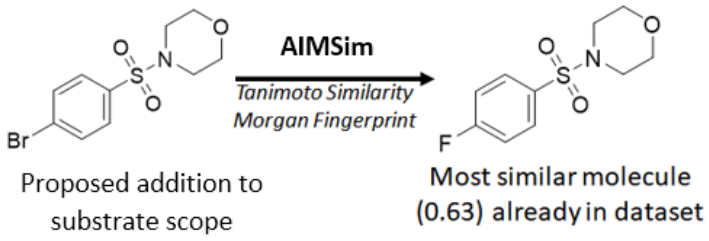

Figure 6. Proposed addition to the dataset and its most similar pre-existing example.

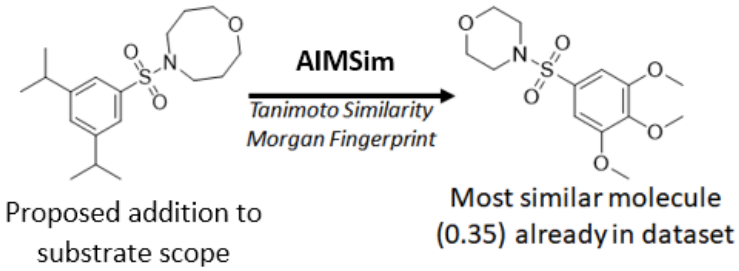

Figure 7. Substantially different potential addition to the scope and the most similar substrate already evaluated.

As shown above, *AIMSim* identifies another phenyl-halide functionalized substrate when queried with the bromide species. This is an obvious conclusion to a practicing chemist, though it can be supported with data and be tractable on too large datasets for human inspection. When presented with what seems to be a highly different species in Figure 7, *AIMSim* identifies a cluster of methoxy groups as the most similar. This unlikely pairing may stem from similar steric behavior by the two species. Still, given the low similarity score, it would be advisable to investigate it on the lab bench. The morpholine fragment, a common feature in the substrate scope, is identified as most similar to the 1,5-oxazocane, which matches expectations.

Case Study #3: Generational Library Diversity Verification

Generative Neural Networks (GNNs) have seen increasing use in virtual high throughput screening in the last few years for evaluating novel targets. Their general purpose is to 'create' new molecules digitally for subsequent evaluation via molecular docking or machine learning. For hypothetical molecules to be of any use, they must be sufficiently different from existing examples to explore unknown chemical space. Tools, such as *MOSES,* set out to quantify various performance metrics for generated molecule sets [1], including similarity score distributions. *AIMSim* extends this effort by providing a more readily accessible and human interpretable representation of molecular diversity, accessible through a GUI, while also including a richer feature set.

One dataset analyzed by *MOSES* was generated using the Hidden Markov Model, referred to as the HMM dataset. This collection includes more than 10,000 individual molecules represented as SMILES strings. *MOSES* reports various scalar descriptors for this dataset, such as the Validity and Uniqueness, and while these are informative, they are inherently reductive. *AIMSim* instead provides a distribution of similarity density based on comprehensive pairwise comparisons, as illustrated in Figure 8.

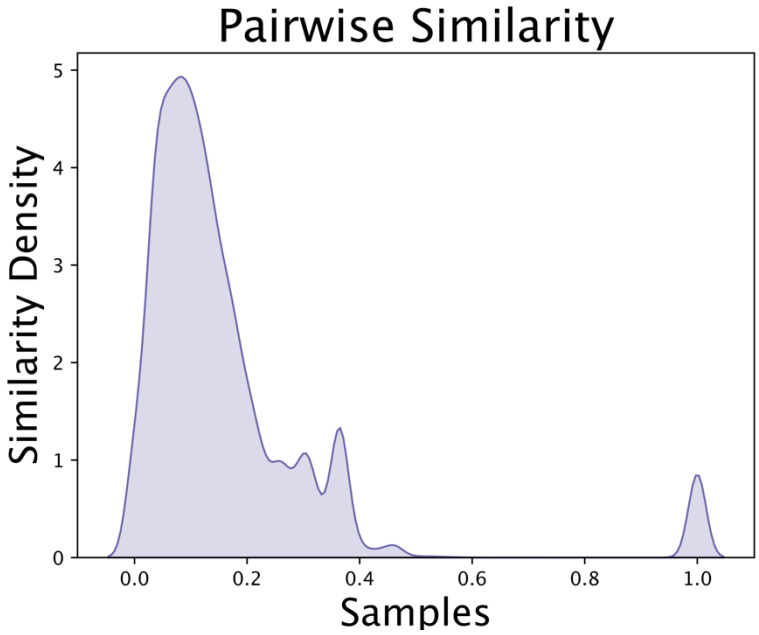

Figure 8. Similarity density for the HMM dataset provided by MOSES [1] and available in AIMSim.

It is now clearly visible that this generative model created a diverse dataset. There is a substantial area of the distribution with similarity below 0.2, i.e., most of the species share only 20% similarity to other examples in the dataset. This representation also reveals a large spike in similarity around 0.35 and a similar spike at 1 (perfect similarity). The former may be attributed to cases as those presented in Use Case #2 above, where one component of a larger substrate is being altered at a time. The peak at perfect similarity indicates that the generative model returns a non-zero number of identical or nearly identical molecules. *AIMSim* provides numerous molecular descriptors and similarity metrics, creating further avenues to ensure proper performance.

Multiprocessing

Execution time becomes a concern on datasets of this order of magnitude due to the underlying algorithm for comparisons being of O(n^2) complexity. To handle the large size of generative data sets, *AIMSim* implements multiprocessing and sampling techniques to reduce execution time. Using the *multiprocess* Python library [48], any number of processes can be spawned to divide the task of comparing molecules. Because much of the execution time is spent on performing molecular comparisons, the continued addition of processes is highly efficient with speedup in excess of 90% on datasets of 500 or more molecules, as shown in Figure 9. Table S2 includes more extensive testing. All molecules are retrieved from the combinatorial dataset of similar size to generative datasets, provided as part of *MOSES* and reproduced in *AIMSim*.

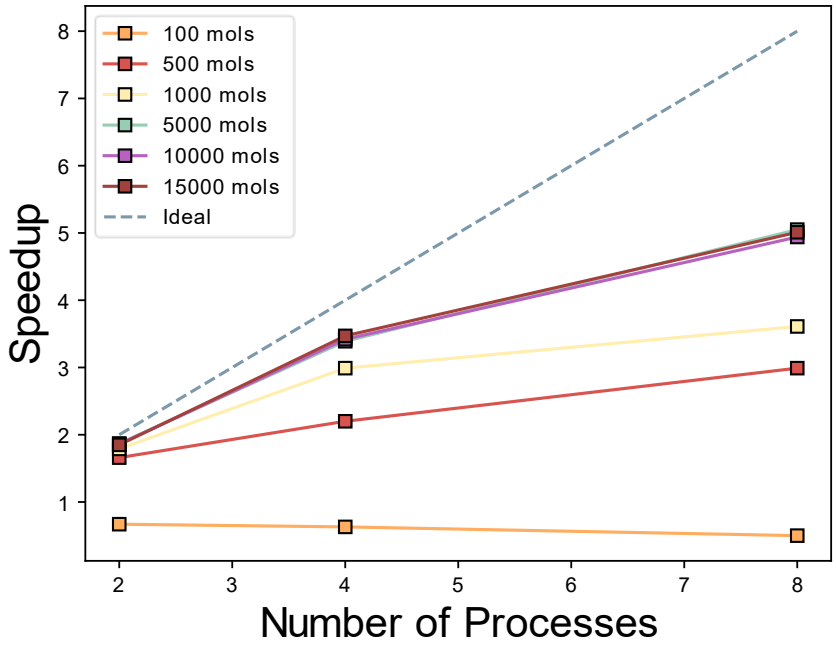

Figure 9. Speedup efficiency for molecular sets of different sizes.

With sufficiently sized datasets, the costs associated with multiprocessing and performance gains become substantial. Minimal efficiency and speedup on small datasets are expected due to the computational expense of spawning and joining processes. Warnings are included in the documentation to prevent multiprocessing on unsuited datasets, and *AIMSim* includes an automatic configuration option which uses the heuristics above to estimate if multiprocessing will result in faster execution and then configure itself accordingly.

Conclusions

AIMSim is a completely open-source cheminformatics software designed for code-free utilization as well as a Python package for more specialized programmatic usage. From a user-friendly graphical user interface, *AIMSim* can calculate similarity density distributions, similarity heatmaps, single-molecule database comparisons, and dimensionality reductions to facilitate research in various fields. Nearly 50 common distance metrics used in cheminformatics are available via *AIMSim*, as well as a host of molecular fingerprints and descriptors. *AIMSim's* full parallelization, supporting multiprocessing capabilities, greatly boosts performance. Thus, *AIMSim* can tractably analyze large-scale datasets typical in machine learning applications. With speedups greater than 90% on datasets with 500 or more molecules, the diversity of generative models can be verified on a molecule-by-molecule basis.

Code Availability

AIMSim is freely available on its GitHub project page (https://github.com/VlachosGroup/AIMSim) along with detailed user documentation (https://vlachosgroup.github.io/AIMSim/). A limited version of *AIMSim* can also be run in a browser. Details can be found on the GitHub page.

Acknowledgements

The authors would like to acknowledge the RAPID manufacturing institute, supported by the Department of Energy (DOE) Advanced Manufacturing Office (AMO), Award Number DE-EE0007888-9.5. RAPID projects at the University of Delaware are also made possible by funding provided by the State of Delaware. The Delaware Energy Institute gratefully acknowledges the support and partnership of the State of Delaware in furthering the essential scientific research conducted through the RAPID projects. The authors would also like to acknowledge Kelly Walker for the design of the logo.

References

1. Polykovskiy, D., et al., *Molecular Sets (MOSES): A Benchmarking Platform for Molecular Generation Models.* arXiv:1811.12823 [cs, stat], 2020.
2. Janet, J.P., et al., *A quantitative uncertainty metric controls error in neural network-driven chemical discovery.* Chemical Science, 2019. **10**(34): p. 7913-7922.
3. Padula, D., J.D. Simpson, and A. Troisi, *Combining electronic and structural features in machine learning models to predict organic solar cells properties.* Materials Horizons, 2019. **6**(2): p. 343-349.
4. Bhattacharjee, H. and D.G. Vlachos, *Thermochemical Data Fusion Using Graph Representation Learning.* Journal of Chemical Information and Modeling, 2020. **60**(10): p. 4673-4683.
5. Bhattacharjee, H., N. Anesiadis, and D.G. Vlachos, *Regularized machine learning on molecular graph model explains systematic error in DFT enthalpies.* Scientific Reports, 2021. **11**(1): p. 14372.
6. Collins, K.D. and F. Glorius, *A robustness screen for the rapid assessment of chemical reactions.* Nature Chemistry, 2013. **5**(7): p. 597-601.
7. Cereto-Massagué, A., et al., *Molecular fingerprint similarity search in virtual screening.* Methods, 2015. **71**: p. 58-63.
8. Muegge, I. and P. Mukherjee, *An overview of molecular fingerprint similarity search in virtual screening.* Expert Opinion on Drug Discovery, 2016. **11**(2): p. 137-148.
9. Capecchi, A., D. Probst, and J.-L. Reymond, *One molecular fingerprint to rule them all: drugs, biomolecules, and the metabolome.* Journal of Cheminformatics, 2020. **12**(1): p. 43.
10. Golbraikh, A., *Molecular Dataset Diversity Indices and Their Applications to Comparison of Chemical Databases and QSAR Analysis.* Journal of Chemical Information and Computer Sciences, 2000. **40**(2): p. 414-425.
11. Sliwoski, G., et al., *Computational methods in drug discovery.* Pharmacological reviews, 2013. **66**(1): p. 334-395.
12. Landrum, G., *RDKit: Open-source cheminformatics*. 2011: http://www.rdkit.org.
13. Moriwaki, H., et al., *Mordred: a molecular descriptor calculator.* Journal of Cheminformatics, 2018. **10**(1): p. 4.
14. Vogt, M. and J. Bajorath, *ccbmlib – a Python package for modeling Tanimoto similarity value distributions.* F1000Research, 2020. **9**: p. Chem Inf Sci-100.
15. Dalke, A., *The chemfp project.* Journal of Cheminformatics, 2019. **11**(1): p. 76.
16. Rogers, M.H.D., *Extended-connectivity fingerprints.* J Chem Inf Model, 2010. **50**(5): p. 742-54.
17. *Daylight Theory: Fingerprints*.
18. Yap, C.W., *PaDEL-descriptor: An open source software to calculate molecular descriptors and fingerprints.* Journal of Computational Chemistry, 2011. **32**(7): p. 1466-1474.
19. Jaccard, P., *THE DISTRIBUTION OF THE FLORA IN THE ALPINE ZONE.1.* New Phytologist, 1912. **11**(2): p. 37-50.

20. Rogers, D.J. and T.T. Tanimoto, *A Computer Program for Classifying Plants.* Science, 1960. **132**(3434): p. 1115-8.
21. Kubinyi, H., *Similarity and dissimilarity: a medicinal chemist's view.* Perspect Drug Discov Des, 1998. **9-11**: p. 225-52.
22. Bender, R.G.A., *Molecular similarity: a key technique in molecular informatics.* Org Biomol Chem, 2004. **2**(22): p. 3204-18.
23. M Vogt, J.B.G.M., D Stumpfe, *Molecular similarity in medicinal chemistry.* J Med Chem, 2014. **57**(8): p. 3186-204.
24. eds., M.A.J.a.G.M.M., *Concepts and applications of molecular similarity*. 1990, Nashville, TN: John Wiley & Sons.
25. Chen, C.R.X., *Performance of similarity measures in 2D fragment-based similarity searching: comparison of structural descriptors and similarity coefficients.* J Chem Inf Comput Sci, 2002. **42**(6): p. 1407-14.
26. J Holliday, P.W.N.S., *Combination of fingerprint-based similarity coefficients using data fusion.* J Chem Inf Comput Sci, 2003. **43**(2): p. 435-42.
27. N Salim, P.W.J.H., M Whittle, *Analysis and display of the size dependence of chemical similarity coefficients.* J Chem Inf Comput Sci, 2003. **43**(3): p. 819-28.
28. VJ Gillet, J.L.M.W., P Willett, A Alex, *Enhancing the effectiveness of virtual screening by fusing nearest neighbor lists: a comparison of similarity coefficients.* J Chem Inf Comput Sci, 2004. **44**(5): p. 1840-8.
29. Willett, P., *Similarity-based virtual screening using 2D fingerprints.* Drug Discov Today, 2006. **11**(23-24): p. 1046-53.
30. Todeschini, R., et al., *Similarity Coefficients for Binary Chemoinformatics Data: Overview and Extended Comparison Using Simulated and Real Data Sets.* Journal of Chemical Information and Modeling, 2012. **52**(11): p. 2884-2901.
31. V Consonni, P.W.R.T., H Xiang, J Holliday, M Buscema, *Similarity coefficients for binary chemoinformatics data: overview and extended comparison using simulated and real data sets.* J Chem Inf Model, 2012. **52**(11): p. 2884-901.
32. Willett, P., *Combination of similarity rankings using data fusion.* J Chem Inf Model, 2013. **53**(1): p. 1-10.
33. X Zhang, P.S.F.R., D Gabriel, *Benchmarking of multivariate similarity measures for high-content screening fingerprints in phenotypic drug discovery.* J Biomol Screen, 2013. **18**(10): p. 1284-97.
34. Chen, X. and C.H. Reynolds, *Performance of Similarity Measures in 2D Fragment-Based Similarity Searching: Comparison of Structural Descriptors and Similarity Coefficients.* Journal of Chemical Information and Computer Sciences, 2002. **42**(6): p. 1407-1414.
35. Bender, A. and R.C. Glen, *Molecular similarity: a key technique in molecular informatics.* Organic & Biomolecular Chemistry, 2004. **2**(22): p. 3204-3218.
36. Whittle, M., et al., *Enhancing the Effectiveness of Virtual Screening by Fusing Nearest Neighbor Lists: A Comparison of Similarity Coefficients.* Journal of Chemical Information and Computer Sciences, 2004. **44**(5): p. 1840-1848.
37. Borg, I. and P. Groenen, *Modern Multidimensional Scaling: Theory and Applications (Springer Series in Statistics)*. 2005.
38. Kruskal, J., *Nonmetric multidimensional scaling: A numerical method.* Psychometrika, 1964. **29**(2): p. 115-129.
39. Kruskal, J., *Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis.* Psychometrika, 1964. **29**: p. 1-27.
40. van der Maaten, L. and G. Hinton, *Viualizing data using t-SNE.* Journal of Machine Learning Research, 2008. **9**: p. 2579-2605.

41. Anzai, Y., *Pattern recognition and machine learning*. 2012: Elsevier.
42. Murtagh, F. and P. Contreras, *Algorithms for hierarchical clustering: an overview.* WIREs Data Mining and Knowledge Discovery, 2012. **2**(1): p. 86-97.
43. Lazic, S., *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd edn*. 2008, Springer.
44. Maranzana, F.E., *On the location of supply points to minimize transportation costs.* IBM Syst. J., 1963. **2**(2): p. 129–135.
45. Park, H.-S. and C.-H. Jun, *A simple and fast algorithm for K-medoids clustering.* Expert Systems with Applications, 2009. **36**(2, Part 2): p. 3336-3341.
46. Lu, Y., et al., *Identifying the Geometric Site Dependence of Spinel Oxides for the Electrooxidation of 5-Hydroxymethylfurfural.* Angewandte Chemie International Edition, 2020. **59**(43): p. 19215-19221.
47. Chen, Y., et al., *Direct Copper-Catalyzed Three-Component Synthesis of Sulfonamides.* Journal of the American Chemical Society, 2018. **140**(28): p. 8781-8787.
48. McKerns, M.M., et al., *Building a framework for predictive science.* arXiv preprint arXiv:1202.1056, 2012.