Reducing Virtual Screening time through selective compound clustering: A cluster algorithm comparison

Suarez Vasquez, Michael

Friday 2nd March, 2018

Abstract

Virtual Scanning run time is simplified by defining a molecular compound space using a similarity metric and reduced by comparing the three different clustering algorithms k-means, HDBSCAN and Markov Stability. They are compared on their implementation ease, time and complexity performance and their prediction ability is tested by comparing the representative strength of clustroids (cluster centres) to their respective clusters using experimental molecule-protein inhibition data. A trade-off between predication rate accuracy and reduction of compound space through the clustroids is identified and a combination of algorithms is suggested for best performance. While difficult to implement Markov Stability shows the most consistent performance overall.

1 Introduction

1.1 Virtual Screening

Virtual Screening (VS) [1] is the leading large scale computational method of iterating through molecular libraries to find binding partners for proteins or enzymes. It is used in drug discovery to reduce the experimental cost and time needed to probe all combinations in real life. Its effectiveness and efficiency is determined by the filtering and modelling methods used to determine protein-molecular target inhibition and by the size of compound libraries to be scanned. Former has been extensively researched and many fast paced algorithms like QSAR [2] and Docking [3] that either build upon experimental results or are entirely computational have been developed. To further reduce the computational time constrain of the overarching problem this report proposes and investigates a general method to reduce the compounds space to be scanned through compound clustering and to find clustroids that act as chemically acceptable representative of its compound cluster for screening.

1.2 Data Set

Online compound libraries like ChEMBL [4] or PubChem [5] contain up to 10^{10} structurally different molecular compounds of less than 300 a.u. Combinatorial Mathematics predict a much higher theoretical chemically relevant compound space of around 10^{60} [6]. Applying Virtual Screening algorithms on this whole compound space with the computer resources of today would take lifetimes to compute.

For the purpose of this investigation to operate in a reasonable time frame the chosen dataset contains 5000 molecular compounds of the ChEMBL database. They were selected because they either bind to the proteins Streptokinase A or Pyruvate kinase M2 as verified by experimental biochemical assays [4]. This makes them ideal for statistical analysis as compound prediction will behave according to a two-class classification system and it is assumed that compounds with similar structural and chemical behaviour bind to the same protein. This assumption is challenged in Section 4 - Validation.

1.3 Clustering Algorithms

Clustering Algorithms have been in use for a long time to group or partition datasets usually based on a similarity metric. Their purpose is to reveal underlying organisation and connection of data and to potentially find meaningful representatives as a means to reduce the dataset. They have applications in all data related disciplines and became very popular in the last decade with the rise of machine learning and big data. Consequently many new algorithms have been developed with a particular focus on exploring hierarchical clustering [7, 8], i.e. revealing organisation on various scales of clustering order. Optimal clustering of a dataset is classified as an NP-hard problem, so that all implementations are approximations with polynomial or sub-polynomial running time. Here three algorithms are introduced and explored in the following sections.

2 Methodology

The aim of this report is to cluster the compound space and find representative clustroids that are able to retain the chemical and structural properties of neighbouring similar compounds. In theory this would reduce the order of magnitude of VS iterations depending on the density of the clustered space as it would suffice to check the representative clustroid instead of every molecule inside the whole cluster. Here three clustering algorithms are investigated and compared on their efficiency, implementation ease and effectiveness to find clustroids. 2D structures are being used and the assumption is made that they are an adequate approximation to predict protein binding, which has shown to not be necessarily true [9], despite some algorithms showing successful predictions [2].

2.1 Morgan Fingerprint and Tanimoto Similarity

The challenge of defining a chemical similarity is an old problem and became particularly relevant with the creation of online databases. The challenge consists of converting the atomistic structure of a compounds into numerical entities that can be compared amongst each other systematically. While many implementations exist, a very common and robust algorithm has shown to be Morgan Fingerprints [10].

Morgan Fingerprints are used to convert a 2D chemical structure into a binary bit-Vector of specified size by using a hash table. This is done through the following two steps:

1. Radius of the local environment to be preserved is chosen:

   The algorithm loops through every non-hydrogen atom in the molecule and constructs an environment based on the input bond length (radius) cut off.

2. Mapping of hash table onto the bit-Vector:

   These environments are then compared to a hash table dictionary and the corresponding keys (integer) are mapped onto the 1024 bit-Vector through the modulus operation by replacing the corresponding positions of the bit-Vectors by a 1 instead of a 0.

It has been shown that this retains the chemical features and connectivity of the molecules through the hash table architecture [10, 9]. It has to be noted that this mapping is not bi-conditional or distinctly reversible due to using the modulus operation, i.e. one unique molecule will only produce one 1024 bit-Vector, whereas one unique 1024 bit-Vector can be mapped back to many molecules.

The similarity or distance between two Vectors can be defined in many ways ranging from Euclidean distances, to cosine distances over arbitrary user defined kernel distances. A very popular [11] measure for binary vectors is the Jaccard index, also called Tanimoto similarity [9] and is calculated between two vectors $\mathbf{a}$ and $\mathbf{b}$ of length k as follows:

$$T_s = \frac{\sum_{j=1}^{k} a_j \times b_j}{\sum_{j=1}^{k} a_j^2 + \sum_{j=1}^{k} b_j^2 - \sum_{j=1}^{k} a_j \times b_j} \tag{1}$$

This measure is used to calculate the similarity between all 5000 molecules and the 5000×5000 difference matrix is computed by subtracting the similarity it from a matrix of ones. This dataset is very skewed and

dense as most molecules are dissimilar to each other with a similarity score of 0 - 0.2, so it is expected for some of the clustering algorithms to perform very differently.

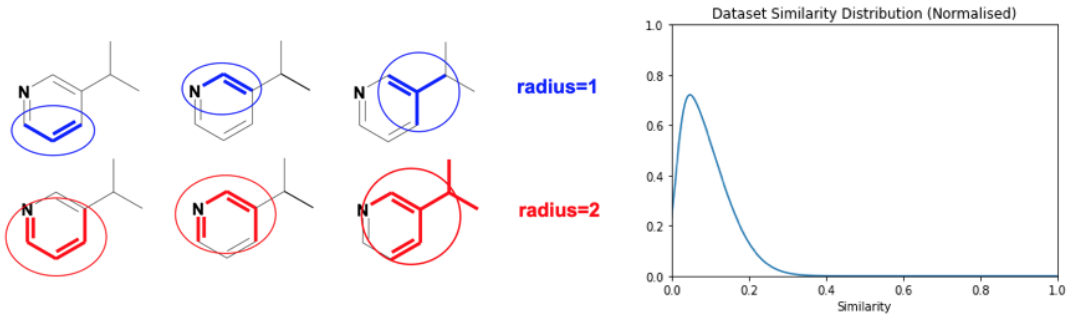

Figure 1: (left) Morgan fingerprint environments to be mapped. (right) Distribution of the 25,000 values in the dataset (note that the 5000 identity values are not shown).

2.2 k-means

The k-means algorithm [12] is one of the most commonly used algorithms to rapidly partition data and is implemented in the most common python machine learning libraries. It operates through defining the number of clusters k and the three steps as follows:

1. Assigning centroids in the data based on the number of clusters k
2. All data points are allocated to a centroid by reducing the in-cluster sum of squares
3. Step 1 and 2 are repeated until convergence in a local minimum is reached

A metric called Silhouette score [13] is used to identify a robust number of clusters k by comparing points within a cluster and identifying their similarity to other points within the same cluster as compared to points in a neighbouring different cluster. Or mathematically using the mean intra-cluster distance a and the mean nearest-cluster distance b over all samples:

$$S = \frac{b - a}{\max(a, b)} \tag{2}$$

2.3 HDBSCAN

HDBSCAN [7] is a recently devloped hierarchical implementation of the DBSCAN algorithm and while it can be run at default parameters, parameter tuning of the four parameters is recommended. It works as follows:

1. Transform the space according to density:

   Every pair of point is assigned a new distance defined as mutual reachability distance based on the local density around a point. This brings distant points apart and close points together in this new transformed space to be more robust against noise.

2. Build the minimum spanning tree (MST):

   Prim's algorithm is used to build a minimum spanning tree on this transformed space by treating the data points as vertices and their mutual reachability distance as weights of connecting edges in a fully connected graph.

3. Build the cluster hierarchy by converting the MST into a dendogram of connected components

4. Condense the cluster tree based on the user defined minimum cluster size:

   Assigning the cut-offs on branches in the dendogram to cluster labels is done by iterating through the whole dendogram hierarchically from root to leaves and assigning a new label at the splits when a new branch is large enough to form its own cluster.

5. Extract the stable clusters from the condensed tree:

   A stability metric is defined as $\sum_{p \in cluster}(\lambda_p - \lambda_{birth})$, where λ_{birth} and λ_p correspond to the inverse distance along the dendogram when the cluster splits off into itself and the inverse distance when the point p splits off that cluster respectively. This metric is computed for all points and clusters along the dendogram and the clusters are extracted where the parent stability is greater than the sum of its children.

2.4 Markov Stability

Markov Stability [8] is a hierarchical clustering algorithm for community detection in networks and operates through an intrinsic parameter that can be understood to control the zooming of the cluster detection. By iterating through this Markov Time an optimised partition is found where a random walk on the graph is likely to remain trapped and a hierarchical structure is obtained. It operates as follows:

1. Construct and sparsify a fully connected graph:

   First a fully connected graph is constructed from the distance matrix. The it is sparisfied by constructing a minimum spanning tree using Prim's algorithm. In order to maintain secondary information multiple edges $z_{i,j}$ are added back into the MST when the following criteria is fulfilled: $mlink_{i,j} + \gamma(d_i + d_j) > z_{i,j}$, where γd_i corresponds to the local density around point i and γ is a tunable parameter to adjust the radius around the point [14].

2. Conduct a random walk on the graph:

   A Markov chain can be constructed on the network having a transition probability defined as:

   $$\mathbf{p}_{t+1} = \mathbf{p}_t \left(D^{-1}A\right) \equiv \mathbf{p}_t M \tag{3}$$

   where $\mathbf{p}_t$ is the 1 x N normalised probability vector, D is the degree matrix defined as $D = \text{diag}(\mathbf{d})$, $\mathbf{d} = A1$ and M is the Markov transition matrix. This Markov chain has a stationary distribution of $\pi = \frac{1}{2m}\mathbf{d}^T$ with $m = \frac{1}{2}\sum_i \mathbf{d}_i$ that has the property $\pi = \pi M$.

3. Partition communities are initialised and optimised according to autocovariance:

   The Markov Stability of a partition $\mathbf{H}$ is subsequently defined as

   $$r(t, \mathbf{H}) = Tr\left[\mathbf{H}^T(\Pi M^t - \pi^T\pi)\mathbf{H}\right] \tag{4}$$

   where the partition matrix $\mathbf{H}$ is the N x c indicator matrix (c being the number of communities) with logical entries H_{ij} to check whether a node i belongs to a community j and $\Pi = \text{diag}(\pi)$. The variable t corresponds to the dimensionless resolution parameter Markov time for which the algorithm ranks the quality of partitions H.

3 Results and Discussion

3.1 k-means

Implementing k-means was straight forward using the scikit-learn [15] python library. The interface with the algorithm follows most default machine learning implementations making it very easy to use (i.e. using .fit(X), .labels_, etc.) and provides access to multiple statistical methods for evaluation.

The assignment of centroids is randomly initialised and depends on the number of clusters k. While the algorithm is robust enough to group similar data together on a global scale consistently, running it

multiple times will lead to different assignments on boarder line data (locally). The resulting clusters come in comparable size and appear to have good separation. The local maxima of the silhouette scores were plotted on the similarity matrix, as they identified as most robust partitions.

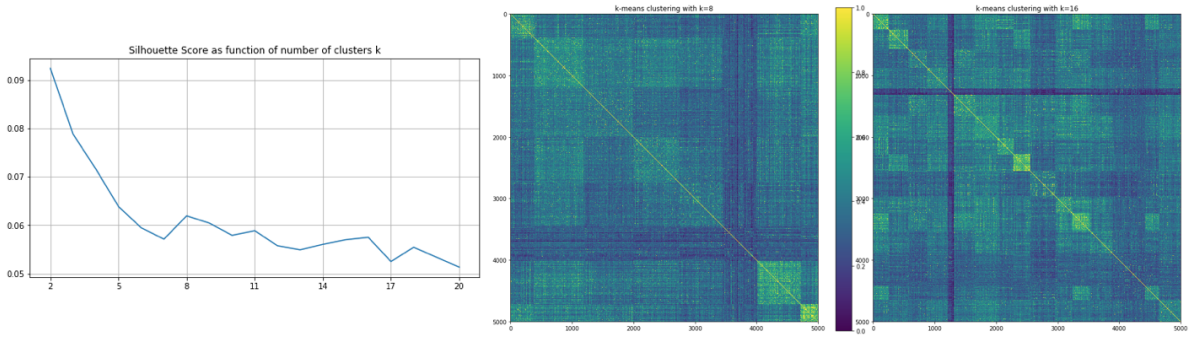

Figure 2: (left) Silhouette score against number of clusters k. Local Maximum observed at k = 8. (middle, right) Clustering visualised on the Similarity matrix for k=8 and k=16. Squares of yellow correspond to clusters that contain compounds with high structural similarity to each other.

The running time of the algorithm is reasonably fast and scalable for larger data sets as it has $O(k \times n \times t)$ complexity. It only requires one parameter to tune (number of clusters k) and is therefore convenient to use. However, the algorithm does not provide any hierarchical information and is lacking organisation within the cluster, that has potential to reveal additional underlying information.

3.2 HDBSCAN

Implementing HDBSCAN was slightly more difficult as the algorithm is not available in common libraries but can be installed via a Github repository. Interfacing with the algorithm is simple as it also follows the structure of most ML algorithms, however with a different indexing style for the labels (starting at -1 instead of 0).

While running the algorithm through one iteration is very fast, tuning the four parameters across multiple order of magnitudes is time consuming and parameter changes can change the result significantly. The resulting clusters for such dense network are very unevenly distributed with one cluster being assigned 70% of the data points and all other clusters averaging around 150. However, those smaller clusters show very detailed partitioning indicating the effectiveness on a local level.

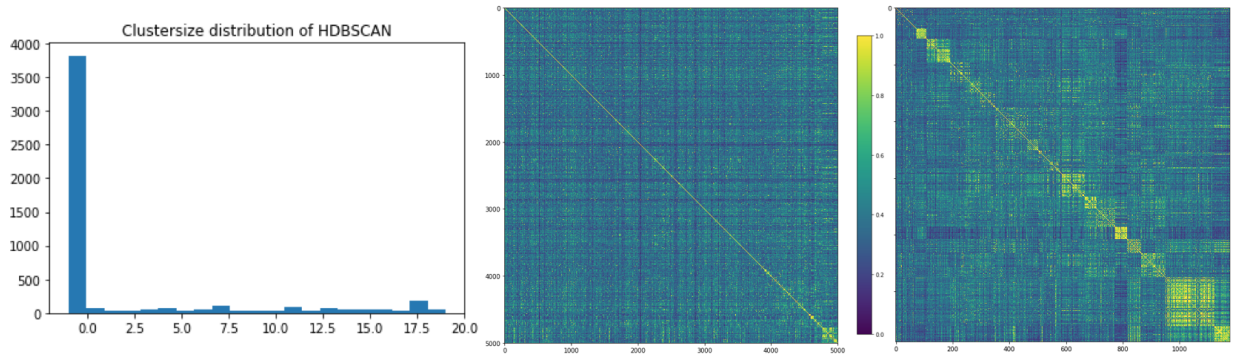

Figure 3: (left) Histogram of the number of data points within each cluster. (middle) Clustering visualised on the 5000 × 5000 Similarity matrix for 20 clusters. (right) The bottom right corner 1200 × 1200 matrix is expanded to show the detailed resolution of the partitioning. Squares of yellow correspond to clusters that contain compounds with high structural similarity to each other.

Assigning centroids by minimising in-cluster sum of squares is straight forward and due to the relative small size of clusters they act as good representations (see Section 4 - Validation). Due to the MST and stability metric that are exhaustively computed across the whole data the algorithm will produce the same clusters given the same input parameters. It runs at sub-$O(n^2)$ complexity and is therefore still viable for medium sized datasets.

3.3 Markov Stability

Markov Stability is an algorithm developed by the Barahona and Yaliraki research groups at Imperial College London and only publicly available as MATLAB version requiring additional non-default toolboxes. Due to similarities to python it was manually converted with the drawback of having a different implementation architecture than the common machine learning libraries. Additional changes on the output of the algorithm were needed to ensure compatibility with their statistical tools.

Running the algorithm was straight forward as there was only one parameter (local density radius γ) to tune. After three attempts an ideal γ was found, i.e. sparsifying the graph until community detection was significantly visible. The resulting diagram showed plateaus for the random walk at multiple number of communities, which can be arranged hierarchically and are of comparable sizes.

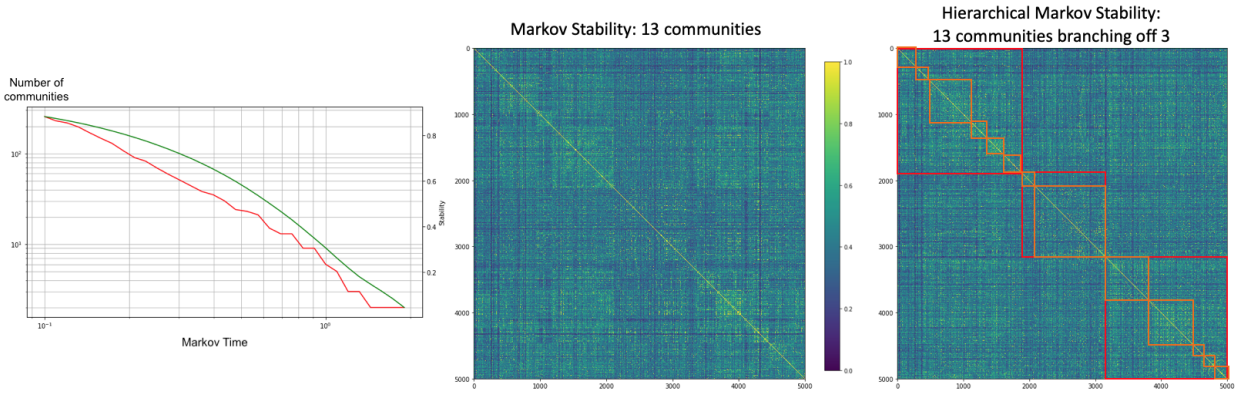

Figure 4: (left) Markov Stability plot generated by the algorithm. Red plateaus correspond to more robust partitioning and indicate favourable number of communities. (middle) Clustering visualised on the 5000 $\times$ 5000 Similarity matrix for 13 clusters. (right) Visualising the hierarchy by sorting the 13 clusters (orange boarder) as a branch of 3 clusters (red boarder). Squares of yellow correspond to clusters that contain compounds with high structural similarity to each other.

Assigning centroids was done through minimising in-cluster sum of squares of the smallest chosen clusters. The robustness of the clusters was confirmed by computing the similarity of clusters at similar Stability, i.e. whether the points within a cluster stay together at different Markov times given the same number of communities. It runs at sub-$O(n^2)$ complexity and is therefore still viable for medium sized datasets and runs much faster on sparse graphs than dense graphs of the same dimensions.

3.4 Algorithm Comparison

The analytical data of the algorithms are presented here for comparison.

	k-means	HDBSCAN	Markov Stability
Total time (s)	812	1500	463
Time per iteration (s)	31	1	119
Parameters	1	4	1
Parameter tuning	minimal	extensive	minimal
Relative cluster sizes	Globally Similar	Locally Similar	Locally and Globally Similar
Partition quality	Local minimum solution	Robust	Robust at high iterations
Cluster Hierarchy	No	Yes	Yes

Figure 5: Comparison of the algorithms across different metrics.

The total time corresponds to the complete running time of the algorithm including iterations used for parameter tuning, whereas time per iteration would correspond to the time it takes to run the algorithm once given optimised parameters. The implications of the relative cluster sizes as well as the partition quality is discussed in the next sections.

4 Validation

Validation is performed using the experimental data provided by the ChEMBL database. All compounds chosen for this investigation either show binding to one of the two proteins Streptokinase A or Pyruvate kinase M2 indicated by a reported IC_{50} value. The clustroids are hence taken for each optimised partitioning scheme and compared to the other molecules within the clusters. A prediction rate of a clustroid is defined as the percentage of data points within a cluster that show protein inhibition on the same protein as the centroid.

	k-means	HDBSCAN	Markov Stability
Best Prediction rate	84.4 %	92.6 %	64.4 %
Best Prediction cluster size	289	54	458
Average Prediction rate	62.1 %	67.4 %	58.1 %
Median Prediction rate	59.9 %	63.4 %	60.7 %

Figure 6: Comparison of the algorithms performance on experimental data. The prediction rate is defined as the percentage of data points within a cluster that show protein inhibition to the same protein as the centroid.

Unsurprisingly HDBSCAN has the highest "Best Prediction rate" given the fine detailed clustering. However, these predictions only cover about 30 % of the compound space as the rest is all assigned to one cluster. Additionally, the cluster size of the highest prediction rate is only at 54 indicating that while very accurate, the amount of compounds the centroid represents is comparably small and would reduce overall Virtual Screening time by around one order of magnitude. k-means and Markov Stability perform similarly with k-means having some better scoring outliers but performing overall slightly worse when looking at the median prediction rate. Markov Stability seems to perform more consistently at larger cluster sizes, hence being able to reduce Virtual Screening by up to 3 orders of magnitude for clusters of size 1000.

It has to be noted that the binding sites on the proteins are not known, so that theoretically multiple binding locations or ways of inhibition could exist. Consequently this implies to a small likelihood that if molecule 1 and 2 show experimental inhibition to Streptokinase A it does not guarantee a structural similarity between the molecules as they could be inhibiting the protein's function using different mechanisms. In general, the assumption is that the same protein binding site would be inhibited by molecules of similar structures or that share specific chemical features. To verify this assumption additional extensive theoretical docking or experimental X-ray crystallisation studies would have to be performed.

The results strongly suggest a trade-off between predication rate accuracy and reduction of compound space through the clustroids, i.e. larger cluster give lower prediction rates. One obvious way to minimise this problem is to combine the clustering algorithms as they operate on different optimisation functions (**k-means:** reducing in-cluster sum of squares, **HDBSCAN:** reducing the stability metric on the dendogram, **Markov Stability:** increasing the quality of partitions using auto-covariance based Markov Stability). A simple suggestion would be to use the HDBSCAN produced large-sized cluster that contains 70 % of data and find partitions using Markov Stability. Alternatively, an initial coarse clustering could be conducted using k-means and HDBSCAN could operate on each resulting cluster. This would improve the impact of the density transformation step as it will be able to distinguish even small local differences since it doesn't need to consider global ones due to the pre-clustering by k-means. Another option would be to re-define the clusteroids or imposing multiple clustroids per cluster to enable representation of undiscovered hierarchies and sub-clusters. This could be readily implemented by using k-means on the small clusters as it operates on the same principle of in-cluster sum of squares minimisation.

5 Conclusion

This report has introduced three clustering algorithms and compared them on their implementation ease, time and complexity performance and tested their prediction ability when compared to experimental data. It has been discussed that ideally a combination of the algorithms would perform most accurately, while individually there are strong trade-offs between predication rate accuracy and reduction of compound space through the clustroids. Further, implementation of the algorithms and integration with existing libraries can be decisive when choosing a preferred algorithm with k-means being most promising. Markov Stability shows promising results consistently over large datasets when the graph is sparsified sufficiently and requires minimal to no further knowledge about the dataset to reveal underlying communities and a hierarchical structure. HDBSCAN performs more inconsistently on this dataset due to its dense nature and requires extensive parameter tuning but shows very good local scale performance on the 30 % subset of data. Overall this investigation thoroughly analyses the three algorithms at hand and identifies issues and advantages of each. Future research is required into the combinational implementation of the algorithms to ensure better consistent prediction rates. Further, the underlying assumption that 2D structures are sufficient to predict protein inhibition should be challenged and potentially computational implementations of 3D structural models can be used for further investigations.

References

[1] T. Cheng, Q. Li, Z. Zhou, Y. Wang, and S. H. Bryant, "Structure-based virtual screening for drug discovery: a problem-centric review," *The AAPS journal*, vol. 14, no. 1, pp. 133–141, 2012.

[2] A. Z. Dudek, T. Arodz, and J. Gálvez, "Computational methods in developing quantitative structure-activity relationships (qsar): a review," *Combinatorial chemistry & high throughput screening*, vol. 9, no. 3, pp. 213–228, 2006.

[3] O. Trott and A. J. Olson, "Autodock vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading," *Journal of computational chemistry*, vol. 31, no. 2, pp. 455–461, 2010.

[4] A. Gaulton, A. Hersey, M. Nowotka, A. P. Bento, J. Chambers, D. Mendez, P. Mutowo, F. Atkinson, L. J. Bellis, E. Cibrián-Uhalte, *et al.*, "The chembl database in 2017," *Nucleic acids research*, vol. 45, no. D1, pp. D945–D954, 2016.

[5] S. Kim, P. A. Thiessen, E. E. Bolton, J. Chen, G. Fu, A. Gindulyte, L. Han, J. He, S. He, B. A. Shoemaker, *et al.*, "Pubchem substance and compound databases," *Nucleic acids research*, vol. 44, no. D1, pp. D1202–D1213, 2015.

[6] J.-L. Reymond and M. Awale, "Exploring chemical space for drug discovery using the chemical universe database," *ACS chemical neuroscience*, vol. 3, no. 9, pp. 649–657, 2012.

[7] L. McInnes, J. Healy, and S. Astels, "hdbscan: Hierarchical density based clustering," *The Journal of Open Source Software*, vol. 2, no. 11, p. 205, 2017.

[8] J.-C. Delvenne, S. N. Yaliraki, and M. Barahona, "Stability of graph communities across time scales," *Proceedings of the national academy of sciences*, vol. 107, no. 29, pp. 12755–12760, 2010.

[9] P. Willett, "Similarity-based virtual screening using 2d fingerprints," *Drug discovery today*, vol. 11, no. 23-24, pp. 1046–1053, 2006.

[10] D. Rogers and M. Hahn, "Extended-connectivity fingerprints," *Journal of chemical information and modeling*, vol. 50, no. 5, pp. 742–754, 2010.

[11] D. Bajusz, A. Rácz, and K. Héberger, "Why is tanimoto index an appropriate choice for fingerprint-based similarity calculations?," *Journal of cheminformatics*, vol. 7, no. 1, p. 20, 2015.

[12] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1027–1035, Society for Industrial and Applied Mathematics, 2007.

[13] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[14] M. Beguerisse-Díaz, B. Vangelov, and M. Barahona, "Finding role communities in directed networks using role-based similarity, markov stability and the relaxed minimum spanning tree," in *2013 IEEE Global Conference on Signal and Information Processing*, pp. 937–940, IEEE, 2013.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python ," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.