Chapter 7

Optimization Algorithms for Chemoinformatics and Material-Informatics

Abraham Yosipof and Hanoch Senderowitz

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/62483

Abstract

Modeling complex phenomena in chemoinformatics and material-informatics can often be formulated as single-objective or multi-objective optimization problems (SOOPs or MOOPs). For example, the design of new drugs or new materials is inherently a MOOP since drugs/materials require the simultaneous optimization of multiple parameters.

In this chapter, we present several algorithms based on global stochastic optimization. These algorithms are applicable to multiple tasks in chemoinformatics and material-informatics including the following: (1) representativeness analysis, namely the selection of a representative subset from within a parent data set. (2) Derivation of quantitative structure–activity relationship models. Such models are used in multiple areas to predict activities from structures and to provide insight into factors (e.g., descriptors) governing activities. (3) Outlier removal to clean a parent data set from objects (e.g., compounds) that may demonstrate abnormal behavior.

The performances of the new algorithms were evaluated using different data sets and multiple measures and were found to outperform previously reported methods.

Due to the modular nature of the algorithms, they could be combined into machine-learning workflows. In the final section, we provide an example of one such workflow and apply it to the development of predictive models in pharmaceutical and material sciences.

Keywords: chemoinformatics, material-informatics, simulated annealing, QSAR, outlier removal, machine learning, representativeness, *k*NN

1. Introduction

Modeling complex phenomena in chemoinformatics and material-informatics can often be formulated as multi-variables/single-objective or multi-variables/multi-objectives optimization problems. Common examples of the former include sampling of complex energy landscapes using conformational search methods [1], docking or molecular simulation techniques [2, 3], derivation of statistical models, namely quantitative structure–activity relationship (QSAR) models [4, 5], and diversity or representativeness analysis [6, 7]. The design of new compounds with pharmaceutical relevance on the other hand is inherently a multi-objective optimization problem (MOOP) since drugs require the simultaneous optimization of many parameters and consequently constitute a compromise between often conflicting requirements. In a similar manner, the design of new materials could also be regarded as a MOOP. For example, the design of new photovoltaic cells requires the simultaneous optimization of the current and the voltage.

This chapter focuses on optimization algorithms from three different areas: (1) Representativeness analysis, that is, the selection of a representative subset from within a parent data set. Representativeness analysis has multiple applications in chemoinformatics and material-informatics, for example, for rationally selecting subsets of compounds for experimental analysis and for rationally partitioning a parent data set into a modeling set and a test set. (2) Derivation of predictive, nonlinear machine-learning models correlating activities with descriptors while inherently incorporating feature selection to identify the most relevant descriptors. Such the so-called QSAR models find many usages in chemistry, biology, environmental sciences, and material sciences to predict the activities of new compounds/materials and to provide insight into the factors governing these activities [8]. (3) Outlier removal, to clean a parent data set from objects (e.g., compounds) that may demonstrate abnormal behavior. Outlier removal is a mandatory step prior to the derivation of any statistical model. In all cases, the corresponding problems (i.e., how to select a representative subset from within a parent data set, how to remove outliers from within a parent data set, and how to build predictive QSAR models) were formulated as either single-objective optimization problems (SOOPs) or MOOPs. These problems were then solved using Monte Carlo (MC)/Simulated Annealing (SA) or Genetic Algorithm (GA) as the optimization engines.

Finally, while developed independently and with multiple potential applications in mind, all algorithms could be used as components of machine-learning workflows for data mining and the derivation of QSAR models (see Section 5).

This chapter is organized as follows: We begin with a short introduction to SOOPs and MOOPs (Section 2), followed by a description of two of the most common optimization engines, MC/SA (Section 3.1) and GA (Section 3.2). Section 4 provides a description of the different optimization-based algorithms, and Section 5 lists a few representative examples. Finally, Section 6 concludes the chapter.

http://dx.doi.org/10.5772/62483

2. Single-objective and multi-objectives optimization

As noted above, multiple problems in chemoinformatics and material-informatics could be formulated as optimization problems. Such a formulation requires the definition of a target (or objective) function(s) (f) and a set of variables (X_1, X_2, X_3,…,X_n), which are related to the scientific problem of interest, and which together define a complex, multi-dimensional surface with an a priori unknown distribution of optima. The task then is to locate the global optimum or preferably, since phenomena in these fields are rarely governed by a single solution, a set of optima.

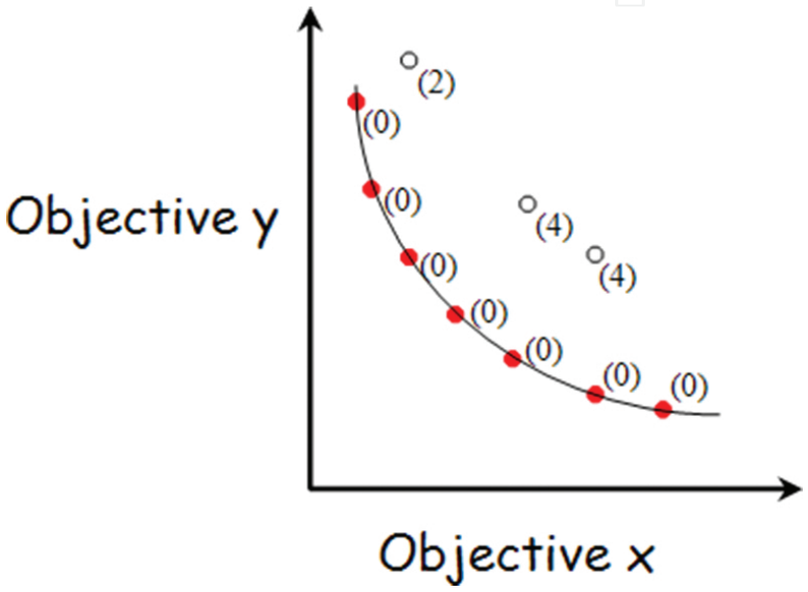

Figure 1. Potential solutions of a two objectives problem represented by the Pareto front. Dominated solutions are shown as empty circle, and the number of solutions which dominate them is written in parentheses.

Optimization problems could be broadly divided into two categories, namely SOOP and MOOP depending on the number of target functions which should be simultaneously optimized. SOOPs search for the best optimum on a surface defined by a single target function and its variables. While this might be a difficult task in particular for complex, multi-dimensional surfaces, a solution exists and a thorough enough search of the space, at least in theory, is bound to locate it. This, however, is not the case for MOOPs, which extend optimization theory by permitting several design objectives to be optimized simultaneously. The principle of MOOP was first formalized by Pareto [9]. In MOOP, a single solution that outperforms all other solutions in all objectives does not necessarily exist. Instead, several equally good (termed non-dominated) solutions exist representing various compromises among the objectives. A solution is said to be non-dominated if it is better than all other solutions in at least one objective. The set of non-dominated solutions represents the Pareto front. This is illustrated in **Figure 1**, where each circle represents a solution to the problem. The curved line represents the Pareto front. Each solution is designated a Pareto rank that is based on the number of solutions which dominate it. The solid circles are non-dominated solutions, which

have Pareto rank of zero and fall on the Pareto front. Dominated solutions are shown as empty circle, and the number of solutions which dominate them is written in parentheses.

Due to the complex nature of many of the chemoinformatics and material-informatics-related target functions (e.g., nonlinearity, non-continuity, non-derivability), non-derivatives-based, stochastic optimization algorithms should be used as the optimization engine. Several global stochastic algorithms including GAs [10], genetic programming [11], particle swarm optimization [12], MC [13]/SA [14], and the iterative stochastic elimination method [15] are reported in the literature. The algorithms presented in this chapter utilize MC/SA- and GA-based optimizers to solve SOOPs and MOOPs.

3. Optimization engines

3.1. Monte Carlo/simulated annealing

MC methods and in particular the Metropolis variant [13] use random moves in order to optimize a multi-dimensional space defined by a dependent target function and a set of independent variables.

Metropolis MC starts from an initial random position. At each iteration, a new position ($position_{i+1}$) is randomly generated by a random displacement from the current position ($position_i$). The "energy" (i.e., the value of the target function) of the resulting new position is computed, and ΔE, the energetic difference between the current and the new position, is determined (Equation 1).

$$\Delta E = E\left(position_{i+1}\right) - E\left(position_i\right) \tag{1}$$

The probability that this new position is accepted is based on the Metropolis test as defined in Equations 2 and 3.

$$if\ \left(\Delta \mathrm{E} < 0\right) then\ accept\ step \tag{2}$$

$$else\ \text{Equation 3}$$

$$if\left(random\left[0,1\right] \le e^{-\frac{\Delta \mathrm{E}}{kT}}\right) then\ accept\ step \tag{3}$$

Thus, if the new position has a lower value of the target function (commonly referred to as lower energy), the transition is accepted. Otherwise, a uniformly distributed random number between 0 and 1 is drawn and the new position will only be accepted if the Boltzmann

http://dx.doi.org/10.5772/62483

probability is higher or equal to the random number as defined in Equation 3, where kT is the effective temperature.

MC simulations are often coupled to a SA [14] procedure and usually called SA optimization. SA gradually decreases the temperature according to a predefined cooling schedule (**Figure 2**). This procedure controls the probability of acceptance or rejection of high energy moves. The SA procedure increases the probability of locating the global minimum.

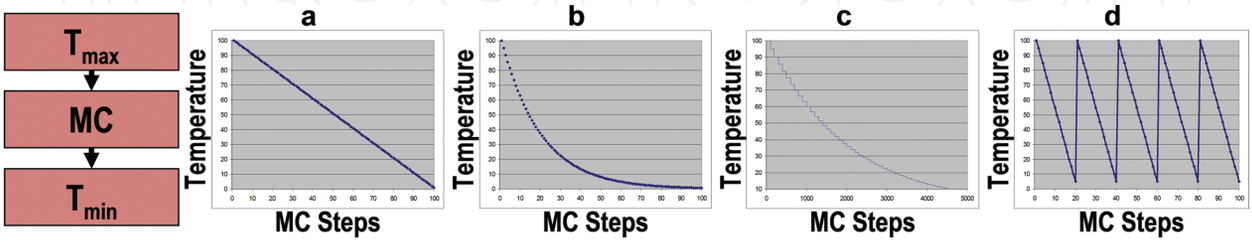

Figure 2. Examples of cooling schedules for SA: (a) smooth linear cooling; (b) smooth exponential cooling (c) stepwise exponential cooling; (d) saw-tooth linear cooling composed of repeating cooling cycles in order to avoid trapping in local minima.

3.2. Genetic algorithms

GAs [10] are global optimizers designed from evolutionary principles. Such algorithms evolve a set of chromosomes, each corresponding to a unique solution to the optimization problem, using a set of genetic operators such as selection of the fittest, mutation, and crossover. The selection of individuals to be subjected to the genetic operators is governed by their fitness values (calculated by a fitness function related to the specific scientific problem) so that chromosomes which represent a better solution to the optimization problem are given more chances to ”reproduce” than those chromosomes which are poorer solutions. This process iterates multiple times (generations) until no improvement in the fitness function is observed or until the number of predefined generation has been exhausted. By considering, at each generation, multiple rather than single chromosomes, GAs provide multiple solutions to the optimization problem.

4. Optimization-based algorithms

4.1. Representativeness and diversity analysis

Advances in combinatorial chemistry and high-throughput screening techniques have greatly expanded the number of drug-like compounds that could be synthesized and tested. However, even then, only a small fraction of the accessible chemistry space could be synthesized and screened [16]. This in turn has led to the development of rational approaches for the design of

screening libraries and in particular libraries composed of diverse compounds (i.e., compounds largely differing from one another). Hassan et al. [6] formulated the selection of a diverse subset from within a parent data set as an optimization problem using several diversity functions. In particular, the MaxMin function calculates the square of the minimal distance, d_{ij}^2, over all (*i,j*) pairs comprising the selected subset according to Equation 4.

$$MaxMin = Max\left(\min_{i \neq j}\left(d_{ij}^{\ 2}\right)\right) \tag{4}$$

where d_{ij} is the distance between compounds *i* and *j*, and the summation runs over all the descriptors (features). The MaxMin function is optimized (maximized) by means of a SA algorithm to produce the subset with the largest value (i.e., the most diverse subset).

Treating the selection of a diverse subset as an optimization problem has the advantage that this objective could be combined with other objectives into a MOOP. For example, it is possible to select a subset that the best balances its internal diversity, pharmacological profile, and price.

However, diverse subsets are often biased toward the inclusion of outliers and as a result do not well represent the parent data set. In drug discovery, focusing on outliers may translate into selecting and testing "extreme" compounds, for example, compounds with too many functional groups or compounds with too high molecular weights. Such compounds are likely to be difficult to optimize into drug candidates [17]. Unless such "extreme" compounds could be easily detected and removed (e.g., if their "extremeness" results from a single property), they are likely to remain in the data set.

Thus, instead of selecting diverse subsets, it might be advisable to select representative subsets, which better mirror the distribution of the parent data set. If properly selected, such subsets will include compounds which are different from one another yet are not "bizarre." However, despite their potential usefulness, representative subsets have gained much less attention than diverse subsets.

Representative subsets could be selected using clustering algorithms such as hierarchical clustering and k-means clustering. Both partition an input data set into a predefined number of clusters. Hierarchical clustering produces a dendogram of clusters in which the root node contains all the compounds and the leaf nodes contain each, a single compound. Divisive hierarchical clustering starts at the root node and iteratively divides clusters until the leaf nodes are reached. Agglomerate hierarchical clustering starts with the leaf nodes and iteratively combines closest neighbors clusters until the root node is reached. Extracting from the dendogram a specific number of clusters and selecting a compound from each cluster leads to a representative/diverse subset. k-means clustering [18] operates by first selecting at random a user defined number of seeds (*k*), then by assigning each compound to its closest seed. This leads to the formation of *k* initial clusters. Centroids of all clusters are then calculated, and compounds are re-assigned to their closest centroid. This process is repeated until the clusters are stable, that is, no compounds are re-assigned. Again selecting a compound from each cluster provides a representative/diverse subset.

http://dx.doi.org/10.5772/62483

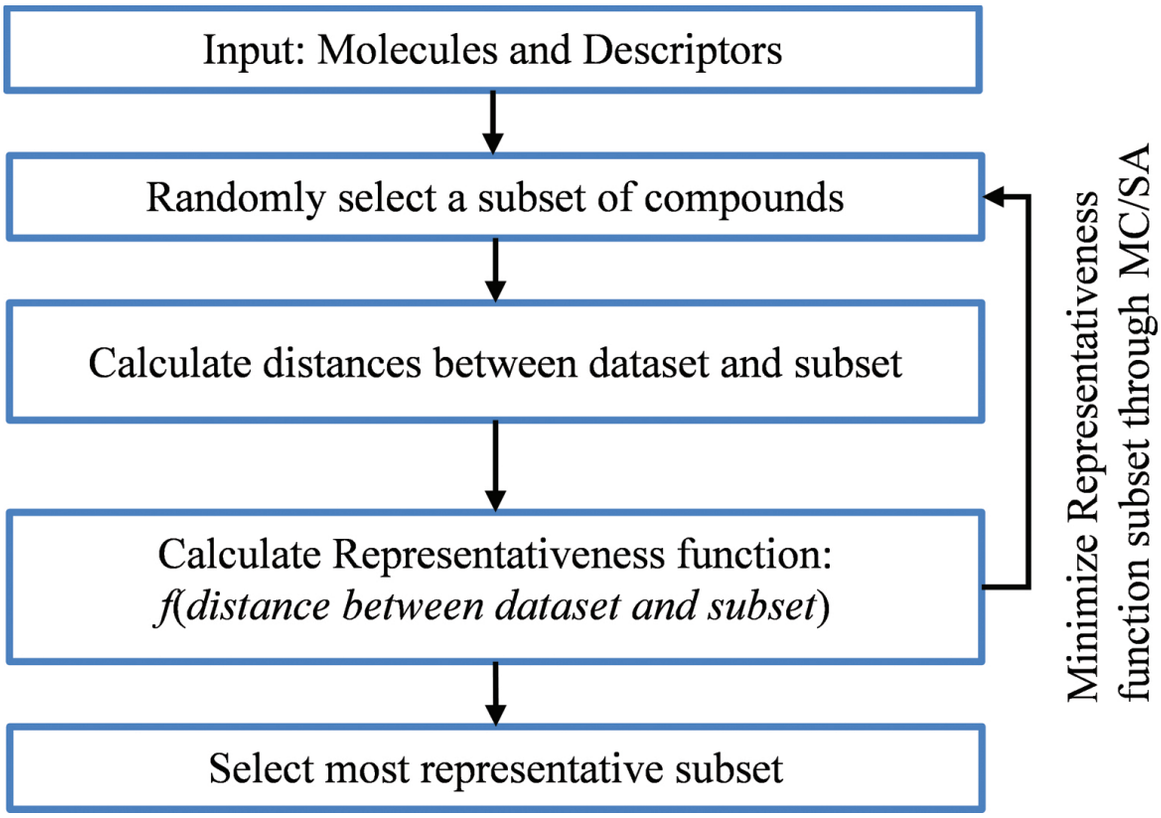

Figure 3. Schematic representation of the representativeness optimization algorithm.

In 2014, Yosipof and Senderowitz [19] introduced an optimization algorithm for the direct selection of a representative subset from within a data set. The algorithm optimizes by means of a MC/SA procedure a representativeness function, based on pairwise distances between subset and data set compounds. The algorithm consists of the following steps (**Figure 3**):

1. Characterize each compound in the data set by a set of descriptors.
2. Normalize descriptors by converting them into *Z*-scores according to Equation 5 where μ is the mean, and σ is the standard deviation:

$$Z = \frac{x_i - \mu}{\sigma} \tag{5}$$

   The resulting scores are then converted to [0,1] range by calculating the cumulative probability assuming a normal distribution ($\mu = 0$; $\sigma^2 = 1$).
3. Select a random subset *s* of size *k* from within the *l* compounds comprising the data set.
4. Calculate the Euclidean distance between compound *i* from the data set and all *k* compounds comprising subset *s*.
5. Take the minimum Euclidean distance from step (4) as the score for compound *i*:

$$score_i = \min(dist_{i,\{s\}})$$

6. Repeat steps (4) and (5) for all (*l*–*k*) compounds remaining in the data set.

7. Calculate the average score over all (*l*–*k*) compounds. This score characterizes subset s:

$$score_s = \frac{1}{(l-k)} \sum_{i=1}^{l-k} score_i$$

Minimize $score_s$ through a MC/SA procedure [14]. At each step replace, at random, a single compound from *s* with a candidate compound from the unselected portion of the data set, calculate a new score, $score_s$', and accept it according to the MC/SA algorithm.

Similar to diversity analysis, a major advantage in treating the selection of a representative subset as an optimization problem is the ability to combine it with additional objectives into a MOOP [20, 21]. This was demonstrated by the Pareto-based optimization of the representativeness and MaxMin functions. The Pareto algorithm evaluates MaxMin (Equation 4) and the representativeness function for a selected subset (termed a solution to the MOOP) and assigns to it a Pareto rank based on the number of solutions dominating it. In this case, solution *i* dominates solution *j* if MaxMin*(i)* < MaxMin*(j)* and Score*(i)* < Score*(j)* (where *Score* is calculated according to step 7 given above). Under this dominance criterion, the value of MaxMin is minimized rather than maximized in contrast with the original implementation of this function for diversity selection. MaxMin minimization biases the selected subset toward the more populated regions of the database allowing the two functions (MaxMin and representativeness) to work in concert. The Pareto rank is then minimized using Metropolis Monte Carlo, and the solutions with rank = 0 (i.e., non-dominant solutions) are kept to construct the Pareto front. Finally, a solution on the Pareto front is randomly selected. Alternatively, all the solutions could be presented to the user for manual inspection, evaluation, and selection.

Representative subsets could be used under two general scenarios: (1) Results obtained for a representative subset are used to infer on the properties of the parent data set. For example, the biological evaluation of a representative subset could provide information on the activities of the entire parent data set. Thus, testing only a representative subset will provide a similar amount of information as would have been gained by testing the entire data set. (2) A representative subset is selected from within a parent data set, set aside, and used to validate models generated by machine-learning algorithms. In the area of chemoinformatics, such models are known as quantitative structure–activity relationship (QSAR) models (see Section 4.2) [22].

To evaluate the representativeness algorithms under the first scenario, Yosipof and Senderowitz [19] selected a subset of 200 compounds from the Comprehensive Medicinal Chemistry (CMC) database using five representative/diversity algorithms, namely representativeness optimization, Pareto-based optimization, hierarchical clustering, k-means clustering, and MaxMin optimization. The CMC database contains 4,855 pharmaceutical compounds classified into 105 different biological indications. The degree with which each subset is able to represent the parent database (i.e., include a similar distribution of indications) was estimated

http://dx.doi.org/10.5772/62483

using the χ^2 goodness-of-fit test. In this test, the null hypothesis (H_0) states that the distribution of biological indications within the subset and database are similar. In contrast, the H_1 hypothesis states that these distributions are significantly different. The objective is therefore to stay on the null hypothesis. The χ^2 statistics is defined as follows:

$$\chi^2 = \sum_{i=1}^{n=105} \frac{(O_i - E_i)}{E_i} \tag{6}$$

where O_i and E_i represent, respectively, the observed and expected frequencies for a biological indication i in the 200 compounds subset. E_i is derived from the frequency of indication i in the parent database. The results of this test demonstrated that the distribution of indications within the subsets selected by the representativeness optimization and the Pareto-based optimization are statistically indistinguishable (p-value >0.05) from that in the parent database. In contrast, subset selected by k-means clustering, hierarchical clustering, and MaxMin display distributions which are markedly different from the database (p-value <0.05).

To evaluate the representativeness algorithms under the second scenario (a representative subset is selected, set aside, and used to validate models built on the rest of the data set), it was incorporated into a workflow developed for the derivation of predictive QSAR models using machine-learning algorithms. Following the work of Tropsha [23] and others [22], it is today recognized that the predictive power of such models could only be evaluated from their performances on external test sets. In particular, test sets that uniformly span the chemistry space of the parent data set provide reliable performance estimates for QSAR models operating in this space. It therefore follows that such performance estimates correlate with how well a set of compounds represents the parent data set from which it was selected. Yosipof and Senderowitz [19] used the new representativeness algorithm to rationally select test sets of varying sizes from two data sets of pharmaceutical relevance (logBBB and *Plasmodium falciparum* inhibition) and estimated the performances of models derived with five classification techniques (decision trees, random forests, ANN, SVM, *k*NN) on these test sets. Similar test sets were also selected from the Pareto front generated by the simultaneous optimization of representativeness and MaxMin as well as by the k-means clustering, hierarchical clustering, and MaxMin for comparison. Model performances were estimated using the corrected classification rate (CCR; Equation 7)

$$CCR = \frac{1}{2}\left(\frac{T_N}{N_N} + \frac{T_P}{N_P}\right) \tag{7}$$

where T_N and T_P represent the number of true negative and true positive predictions, respectively, and N_N and N_P represent the total number of the two activity classes.

The results (**Table 1**) indicate that the best performances were obtained with the Pareto method followed by the representativeness function and k-means clustering. The other two methods,

namely hierarchical clustering and the MaxMin function led to poorer performances. Thus, representativeness-based methods indeed produce subsets which are more representative of the parent data sets.

Method	CCR
Hierarchical clustering	**0.78**
k-meansclustering	**0.80**
MaxMin	**0.74**
Representativeness	**0.80**
Pareto-based	**0.83**

Table 1. Average performances of QSAR models on test sets selected with the hierarchical clustering, k-means clustering, MaxMin, representativeness, and Pareto-based methods. Averaging performed over the two data sets, the five model building algorithms, and the different sizes of the selected test sets.

4.2. Derivation of predictive QSAR models

QSAR (or QSPR, quantitative structure–property relationship) is a general name for a host of methods that attempt to correlate a specific activity for a set of compounds with their structure-derived descriptors (i.e., features) by means of a mathematical model.

QSAR models take the form $Ai = f(D_1, D_2, \ldots, D_n)$ where Ai is the dependent variable representing the activity (or any other property of interest) for a set of objects (e.g., compounds or materials), and $D_1, D_2, \ldots, D_n$ are calculated (or experimentally measured) independent variables (i.e., descriptors). f is an empirical mathematical transformation that should be applied to the descriptors in order to calculate the property values for the objects.

QSAR models are typically built according to a basic workflow which involves the following steps: (1) data collection; (2) preprocessing (data set preparation and curation, descriptors calculation, descriptors filtering); (3) model generation; (4) model validation.

Data collection involves the assembly of a data set of compounds/materials with known activities. Once collected, the data should be carefully curated, errors should be corrected (or if not possible, problematic compounds should be removed), and a set of descriptors should be obtained (calculated or measured). Finally, constant or nearly constant descriptors should be removed and usually, correlated descriptors are removed as well.

QSAR models are built using multiple machine-learning approaches. The modeling process begins with a modeling set and proceeds by performing regression-based or classification-based analysis to construct a model of the activity as a function of the descriptors. Machine-learning techniques are mostly used in this area, because they can deal with very complex relationships between descriptors and activities [8].

There are two types of regression-based methods, namely linear and nonlinear. An example of a linear method is provided by multiple linear regression which is extensively applied in Hansch analysis [24]. Examples of nonlinear methods are the *k*-nearest neighbors (*k*NN) [5],

http://dx.doi.org/10.5772/62483

and the random forest (RF) [25] methods. Finally, the resulting model should be validated on an external test set. An external set can be obtained by splitting the input and curated data set prior to the model development process or by obtaining additional data [26, 27].

Inherent to the most QSAR methods is feature (i.e., descriptor) selection. The purpose of this process is to select from within a large number of calculated/experimental descriptors those that best correlate with the activity. This stage is required since typically, the number of calculate-able descriptors by far exceeds that of compounds with measured activity data. Having too many descriptors calculated for too few compounds may lead to over fitting and chance correlation [23, 28]. In some cases, features selected for QSAR modeling are more important than the predictive model itself. These selected descriptors contain useful information that can reveal significant information and can help in understanding and rationalization the data and the results. The selection of descriptors subset could be treated as an optimization problem whereby a function related to model performances is optimized in the space of the descriptors.

One algorithm that couples feature selection procedure with a machine-learning algorithm at the model derivation stage is the *k*NN algorithm. This algorithm assumes that the activity of a compound could be predicted from the average activities of the *k* compounds most similar to it (*k*NN). This idea follows directly from the similar property principle [29] which states that similar compounds have similar properties. The similar property principle is well-validated in pharmaceutical sciences and was recently extended to photovoltaic cells [30]. Since chemical similarity between two compounds depends on the molecular descriptors used to characterize them, the algorithm searches the space of available descriptors subsets for that subset in terms of which the similar properties principle is best satisfied. This is done by optimizing the leave-one-out (LOO) cross-validated value (Q^2_{LOO}, Equation 8) in the space of the descriptors (the space of the descriptors is a multidimensional space where each dimension corresponds to one descriptor. In this space compounds are represented by points and the distance between any two points represents the degree of similarity between the corresponding compounds). Q^2_{LOO} is given by

$$Q^2_{LOO} = 1 - \frac{\sum_Y \left(Y_{exp} - Y_{LOO}\right)^2}{\sum_Y \left(Y_{exp} - \overline{Y}_{exp}\right)^2} \tag{8}$$

where Y_{exp} is the experimental value Y_{LOO} is the predicted value and $\overline{Y}_{exp}$ is the mean of the experimental results.

The *k*NN algorithm was first implemented by Zheng and Tropsha [5] using SA as the optimization engine. In this original implementation, only the descriptors space was searched with the SA procedure, while the number of nearest neighbors (*k*) was evaluated exhaustively (between 1 and 5). In 2015, Yosipof and Senderowitz [31] implemented the *k*NN algorithm optimizing Q^2_{LOO} in the space of the descriptors and the number of nearest neighbors. A schematic representation of the *k*NN optimization algorithm is provided in **Figure 4**.

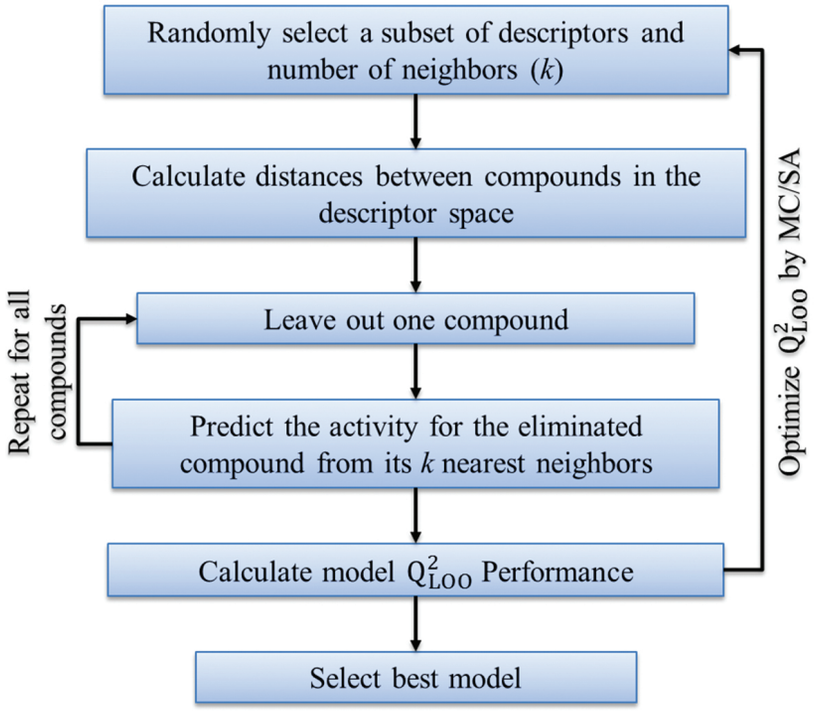

Figure 4. Schematic representation of the *k*NN optimization algorithm.

The *k*NN algorithm has been extensively used to build predictive QSAR models in many fields including computer-aided drug design and environmental sciences [8]. Recently, the method has been applied in the field of material-informatics and used to predict the photovoltaic properties of solar cell libraries (see Section 5) [30].

4.3. Outlier removal

Data sets in general and data sets consisting of molecular compounds in particular often contain objects (e.g., compounds) that are different in some respect from the rest of the data set. Such compounds are called outliers. The presence of outliers in a data set can affect machine-learning-related activities including model derivation, interpretation, and subsequent decision-making. In particular, outliers can compromise the ability of machine-learning algorithms to develop predictive models since many algorithms typically used for this purpose will attempt to fit the outliers at the expanse of the bulk. Thus, while outliers may point to an interesting behavior that needs to be investigated separately [32], they should be removed from the data set prior to the model construction.

Several methods for outliers removal have been reported in the literature [33]. Statistical estimators could be used to identify outliers if their values follow a well-defined distribution [34, 35]. These methods are called parametric methods. For example, for a Gaussian distribution, outliers could be defined as compounds with descriptors values deviating from the mean by a certain number of standard deviations. For non-well-defined distributions, nonparametric methods should be used. For example, distance-based methods identify outliers by measuring

http://dx.doi.org/10.5772/62483

Euclidean distances between objects in a predefined descriptors space. In the basic distance-based method, outliers are defined as compounds having at least p percent (define by the user) of their distances to the other compounds larger than a user defined threshold distance [36]. An improvement to this method was proposed by Ramaswamy et al. [37] and termed the K-based method. According to this method, compounds are ranked according to their Euclidean distances to their k^{th}-nearest neighbors and the n compounds with the highest rank (largest distances) are considered as outliers. Another method is based on compounds clustering and on subsequent removal of compounds (i.e., outliers) populating small clusters (e.g., singletons) [38]. Another nonparametric method utilizes a variant of the support vector machine (SVM) algorithm, namely one-class SVM, which isolates the outlier's class from the rest of the compounds [39].

The above-described techniques remove outliers in a single step from one predefined descriptors space and are therefore termed one-pass methods. These methods have several disadvantages. First, outliers are descriptor space-dependent with different spaces giving rise to different outliers. Second, if several outliers are present in a data set, they may mask each other so that some will not be recognized [40]. Finally, the removal of one outlier based on a specific descriptor space may affect the distribution of the remaining compounds either in the same space or in different spaces, leading to the potential appearance of new outliers. These challenges could be met by removing outliers in an iterative manner.

Recently, Yosipof and Senderowitz [31] presented a new method for the iterative identification and subsequent removal of outliers identified in potentially different descriptors spaces based on the k nearest neighbors optimization algorithm (kNN-OR algorithm). According to this approach, an outlier is defined as a compound whose distance to its kNNs is too large for its activity to be reliably predicted (see **Figure 5**). At each iteration, the algorithm builds a kNN model, evaluates it according to the LOO cross-validation metric (Q^2_{LOO} ; see Equation 8) and removes from the data set the compound whose elimination results in the largest increase in Q^2_{LOO} . This procedure is repeated until Q^2_{LOO} exceeds a pre-defined threshold.

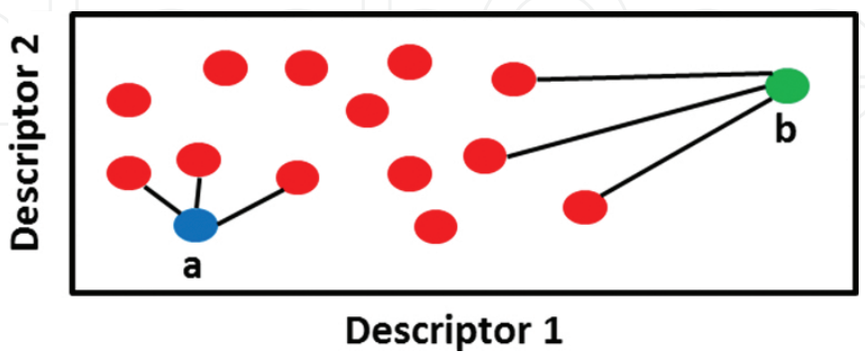

Figure 5. Compounds (red spheres) embedded in a two-dimensional descriptors space. Compound a (blue sphere) is in close vicinity to at least some of the other compounds leading to short distances to its (three) nearest neighbors and to a likely reliable kNN-based activity prediction. Compound b (green sphere) has large distances to its (three) nearest neighbors and is therefore an outlier. The kNN-based activity prediction for this compound is likely to be erroneous.

In the example presented in **Figure 5**, the removal of compound *b* from the data set is expected to improve the model and to increase the value of *k*NN-Q^2_{LOO} More generally,for a set of compounds whose activities are predicted via *k*NN, outlier(*s*) removal will lead to an increase in Q^2_{LOO} . Thus, the following procedure for outlier removal was developed (**Figure 6**):

1. For a set of compounds, run *k*NN to obtain the model with the highest Q^2_{LOO} .
2. For each compound, calculate the improvement in Q^2_{LOO} upon its removal from the data set.
3. Remove the compound whose elimination from the data set results in the largest increase in Q^2_{LOO} . When a compound is removed from the data set, it is also removed from the list of nearest neighbors of all other compounds. In such cases, the removed compound will be replaced by the next-in-line nearest neighbor for the purpose of activity prediction.
4. If no compound could be removed from the data set based on the first model (i.e., for all compounds, their removal from the data set does not lead to an improved Q^2_{LOO}), repeat steps 2–3 for the second best model (which is built in a different descriptor space).
5. Repeat steps 1–4 above until Q^2_{LOO} is sufficiently high (stopping criterion).

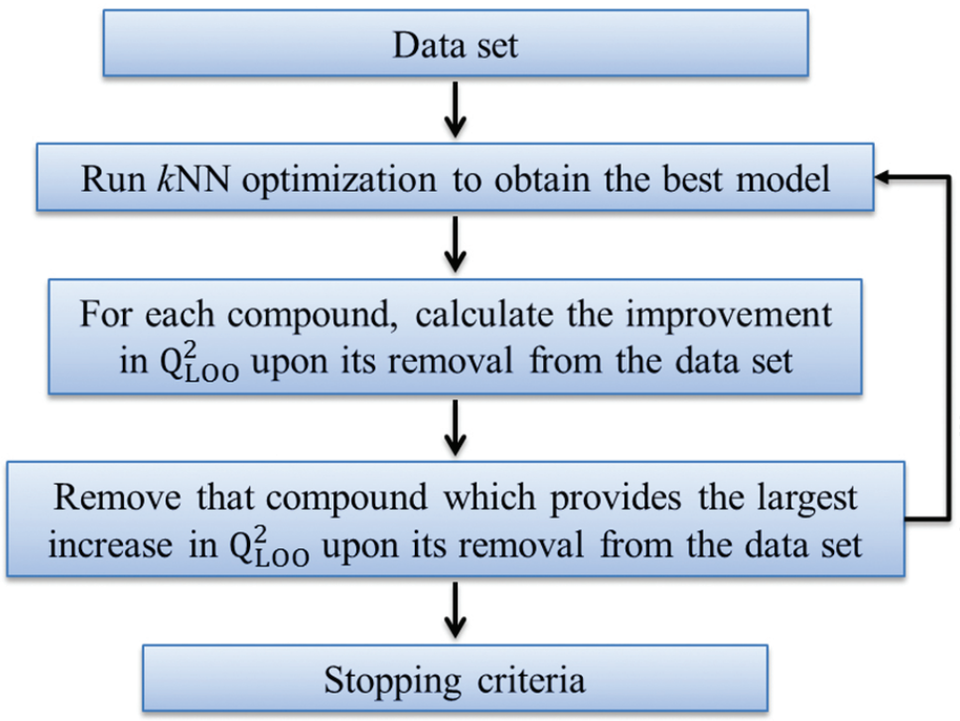

Figure 6. Schematic representation of the *k*NN optimization-based outlier removal algorithm.

The above-described *k*NN-OR algorithm is "greedy" in nature, removing at each iteration the compound that leads to the largest improvement in Q^2_{LOO} without considering the possibility

http://dx.doi.org/10.5772/62483

that a sub-optimal improvement at a given iteration may pay off later. Nahum et al. [41] introduced a "look ahead" mechanism into outlier removal by treating it as a multi-objective optimization problem using genetic algorithm (GA-*k*NN). The new method simultaneously minimizes the number of compounds to be removed and maximizes *k*NN-derived Q^2_{LOO} . The multi-objective optimization is performed using the strength Pareto evolutionary algorithm 2 (SPEA2) [42], which approximates the Pareto front for MOOPs. SPEA2 uses an external set (archive) for storing primarily non-dominated solutions. At each generation, it combines archive solutions with the current population to form the next archive that is then used to produce offspring for the next generation. Each individual *i* in the archive A_t and the population P_t is assigned a raw fitness value *R(i)*, determined by the number of its dominators in both archive and population. *R(i)*=0 corresponds to a non-dominated individual, whereas a high *R(i)* value means that individual *i* is dominated by many individuals. These raw values are then used to rank the individuals for the purpose of selecting candidates for reproduction. However, the raw fitness value by itself may be insufficient for ranking when the most individuals do not dominate each other. Therefore, additional information, based on the k^{th} nearest neighbor density of the individuals, is incorporated to remove rank redundancy. The workflow of the SPEA2 algorithm is described below:

Algorithm—SPEA2

Input:	N—Archive size
	M—Offspring population size
	T—Maximum number of generations
Output:	A∗—Non-dominated set of solutions to the optimization problem

1. Initialization: Generate an initial population P_0 and create the empty archive (external set) A_0=∅. Set t=0.

2. Fitness assignment: Calculate fitness values of individuals in P_t and A_t.

3. Environmental selection: Copy all non-dominated individuals in P_t and A_t to A_{t+1}. If size of A_{t+1} exceeds N then reduce A_{t+1} by means of the truncation operator, otherwise if size of A_{t+1} is less than N then fill A_{t+1} with dominated individuals in P_t and A_t.

4. Termination: If $t \geq T$ or another stopping criterion is satisfied then set $A*$ to the set of decision vectors represented by the non-dominated individuals in A_{t+1}. Stop.

5. Mating selection: Perform selection with replacement on A_{t+1} in order to fill the mating pool.

6. Variation: Apply recombination and mutation operators to the mating pool and set P_{t+1} to the resulting population. Increment generation counter (t=t+1) and go to Step 2.

Each solution to the multi-objective optimization problem specifies the set of descriptors, and the number of neighbors used by the *k*NN algorithm as well as the identity of compounds considered as outliers. This information was coded by a three component binary array (i.e., chromosome). The first part of the array encoded the number of neighbors using a binary representation. The second part described the descriptors identity ("1" and "0" representing,

respectively, selected and unselected descriptors for the current solution). The third part listed the compounds considered as outliers using the same representation as for the descriptors.

The resulting chromosomes were subjected to a multi-site crossover operator to produce new chromosomes. These contained a new combination of the number of neighbors, descriptors, and outliers. The new chromosomes were further mutated to increase the diversity of the solutions population and to prevent trapping in local minima. Mutations were performed on the entire chromosome and consequently affected the descriptors, the number of nearest neighbors, and the identity of outliers.

For each new generation, raw fitness values were calculated for each individual based on the information encoded in its chromosome. This calculation was based on Q^2_{LOO} (which in turn depends on the descriptors selected, the number of neighbors selected, and the identity of outliers removed via the *k*NN algorithm) and on the number of outliers removed. The process was repeated until the termination criteria were met.

The performances of the two algorithms (*k*NN-OR and GA-*k*NN) were tested by removing outliers from three data sets of pharmaceutical relevance (logBBB, Factor 7 inhibitors, and dihydrofolate reductase inhibitors [DHFR]) and using the remaining compounds for two purposes: (1) to compare the internal diversities of the filtered data sets with those of the parent sets; (2) to build and validate QSAR models with two machine-learning methods, *k*NN, and random forest. The results clearly demonstrated that data sets rationally filtered with the two new outlier removal algorithms were more internally diverse and support QSAR models which provided better prediction statistics than data sets filtered by removing the same number of compounds using five other outlier removal methods (distance based, distance K-based, one-class SVM, statistics and random removal).

As an example, the results for the DHFR data set are presented below. This data set contained 673 compounds with known activity data (dependent variable). Each compound was characterized by 19 descriptors as the independent variables and by its biological activity in the form of an IC50 value as the dependent variable (*Y*). The compounds were subjected to the *k*NN-OR and the GA-*k*NN algorithms using a stopping criterion of $Q^2_{LOO} > 0.85$. For *k*NN-OR,this criterion was met after the removal of 87 compounds,leaving a total of 586 compounds for the subsequent diversity analysis and QSAR modeling. For the GA-*k*NN algorithm,this criterion was met after the removal of only 75 compounds. For comparison, 87 compounds were removed using the GA-*k*NN algorithm and the other six methods considered in that work.For the GA-*k*NN model,the removal of 87 compounds led to a model with $Q^2_{LOO} > 0.87$.

The internal diversity of the data set prior to and following compounds removal was evaluated by calculating pairwise Euclidean distances between all compounds in the original descriptors space. The results are presented in **Figure 7** and demonstrate that the removal of either 75 compounds (11.1% of the data set) by the GA-*k*NN algorithm or 87 compounds (12.9% of the data set) by the GA-*k*NN and *k*NN-OR algorithms did not change the distribution of distances, suggesting that the coverage of chemistry space was largely unaffected by the removal of outliers. This in turn implies that the applicability domain for models derived from the filtered

http://dx.doi.org/10.5772/62483

data set will not be reduced. In contrast, all other methods (except for random removal) demonstrated truncation at long distances implying reduced internal diversity.

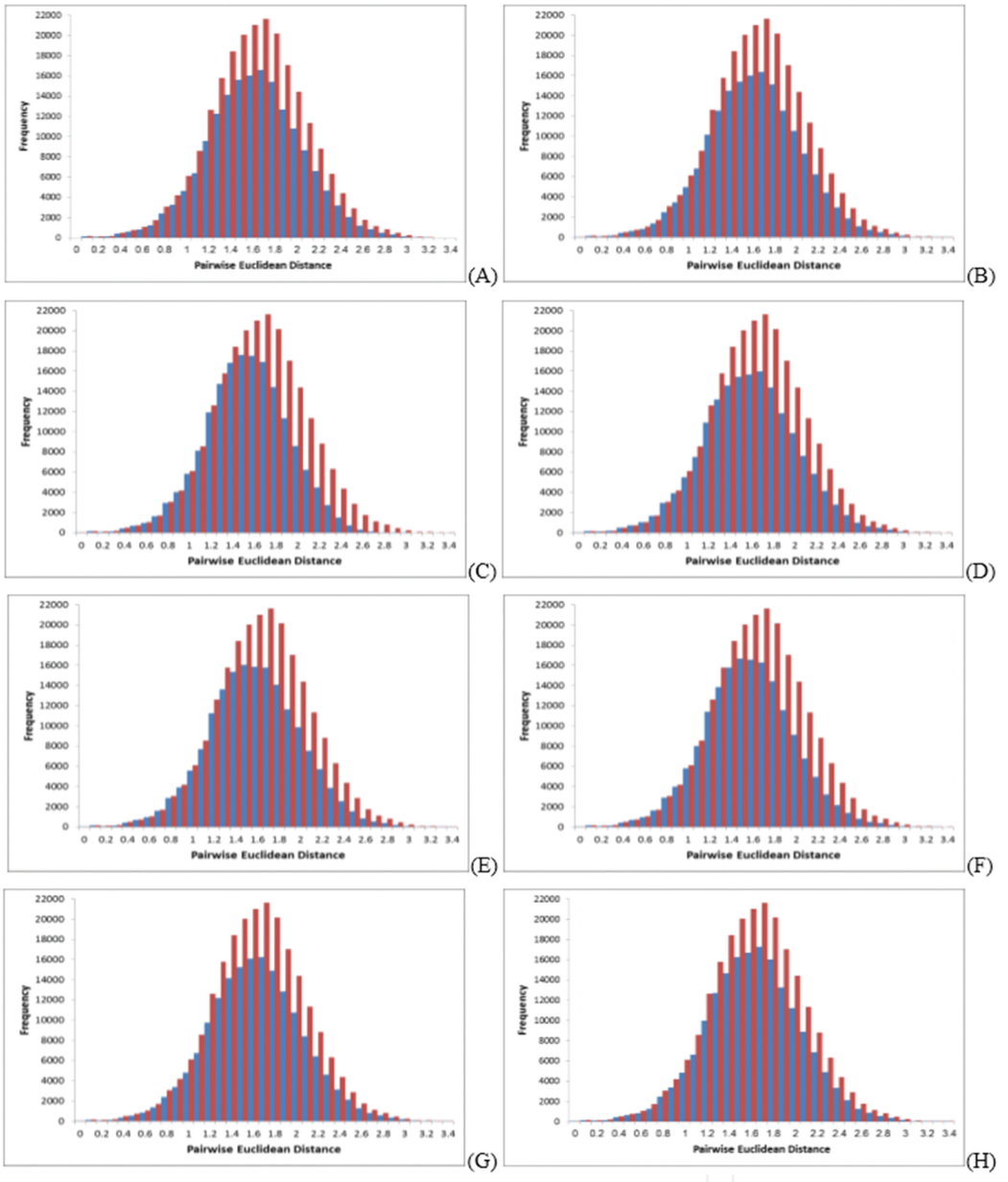

Figure 7. A comparison between pairwise Euclidean distances distributions before (red lines) and after (blue lines) the removal of outliers from the DHFR data set. (A) GA–*k*NN optimization-based outlier removal (75 compounds removed); (B) GA–*k*NN optimization-based outlier removal (87 compounds removed); (C) *k*NN–OR optimization-based outlier removal (87 compounds removed); (D) distance-based outlier removal (87 compounds removed); (E) distance K-based outlier removal (87 compounds removed); (F) one-class SVM-based outlier removal (87 compounds removed); (G) statistics-based outlier removal (87 compounds removed); (H) random "outlier" removal (87 compounds removed).

Compounds surviving the filtration process were divided into a modeling set (469 compounds) and a test set (117 compounds) and subjected to QSAR modeling using *k*NN and RF. The

resulting models were evaluated by standard parameters. Modeling sets subjected to *k*NN were evaluated using LOO cross-validation (Equation 8), while modeling sets subjected to RF (also known as out-of-bag set) were evaluated by the determination coefficient (R^2_{OOB}; Equation 9,).Test sets (external validation sets) were evaluated by the external explained variance Q^2_{ext}; equation Equation 10).

$$R^2_{OOB} = 1 - \frac{\sum_Y \left(Y_{exp} - Y_{OOB}\right)^2}{\sum_Y \left(Y_{exp} - \overline{Y}_{exp}\right)^2} \tag{9}$$

$$Q^2_{ext} = 1 - \frac{\sum_Y \left(Y_{exp} - Y_{pre}\right)^2}{\sum_Y \left(Y_{exp} - \overline{Y}_{exp}\right)^2} \tag{10}$$

where Y_{exp} is the experimental value Y_{OOB} and Y_{pre} are the predicted value and $\overline{Y}_{exp}$ is the mean of the experimental results over modeling set (training set) compounds.

The results are presented in **Table 2** and demonstrate that the rational removal of outliers using the *k*NN-OR and the GA-*k*NN algorithms led to filtered data sets which produced both *k*NN-based and RF-based QSAR models with the best prediction statistics.

	***k*NN**		**RF**	
	Q^2_{LOO}	Q^2_{ext}	R^2_{OOB}	Q^2_{ext}
GA–*k*NN	**0.78**	**0.83**	**0.73**	**0.75**
***k*NN–OR**	**0.79**	**0.83**	**0.71**	**0.77**
Distance-based	**0.54**	**0.63**	**0.55**	**0.68**
Distance K-based	**0.64**	**0.74**	**0.66**	**0.73**
One-class SVM	**0.62**	**0.65**	**0.62**	**0.72**
Statistics	**0.55**	**0.54**	**0.55**	**0.61**
Random consensus	**0.59**	**0.53**	**0.61**	**0.62**

Table 2. Results obtained for the DHFR data set using *k*NN and RF. The results given under "random consensus" were averaged over all 10 random models.

5. Applications

While developed independently, all algorithms, together with additional tools, were incorporate into a machine-learning workflow for data mining and were used for the derivation of multiple QSAR models. The resulting workflow is depicted in **Figure 8**.

http://dx.doi.org/10.5772/62483

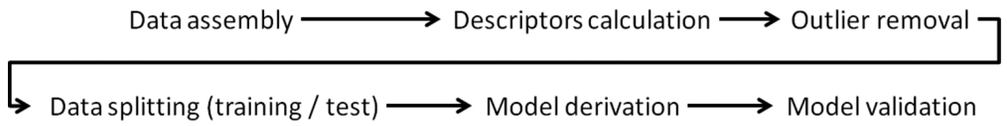

Figure 8. A schematic representation of the machine-learning workflow.

Below, we provide two examples taken from the fields of pharmaceutical and material sciences.

5.1. Blood–brain barrier permeation (logBBB) model

Blood–brain barrier permeability is an important parameter in drug design. Drugs targeting the central nervous system (CNS) are required to permeate through the blood–brain barrier. Conversely, drugs that do not affect the CNS should not penetrate the barrier due to potential side effects. Blood–brain barrier permeability is typically expressed as logBBB with positive and negative values indicating, respectively, permeating and non-permeating compounds. Since the experimental determination of logBBB values is resources consuming, multiple QSAR models have been developed to predict this property [19, 31, 41].

A data set of 152 compounds with known logBBB values was compiled from the literature, manually curated and characterized by 15 descriptors [19, 31]. This data set was subjected to the *k*NN-OR algorithm (Section 4.3) using a stopping criterion of Q^2_{LOO} >0.85. The application of this criterion led to the removal of 19 compounds, leaving 133 compounds for QSAR modeling. The stepwise Q^2_{LOO} values upon compounds removal is presented in **Figure 9**.

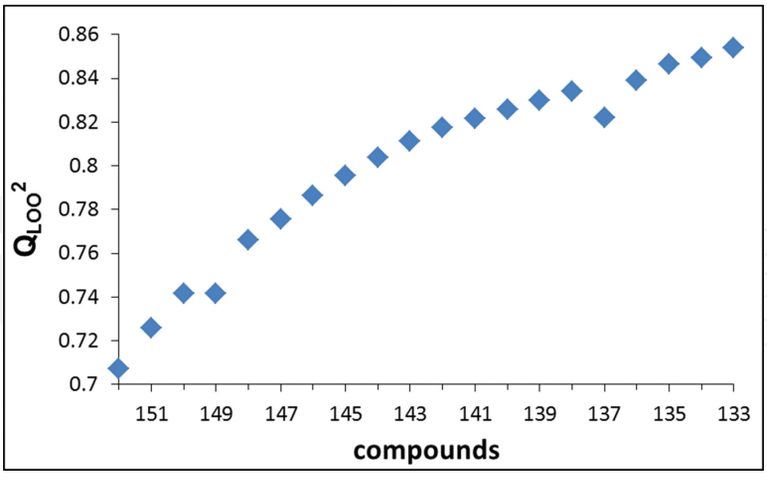

Figure 9. Q^2_{LOO} values as a function of compound removal for the logBBB data set. Outlier removal began with a set of 152 compounds, and the stopping criterion was met after the removal of 19 compounds.

Compounds surviving the filtration procedures were divided into a modeling set (106 compounds) and a validation test set (27 compounds) using the representativeness function described in Section 4.1. QSAR models were built on the modeling sets using either *k*NN

(Section 4.2) or the random forest (RF) algorithms, and the resulting models were tested on the validation sets. Model performances were evaluated with Equations 8–10 and were found to be Q^2_{LOO} =0.81 and Q^2_{ext} =0.88 for *k*NN and R^2_{LOO} =0.60 and Q^2_{ext} =0.65 for RF [31]. These values are similar to those obtained by other logBBB QSAR models.

Similar results were obtained when this data set was filtered using the GA–*k*NN algorithm [41]. In this case, the stopping criterion (Q^2_{LOO} >0.85) was met after the removal of 13 compounds. As before compounds surviving the filtration procedure were divided into a modeling set (111) and a test set (28) and subjected to QSAR modeling leading to Q^2_{LOO} =0.80 and Q^2_{ext} =0.86 for *k*NN and R^2_{OOB} = 0.67 and Q^2_{ext} =0.65 for RF [41].

5.2. Photovoltaic activities of solar cells

The raise in demands for clean energy is likely to increase the importance of solar cells as future energy resources. In particular, cells entirely made of metal oxides have the potential to provide clean and affordable energy if their power conversion efficiencies are improved. Designing solar cells with improved photovoltaic properties could be greatly assisted by the application of optimization algorithms as described in this chapter.

Yosipof et al. [30] used the workflow described in **Figure 8** to build predictive *k*NN-based models for a library of 169 solar cells made of a combination of titanium and copper oxides ($TiO_2|Cu_2O$). In this case, the dependent variables which had to be predicted were the open-circuit voltage (V_{OC}), the short-circuit current (J_{SC}), and the internal quantum efficiency (*IQE*), and the dependent variables consisted of the thicknesses of the copper oxide and the titanium oxide layers, the ratio between the layers thicknesses, the experimentally measured bandgap and the maximum theoretical calculated photocurrent. In this case, subjecting the cells to the *k*NN-OR algorithm did not lead to the removal of any outliers and consequently, the entire library could be used for QSAR modeling. The library was divided into a modeling (training) set and a validation test set using the algorithm described in Section 4.1, and the predictive QSAR models were derived with the *k*NN algorithm described in Section 4.2. The results are presented in **Table 3**.

End point	Q^2_{LOO}	Q^2_{ext}	Descriptors selected
J_{SC}	**0.92**	**0.92**	The thickness of the titanium oxide layer and the thickness of the copper oxide layer
V_{OC}	**0.78**	**0.89**	The thickness of the titanium oxide layer and the thickness of the copper oxide layer
IQE	**0.91**	**0.87**	The thickness of the titanium oxide layer and the thickness of the copper oxide layer

Table 3. Results obtained with the *k*NN algorithm for the $TiO_2|Cu_2O$ library.

Overall, the resulting models demonstrated good prediction statistics suggesting that they are likely to be useful for the design of new and improved solar cells. In addition, the feature

http://dx.doi.org/10.5772/62483

selection procedure inherent to the *k*NN algorithm highlighted the importance of the metal oxide layer thickness in controlling the photovoltaic properties (last column in **Table 3**).

6. Conclusions

In this chapter, we introduced several tools based on global stochastic optimization for chemoinformatics and material-informatics applications in three areas. To select representative subsets from within parent data sets, we described a new representativeness function and its optimization either alone or simultaneously with the MaxMin function. The two resulting algorithms were found to outperform previously reported subset selection methods.

For the derivation of predictive, nonlinear machine-learning models we reviewed the *k*NN algorithm, which searches the space of available descriptors combinations to identify those that best satisfy the similar properties principle. Descriptors spaces compatible with this principle give rise to models with good prediction statistics.

Finally, we introduced two new algorithms for the identification and subsequent removal of outliers based on the *k*NN method. The *k*NN–OR algorithm iteratively removes from the parent data set compounds (outliers) which the best improve model performances. The GA–*k*NN algorithm simultaneously optimizes model performances together with the number of outliers. This algorithm is usually able to remove a smaller number of outliers while still maintaining good model performances. Retaining in the data set submitted to QSAR modeling as many compounds as possible is likely to increase the applicability domain of the resulting model. The new algorithms were found to outperform other outlier removal methods when tested on three data sets.

Finally, the new algorithms, together with additional tools, were combined into a machine-learning workflow and used for the derivation of predictive QSAR models.

The algorithms presented in this chapter are likely to be useful for multiple applications in the fields of chemoinformatics and material-informatics.

Author details

Abraham Yosipof[1*] and Hanoch Senderowitz[2]

*Address all correspondence to: avi.yosipof@gmail.com

1 Department of Business Administration, Peres Academic Center, Rehovot, Israel

2 Department of Chemistry, Bar-Ilan University, Ramat-Gan, Israel

References

[1] Leach A.R. Molecular modelling: principles and applications. Pearson Education: Harlow, England; 2001.

[2] Liu M., Wang S. MCDOCK: a Monte Carlo simulation approach to the molecular docking problem. Journal of Computer-Aided Molecular Design. 1999;13(5):435–451.

[3] Jorgensen W.L. Efficient drug lead discovery and optimization. Accounts of Chemical Research. 2009;42(6):724–733.

[4] Agrafiotis D.K., Cedeño W. Feature selection for structure–activity correlation using binary particle swarms. Journal of Medicinal Chemistry. 2002;45(5):1098–1107.

[5] Zheng W., Tropsha A. Novel variable selection quantitative structure–property relationship approach based on the k-nearest-neighbor principle. Journal of Chemical Information and Computer Sciences. 1999;40(1):185–194.

[6] Hassan M., Bielawski J., Hempel J., Waldman M. Optimization and visualization of molecular diversity of combinatorial libraries. Molecular Diversity. 1996;2(1):64–74.

[7] Agrafiotis D.K. Stochastic algorithms for maximizing molecular diversity. Journal of Chemical Information and Computer Sciences. 1997;37(5):841–851.

[8] Cherkasov A., Muratov E.N., Fourches D., Varnek A., Baskin I.I., Cronin M., et al. QSAR modeling: where have you been? Where are you going to? Journal of Medicinal Chemistry. 2013;57(12):4977–5010.

[9] Pareto V. Manual of political economy. Milan: Milan, Italy 1906.

[10] Mitchell M. An introduction to genetic algorithms. London: MIT Press; 1998.

[11] Banzhaf W., Nordin P., Keller R.E., Francone F.D. Genetic programming: an introduction. San Francisco, USA: Morgan Kaufmann; 1998.

[12] Kennedy J., Eberhart R. Particle swarm optimization. In: IEEE International Conference on Neural Networks; Perth, WA. IEEE; 1995. pp 1942–1948.

[13] Metropolis N., Ulam S. The Monte Carlo method. Journal of the American Statistical Association. 1949;44(247):335–341.

[14] Kirkpatrick S., Gelatt C.D., Vecchi M.P. Optimization by simulated annealing. Science. 1983;220(4598):671–680.

[15] Glick M., Rayan A., Goldblum A. A stochastic algorithm for global optimization and for best populations: a test case of side chains in proteins. Proceedings of the National Academy of Sciences. 2002;99(2):703–708.

[16] Drew K.L.M., Baiman H., Khwaounjoo P., Yu B., Reynisson J. Size estimation of chemical space: how big is it? Journal of Pharmacy and Pharmacology. 2012;64(4):490–495.

[17] Clark R.D. OptiSim: an extended dissimilarity selection method for finding diverse representative subsets. Journal of Chemical Information and Computer Sciences. 1997;37(6):1181–1188.

[18] MacQueen J. Some methods for classification and analysis of multivariate observations. In: Fifth Berkeley Symposium on Mathematical Statistics and Probability; Berkeley, California. University of California Press; 1967. pp 281–297.

[19] Yosipof A., Senderowitz H. Optimization of molecular representativeness. Journal of Chemical Information and Modeling. 2014;54(6):1567–1577.

[20] Gillet V.J., Khatib W., Willett P., Fleming P.J., Green D.V.S. Combinatorial library design using a multiobjective genetic algorithm. Journal of Chemical Information and Computer Sciences. 2002;42(2):375–385.

[21] Gillet V.J., Willett P., Fleming P.J., Green D.V.S. Designing focused libraries using MoSELECT. Journal of Molecular Graphics and Modelling. 2002;20(6):491–498.

[22] Eriksson L., Jaworska J., Worth A.P., Cronin M.T., McDowell R.M., Gramatica P. Methods for reliability and uncertainty assessment and for applicability evaluations of classification- and regression-based QSARs. Environmental Health Perspectives. 2003;111(10):1361–1375.

[23] Tropsha A. Best practices for QSAR model development, validation, and exploitation. Molecular Informatics. 2010;29(6–7):476–488.

[24] Hansch C., Fujita T. p-sd-pi analysis. A method for the correlation of biological activity and chemical structure. Journal of the American Chemical Society. 1964;86(8):1616–1626.

[25] Polishchuk P.G., Muratov E.N., Artemenko A.G., Kolumbin O.G., Muratov N.N., Kuz'min V.E. Application of random forest approach to QSAR prediction of aquatic toxicity. Journal of Chemical Information and Modeling. 2009;49(11):2481–2488.

[26] Gramatica P. Principles of QSAR models validation: internal and external. QSAR & Combinatorial Science. 2007;26(5):694–701.

[27] Tropsha A., Gramatica P., Gombar V.K. The importance of being earnest: validation is the absolute essential for successful application and interpretation of QSPR models. QSAR & Combinatorial Science. 2003;22(1):69–77.

[28] Whitley D.C., Ford M.G., Livingstone D.J. Unsupervised forward selection: a method for eliminating redundant variables. Journal of Chemical Information and Computer Sciences. 2000;40(5):1160–1168.

[29] Johnson M.A., Maggiora G.M. Concepts and applications of molecular similarity. New York: John Wiley & Sons; 1990.

[30] Yosipof A., Nahum O.E., Anderson A.Y., Barad H.N., Zaban A., Senderowitz H. Data mining and machine learning tools for combinatorial material science of all-oxide photovoltaic cells. Molecular Informatics. 2015;34(6–7):367–379.

[31] Yosipof A., Senderowitz H. k-Nearest neighbors optimization-based outlier removal. Journal of Computational Chemistry. 2015;36(8):493–506.

[32] Kim K. Outliers in SAR and QSAR: is unusual binding mode a possible source of outliers? Journal of Computer-Aided Molecular Design. 2007;21(1–3):63–86.

[33] Ben-Gal I. Outlier detection. In: Maimon O., Rokach L., editors. Data Mining and Knowledge Discovery Handbook. New York: Springer US; 2005. pp 131–146.

[34] Barnett V., Lewis T. Outliers in statistical data. New York: Wiley; 1994.

[35] Hawkins D.M. Identification of outliers. London: Chapman and Hall; 1980.

[36] Knorr E., Ng R. Algorithms for mining distance-based outliers in large datasets. In: the 24th International Conference on Very Large Data Bases, VLDB; New York, USA: Morgan Kaufmann Publishers Inc.; 1998.

[37] Ramaswamy S., Rastogi R., Shim K. Efficient algorithms for mining outliers from large data sets. ACM SIGMOD Record. 2000;29(2):427–438.

[38] Kaufman L., Rousseeuw P.J. Finding groups in data: an introduction to cluster analysis. New York: John Wiley & Sons; 2009.

[39] Schölkopf B., Smola A.J., Williamson R.C., Bartlett P. L. New support vector algorithms. Neural Computation. 2000;12(5):1207–1245.

[40] Cao D.S., Liang Y.Z., Xu Q.S., Li H.D., Chen X. A new strategy of outlier detection for QSAR/QSPR. Journal of Computational Chemistry. 2010;31(3):592–602.

[41] Nahum O.E., Yosipof A., Senderowitz H. A multi-objective genetic algorithm for outlier removal. Journal of Chemical Information and Modeling. 2015;55(12):2507–2518.

[42] Zitzler E., Laumanns M., Thiele L. SPEA2: improving the strength Pareto evolutionary algorithm. Eidgenössische Technische Hochschule Zürich (ETH), Institut für Technische Informatik und Kommunikationsnetze (TIK) 2001.