Chapter 2

Biological Activity Prediction in Computational Drug Design: Focusing Attention on the Neural Network and Deep Learning Algorithms

Fahimeh Ghasemi

Department of Bioinformatics and Systems Biology, School of Advanced Technologies in Medicine, Isfahan University of Medical Sciences Isfahan, Iran

Email: f_ghasemi@amt.mui.ac.ir

Abstract

Because of naturally time-consuming, complex and costly processes of laboratory procedures, two last decades can be regarded as the golden period of using mathematical or statistical methods in computational drug design as an alternative approach, called Quantitative Structure−Activity Relationship (QSAR). QSAR studies are depended on the 2D and 3D molecular representation as an input model to predict biological activities. Biological activity is a beneficial or side effects of a drug on living matter that plays a critical roles in medical applications. Molecular representations, frequently well-known as molecular descriptors, have been converted to the useful numerical information via mathematical procedures. Molecular descriptors can be divided to two major groups, experimental values, i.e. generally physico-chemical properties, and theoretical data. However, most of them are not actually capable of precisely ranking biological targets inhibitors but make QSAR models suitable for high throughput virtual screening (e.g. 10 million of compounds), in general high reliability is not required. On the other hand, small number of molecules of in-house compounds with thousands of descriptors usually leads to redundancy problem in QSAR studies. Over the previous decades, various machine learning algorithms have been devised to avoid these predicaments in drug discovery. Nevertheless, pruning over-fitting problem has merged as another challenge in the recent years leading to the advent of deep learning networks.

Keywords: Computational drug design; QSAR studies; Molecular descriptors; Deep learning; Machine learning.

1. Introduction

Because of the vast number of targets and potentially active molecules each year being detected, design of a de-novo drug is an incredibly difficult. Massive costs and time-consuming nature of laboratory procedures would persuade pharmaceutical companies and research groups to take advantage of computational techniques as alternative methods. These methods are not utilized to design a new pharmaceutical product, but there are applied as a collaborative process for designing new drug in order to achieve the desired results [1]. Computational approaches called *in silico* methods are mostly applied to predict ligand target interactions (LTI) by assisting computer sciences [1,2].

In silico prediction methods are potent to speed rate of drug discovery and provide important tools for discovering two dimensional (2D) and three dimensional (3D) structures of new molecules and estimates of their biological activities [3]. Two basic approaches have been proposed for *in silico* LTI prediction. The first one is structure-based virtual screening (SBVS) and the second is ligand-based virtual screening (LBVS) [4]. The SBVS is helpful in condition to availability of the 3-D target structure. The LBVS is utilized in case the researcher lacks the knowledge of exact target structure. One of the major approaches used in LBVS research are known as quantitative structure activity relationship (QSAR) approach [5]. The simplest form of mathematical QSAR modeling can be defined as:

$$Y = f(x) + \ error \qquad (1)$$

where y is a vector of molecular biological activities and x is a matrix of molecular descriptors.

Molecular docking simulation is the most applied SBVS methods exclusively used to determine the ligand target interactions for a large number of compounds. LBVS approaches are statistical and mathematical methods based on information extracted from molecular structures classifying or predicting their biological activities (**Figure 1**) [6,7].

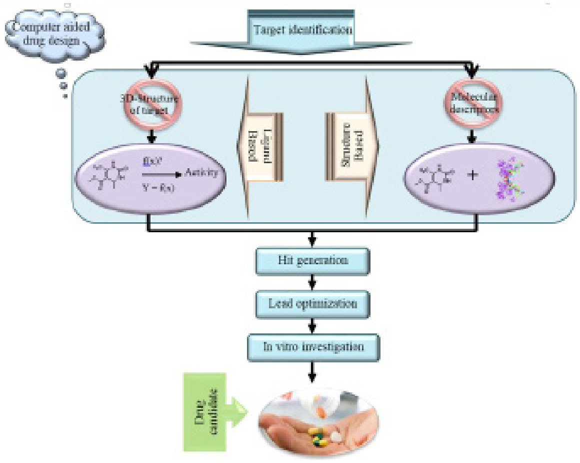


Figure 1: Schematic representation of drug design steps

2. Quantitative Structure Activity Relationship

Virtual screening (VS) is a turning point in drug discovery which can be an efficient complement for laboratory procedures to reduce experimental problems. Therefore, in many pharmaceutical companies, various academic and industrial projects were determined based on VS to estimate the biological activities of new molecules [1]. On the other hand, the major goal of VS is detecting the best chemical structures with the best conformer to interact with a special target. Ligand-based virtual screening (LBVS) introduced by Hansch *et al* in 1963 is one of these approaches to estimate the compound's performance when no information is in hand about the 3-D structure of the target [8].

In 1973, Hiller *et al* introduced neural network in LBVS. In their study, one-layer perceptron neural network was utilized to describe the structure of chemical compounds. It was indicated that neural network could be helpful for the classification of molecules into two categories: active and inactive [9]. Later on, in 1990, Aoyama *et al* applied neural network in decision making about compound interactions in contrast with multiple linear regression (MLR). They proved neural network as a multi-regression method with one neuron at the output layer to predict the molecular biological activity. In the QSAR models, diverse theoretical molecular descriptors derived from different molecular representations were utilized as an input model.

Molecular descriptors are the numerical information that encode physicochemical and structural features and plays a fundamental role in identifying model parameters. These descriptors have been generally fast to compute, making QSAR models suitable for initial biological activity prediction of large chemical sets and the prediction accuracy of the current models is far from being excellent. Hence, there is a need for next-generation, hyper-predictive QSAR models capable of facilitating the reliable screening of small focused chemical libraries of analogues in order to identify and prioritize the most promising chemicals. Interestingly, the recent development of GPU computing has dramatically decreased the resources needed to run molecular dynamics (MD) simulations of protein−ligand complexes (e.g., up to one microsecond of simulation per day on a GPU workstation). It was hypothesized that chemical descriptors computed from MD trajectories could be utilized to better characterize dynamic non-covalent protein−ligand interactions and thus build target specific QSAR models with enhanced prediction performances [10].

Prediction or partitioning of compounds' biological activities in all drug discovery approaches creates a theoretical framework for statistical machine learning techniques. Machine learning is a computer programming technique applicable in statistical and mathematical research. It is evolved from the study of pattern recognition and computational learning theory in artificial intelligence. In these statistical models, diverse theoretical molecular descriptors derived from different molecular representations encode physicochemical and structural features of the molecules [11]. This information plays a fundamental role in identifying model parameters. Thus, they are usually utilized to construct statistical models applicable for biological activity prediction [12,13].

Partitioning has always been a subject of debate in drug design when predicting drug-target interactions. Supervised and unsupervised learning techniques are two major partitioning approaches. These approaches are useful when the purpose in LBVS is splitting molecules into various categories. Supervised algorithms are constructed based on a molecular biological activity named classification technique. Unsupervised learning algorithm is built on the molecular descriptors regardless of data availability of a biological activity called clustering method. Over the previous two decades, numerous classification and clustering methods have turned into popular tools used in LBVS. Some of the most relevant machine learning algorithms are: K-nearest neighbors (KNN) [14-16], random forest (RF) [17-19], support vector machine (SVM) [20-22], Bayes classifier [23-25], kernel based methods such as Gaussian process [23], restricted Boltzmann machine (RBM) [26] and fuzzy clustering such as k-means algorithm [27]. One of the problems in clustering approach is determining cluster numbers, since it can extensively affect the accuracy of the predictions.

In the field of prediction, estimation of the biological activity of new compounds is the main goal while there is no information about their activity. Prediction techniques are based on both molecular descriptors and their biological activities in the training set and just molecular descriptors in the test set. Lots of linear and none linear models have been applied in QSAR approach. But, these methods mostly suffer from the same drawbacks, i.e. relying on a small number of ligands and a limited selection of descriptors, therefore, they are called shallow learning techniques. The training of these methods is simple, and they are applicable for few molecules and descriptors.

Artificial neural network (ANN) is one of the most popular non-linear methods for the prediction or classification of molecular bioactivity [28-30]. The years from the late 1990s to the early 2000s can be regarded as the golden period of using ANNs in computer-aided drug design. The efficacy of this technique depends on selecting the optimal features among molecular descriptors. In contrary to simplicity and runtime of the above-mentioned method, the facing non satisfaction result is the main disadvantage by the increment of molecular descriptor numbers. Thanks to several recently developed descriptor-generating softwares, thousands of descriptors are available for a large number of compounds being reported nowadays in literatures. Forasmuch as the shallow learning techniques are inefficient to model complex relationships between the molecular descriptors, deep architecture becomes essential. Parallel to this increment, deep neural network (DNN) which is a multilayer perceptron (MLP) network with many hidden layers and plenty of nodes in each layer could not overcome prone to over-fitting and getting stuck in local minima problems in drug discovery. The same is true in other research areas such as image processing and speech processing [31,32]. Therefore, deep learning (DL) techniques become essential when high amount of data are under process.

DL configuration is based on the hierarchical construction in which higher level features are founded on lower level ones. In fact, this approach comprises of multiple levels of linear and nonlinear operations. The number of these operations (depth model) refers to the longest path from an input node to an output one [33].

3. Molecular Descriptors as QSAR Models Input

There are several types of molecular descriptors depending on the method by which the molecular representation is transformed into a bit string. Most methods use only the 2D molecular graph and are thus called 2D fingerprints; however, some methods are capable of storing 3D information, most notably pharmacophore fingerprints and recently 4D finger print that generate for the ligand-target interaction in the drug discovery and enzyme activity. The main approaches are substructure keys-based fingerprints, topological or path-based fingerprints, and circular fingerprints [34].

Substructure keys-based fingerprints set the bits of the bit string depending on the presence in the compound of certain substructures or features from a given list of structural keys. This usually means that these fingerprints are most useful when used with molecules that are likely to be mostly covered by the given structural keys, but not so much when the molecules are unlikely to contain the structural keys, as their features would not be represented. Their number of bits is determined by the number of structural keys, and each bit relates to presence or absence of a single given feature in the molecule, which does not happen with other (hashed) types of fingerprints.

3.1. MACCS

MACCS comes in two variants, one with 960 and the other with 166 structural keys based on SMARTS patterns. The shorter one is the most commonly used, as it is relatively small in length (only 166 bits) but covers most of the interesting chemical features for drug discovery and virtual screening. Additionally, several software packages are able to calculate it, which is not true for the longer version [35].

3.2. PubChem fingerprint

PubChem fingerprint, with 881 structural keys covers a wide range of different substructures and features. It is the fingerprint used by PubChem for similarity searching and neighboring [36].

3.3. BCI fingerprints

BCI fingerprints can be generated using different numbers of bits and can be modified by the user in several ways, but the standard substructure dictionary includes 1052 keys [37].

3.4. TGD and TGT fingerprints

These are two-point and three-point pharmacophoric fingerprints calculated from a 2D molecular graph, which are implemented in the Molecular Operating Environment (MOE), consisting, respectively of 735 and 13,824 bits. TGD encodes atom-pair descriptors using seven-atom features and distances up to 15 bonds [37,38].

Topological or path-based fingerprints work by analyzing all the fragments of the molecule following a (usually linear) path up to a certain number of bonds, and then hashing every one of these paths to create the fingerprint. This means that any molecule can produce a meaningful fingerprint, and its length can be adjusted. They can also be used for fast substructure searching and filtering. These are hashed fingerprints, which means that a single bit cannot be traced back to a given feature. A given bit may be set by more than one different feature, which is called “bit collision”.

Circular fingerprints are also hashed topological fingerprints, but they are different in that instead of looking for paths in the molecule, the environment of each atom up to a determined radius is recorded. They are therefore not suitable for substructure queries (as the same fragment may have different environments) but are widely used for full structure similarity searching.

3.5. Molprint2D

Molprint2D encodes the atom environments of each atom of the molecular connectivity table, which are represented by strings of varying size. This fingerprint is available in several software packages, such as Open Babel and Compound Mapper.

In molprint2D, molecular similarity searching technique based on atom environments, information-gain-based feature selection, and the naive Bayesian classifier has been applied to a series of diverse datasets. Atom environments are count vectors of heavy atoms present at a topological distance from each heavy atom of a molecular structure. The atom environment approach appears to outperform fusion of ranking scores as well as binary kernel discrimination, which are both used in combination with Unity fingerprints [39].

3.6. Extended-connectivity fingerprints

The de facto standard circular fingerprints are the Extended-Connectivity Fingerprints (ECFPs), based on the Morgan algorithm, which were specifically designed for their use in structure–activity modeling [40]. The ECFP algorithm makes several changes to the standard Morgan algorithm. First, ECFP generation terminates after a predetermined number of iterations rather than after identifier uniqueness is achieved. The initial atom identifiers, and all identifiers after each iteration, are collected into a set; it is this set that defines the extended connectivity fingerprint. Second, since perfectly accurate disambiguation is not required, algorithmic optimizations are possible. Consider, for example, that in the standard Morgan process, the identifiers must be carefully recoded after each iteration to avoid mathematical overflow and possible “collision” (where two different atom environments are accidentally given the same identifier).

4. Molecular Descriptor Mining

Data mining is concerned with extracting the best piece of information from input database through data mapping in another space or discovering the most effective subset of features. On the other hand, data mining serves the purpose of preparing the applicable input network variables based on input database. Furthermore, selecting the appropriate technique is crucial when the input database contains irrelevant and redundant information.

There are two data mining approaches applied with different performances in QSAR: *i.e.* extracting the best descriptors among thousands of descriptors and mapping the data into another space. To extract the best descriptors, stepwise selection and genetic algorithm have been used. Also, principle component analysis (PCA) and self-organization map (SOM) are primary mapping data approaches [41,42].

4.1. Stepwise Regression

Stepwise descriptor selection or stepwise regression is a popular descriptor selection based on the combination of forward selection and backward elimination. In backward elimination, the algorithm starts by all variables, and then features drop with the highest p-value greater than 0.05, hierarchically [43]. Forward selection is reversing the backward elimination. In this method, features are added to the model step by step (**Figure 2-2**). In stepwise selection, making decision about the descriptors to be removed occurs later [9].

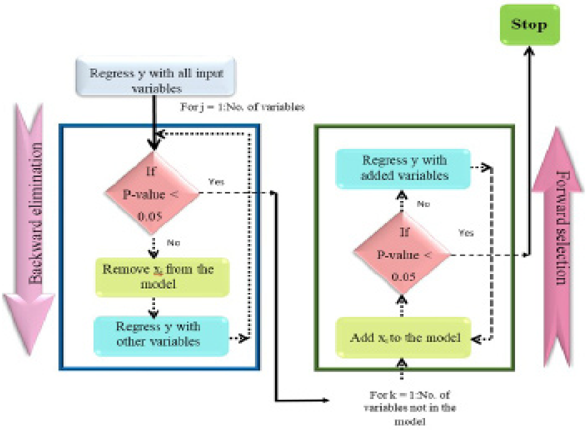

Stop
Regress y with all input variables
For j = 1:No. of variables
Backward elimination
If P-value < 0.05
Yes
No
Remove x_j from the model
Regress y with other variables
Regress y with added variables
No
If P-value < 0.05
Yes
Add x_i to the model
For k = 1:No. of variables not in the model
Forward selection

Figure 2: The main steps of stepwise selection

In addition to reducing network running time through few variables, stepwise method simplifies the interpretation of descriptors for the user. However, stepwise selection is not really efficient for QSAR methods as: (1) the relationship between descriptors is non-linear, (2) some of the optimal descriptors may be removed at add or drop step [44,45].

4.2. Genetic Algorithm

Genetic algorithm (GA) is a stochastic optimization method introduced by Holland in 1975 [46]. GA tried to discover optimal subset of descriptors as input network variables through random search in the descriptors space. In this method, each variable is named as a chromosome. Thus, at the first generation, all parameters are initialized by random values. Then the fitness of each chromosome is assessed by an objective function. In the reproduction step, parent selection, cross over and mutation are utilized to generate the first offspring. Finally, if the error is more than the determined threshold value, the second and third steps are repeated (**Figure 2-3**).

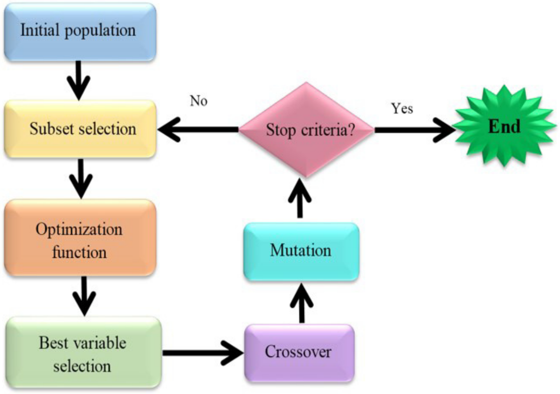


Figure 3: The main steps of genetic algorithm

Based on the unsupervised GA algorithm, a major advantage of hybrid network is being efficient in constructing a fully connected neural network. GA's weak point is randomized subset selection which leads to having various outputs [47,48].

4.3. Principle component analysis

Principle component analysis (PCA) is a statistical modeling method that maps the input variables to uncorrelated components named principle component through preserving the fundamental information. The number of principle components is usually less than input data, thus PCA is applied as a feature reduction method [49]. Nevertheless, principle components are computed based on the linear operation applied on the input variables, thus the output of neural network (in hybrid models) will probably be inefficient. Besides, the meaning of molecular descriptors based on the expert's opinion can be effective in achieving the best results [50,51].

4.4. Self-organization map

Self-organization map (SOM) or Kohenen network is a kind of artificial neural network (ANN) employed to train parameters as an unsupervised method. Unlike other learning network procedures used in error reduction learning such as back-propagation, SOM is utilized to transfer higher dimensional variables into lower dimensional views (usually 2D) called map. On the other hand, the main goal of SOM is to estimate the distribution of input pattern through preserving the topological properties of input surface (**Figure 2-4**) [52, 53].

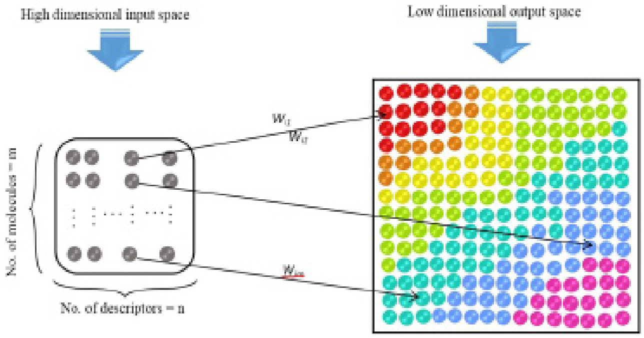


Figure 4: The schematic of SOM method

SOM network involves four main steps: initialization, competition, corporation and learning. At first, all network parameters were initialized by random variables. Then, the values of each hidden units are computed in the same way as feed forward neural network (to be described in the next section). The hidden unit with the smallest value is the winner. In the corporative step, the closest neighbors to the winner units tend to have the same behavior; this is called topological neighborhood. Finally, the feature map between input and output is formed. Therefore, parameters must be updated. These steps are repeated until the algorithm converges to the desirable value [54].

The main problem in SOM is initial parameters selection that is usually carried out randomly. Another disadvantage of this technique is representing molecular fields based on abstract nonphysical characters of such maps, which can hardly be understood by chemists and biologists outside QSAR community [55].

5. Learning Algorithms of Neural Network

The concept of neural network by McCulloch and Pitts (1940) was founded on human brain performance [56]. Afterwards, a number of supervised and unsupervised learning algorithms were proposed based on the neural network that made it a powerful technique with a wide range of applications in drug discovery.

All learning network algorithms rely on feed forward neural network (**Figure 2-5**). In this model, the minimum number of layers must be three: input, hidden and output layer. Molecular descriptors are applied as input neurons with the purpose of predicting biological activity or classification molecules. In order to compute hidden variables or output values, previous layer neurons are multiplied to their weights and summed by the biases. The calculated values are applied to the activation function.

$$y_i^l = f(b + \sum_j w_j^l y_j^{l-1}\Big) \qquad (2)$$

where y_i^l is related to the i^{th} unit of l^{th} layer. b and w are bias and weight, respectively. $f(.)$ is activation function. Figure 2-5 shows the scheme of deep neural network with two hidden layer.

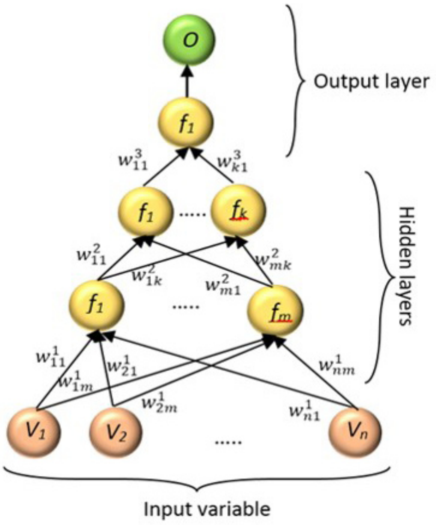


Figure 5: The structure of DNN with two hidden layers. Input and output layers are related to the ligand's descriptors and predicted biological activity, respectively and f_i indicates the sigmoid function.

5.1. Back-Propagation

Back-propagation (BP), a short form for “backward-propagation of errors”, is the first learning method in artificial neural network used in drug design by Aoyama *et al* (1990) [57]. It is commonly utilized in neural network in order to learn parameters, weights and biases, based on error derivative computation. In order to calculate the error, the predicted output must be computed in the same way as feed forward neural network. In this step, all input variables are multiplied by their weights and summed by their biases [58]. Sigmoid is a popular activation function used in back-propagation, defined as:

$$f(Y^l) = \frac{1}{1 + \exp\left(\sum\left(w_i y_i^l + b_i\right)\right)} \qquad (3)$$

This step is repeated for all layers until the outputs are predicted to minimize the error function, defined as:

$$E = \frac{1}{2}\sum\left\|Y_{real} - Y_{predicted}\right\|^2 \qquad (4)$$

Gradient descent method is a conventional optimization method challenged to minimize the error function in order to achieve the global error, defined as:

$$E = \frac{1}{2}\sum\left\|Y_{real} - Y_{predicted}\right\|^2 \qquad (5)$$

After each iteration, all network parameters are updated which is called back-error propagation. Update steps are defined as:

$$w_{i,new}^l = w_{i,old}^l + \Delta w_i \quad , \quad \Delta w_i = -\gamma \frac{\partial E}{\partial w_i} \qquad (6)$$

$$b_{i,new}^l = b_{i,old}^l + \Delta b_i \quad , \quad \Delta b_i = -\gamma \frac{\partial E}{\partial b_i} \qquad (7)$$

where $w_{i,new}^l$ and $b_{i,new}^l$ are related to the weights and biases of l^{th} layer (**Figure 2-6**) [59].

Two major problems occur in back-propagation algorithm:

1. Settling down in local minima is a critical problem when the number of input network variables is increased. On the other hand, the complex and non-linear relationship between descriptors introduces many local minima in error surface. Thus, the network may be trapped in one of these local minima and in the proximity of local minima to the global minimum have a direct effect on the output of neural network. Conventionally, initial network parameters are randomly selected. Thus, the situation of getting stuck in local minima occurs as a consequence of initial starting condition selection [60].

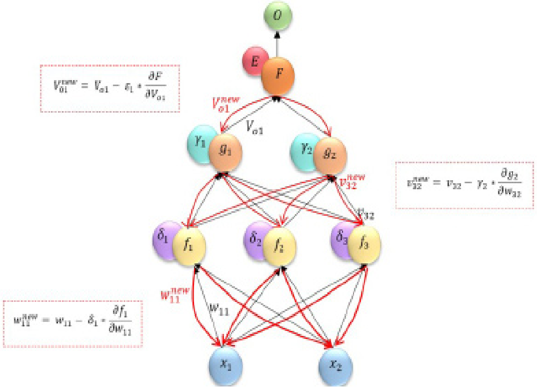

O
E
F
$V_{01}^{new} = V_{o1} - \varepsilon_1 * \frac{\partial F}{\partial V_{o1}}$
V_{o1}^{new}
V_{o1}
γ_1
g_1
γ_2
g_2
v_{32}^{new}
$v_{32}^{new} = v_{32} - \gamma_2 * \frac{\partial g_2}{\partial w_{32}}$
v_{32}
δ_1
f_1
δ_2
f_2
δ_3
f_3
w_{11}^{new}
w_{11}
$w_{11}^{new} = w_{11} - \delta_1 * \frac{\partial f_1}{\partial w_{11}}$
x_1
x_2

Figure 2.6: An example of back propagation algorithm with two ligands as input variables, two hidden layers and one output as activity.

2. Vanishing gradient in initial layers is another problem, when the parameters are updated in the back propagation step. On the other hand, tiny portions of gradient are realized in the primary layers [61].

5.2. Counter propagation

Counter propagation (CP) (1992) is a kind of hybrid neural network in QSAR utilized for the first time to predict Kovats indices for substituted phenols [62]. CP network is constructed based on the combination of Kohenen network and outstar structure of Grossberg [63]. Kohenen hidden layer and Grossberg layer are respectively utilized to determine winning units for input variables, and map the winners into their classes. On the other hand, learning procedure is carried out based on two different steps: (1) discovering the winner based on the Euclidean distance between input neurons and weights, (2) updating the winner's weight. CP guarantees to find the best weights instead of BP algorithm [64].

5.3. Radial basis function network

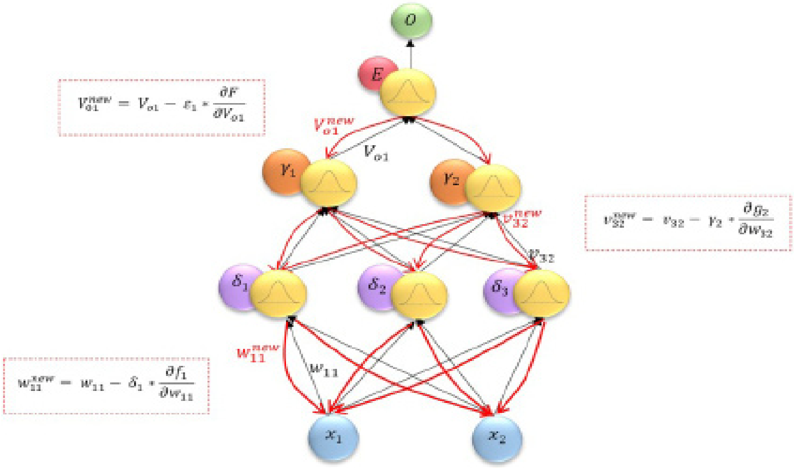
Radial basis function network (RBFN), a kind of neural network introduced by Broomhead and Lowe in (1988) [41], was used to predict boiling points from structural parameters by Lohninger (1993) [65]. In this network, RBF is used, in particular, as an activation function of learned algorithm based on two main steps:

1- Selecting parameters' centers, c_i. They are initialized by random values or calculated by *k*-means clustering algorithm.

2- Fine tuning network parameters by back propagation algorithm. The kernel function is commonly obtained by Gaussian distribution (**Figure 2-7**), defined as:

Simple computational learning and significantly reduced run of training time are the most significant advantages of RBFN [44]. Though random selection of center points and function weights lead to achieving unsatisfactory results [52].

$$f(x) = \sum w_i \,.\, \emptyset(x - c_i) \quad , \qquad \emptyset\,(x) = \exp(-\beta . x\Big) \qquad (8)$$



O
E
$V_{o1}^{new} = V_{o1} - \varepsilon_1 * \frac{\partial F}{\partial V_{o1}}$
V_{o1}^{new}
V_{o1}
γ_1
γ_2
$v_{32}^{new} = v_{32} - \gamma_2 * \frac{\partial g_2}{\partial w_{32}}$
v_{32}^{new}
v_{32}
δ_1
δ_2
δ_3
w_{11}^{new}
w_{11}
$w_{11}^{new} = w_{11} - \delta_1 * \frac{\partial f_1}{\partial w_{11}}$
x_1
x_2

Figure 2.7: An example of radial basis function neural network. As it is shown, it is the same as BP with Gaussian function as an activation function.

6. Deep Learning Algorithm

All learning algorithms of neural network mentioned above contain one or two hidden layers with limited number of units in each layer for feature transformation. These methods are useful to solve simple problems. However more complicated real world applications (such as a large number of descriptors as well as the non-linear relationship between them in drug discovery) forced the researchers to use deep learning algorithms instead of shallow learning methods. In this section, the main techniques proposed in deep learning are presented [66].

6.1. Deep belief network

Deep belief network (DBN) was the first proposed algorithm in deep architecture in order to fine tune network initial parameters, weights and biases, instead of neural network random variables, especially for back propagation algorithm [67]. DBN is a kind of generative unsupervised learning algorithm in which higher level features are constructed by lower level ones and composed of l stacks of restricted Boltzmann machine (RBM) (**Figure 2-8**). RBM is an energy-based method used as a discriminative or generative model for labeled or unlabeled data that has a single layer of hidden units with no internal layer of visible and hidden neurons (68). In this model, the probability of joint configuration *(l,h)* is defined as follows [69]:

where l and h is related to the input and hidden variables, respectively and $Z = \sum_{i,j} \exp\left(-Energy\left(l_i, h_j\right)\right)$ is called normalization factor. The derivative of probability equation logarithm has two main parts, defined as:

$$Pr(l,h) = \frac{exp(-Energy(l,h))}{Z} \qquad (9)$$

$$\frac{\partial\, log\big(P(v)\big)}{\partial\theta} = \varphi^{+} - \varphi^{-} \qquad (10)$$

where φ^{+} and$\varphi^{-}$are named as positive and negative phases, respectively. Due to lack of internal connection between visible or hidden units, estimating the positive phase is simple. Though the negative phase is difficult as it must be calculated for all visible and hidden units [70]. To solve this problem, Gibbs sampling method was suggested which is a Markov Chain Monte Carlo (MCMC) introduced by Geman (1984) to obtain a sequence of observations approximated from the joint probability distribution of two or more random variables [71]. This iterative approach is a randomizing algorithm and guarantees obtaining the best results from the model [72]. It is obvious that running this algorithm for many steps is too time consuming to be practical. Hinton *et al* (2006) introduced a fast greedy algorithm that quickly produced a fairly good set of parameters. This algorithm became the core of deep architectures with millions of parameters and many hidden layers [73]. The key factor of success for this algorithm was using contrastive divergence (CD) method instead of Gibbs sampling [74]. CD training algorithm is a special Gibbs sampling method that starts the Gibbs chain from real data points rather than random values. Based on the number of steps, CD algorithm is demonstrated as CD_n. CD_1 is fast but significantly different from the likelihood gradient. When there is enough time for computation, CD_10 is generally shown to be better [75]. Teileman (2008) presented another gradient approximation algorithm named persistent contrastive divergence (PCD) or stochastic maximum likelihood (SML). PCD is the same as Gibbs sampling algorithm in which is trained and tested with random variables but only on the first step, while for other iterations of Gibbs chain, it is initialized by the previous step rather than random values [76]. It was shown that this algorithm produces more meaningful feature detectors and outperforms other algorithms. In 2009, PCD was improved by Teileman and Hinton to provide fast PCD (FPCD) algorithm obtained by decoupling the parameters used in positive and negative phases [75]. In this model, adding the new parameter, "fast", to the update step caused the weights and biases to learn rapidly and the model to be optimized in order to approach the global minimum [77]. Thus, in this algorithm, the two main parts for updating parameters are $\theta_{regular}$ and θ_{fast} . In fact, FPCD will be the same as PCD if θ_{fast} is equal to zero. Two elements are used to assemble θ_{fast} and α [78].

$$\theta_{fast,new} = \alpha * \theta_{fast,old} + \varphi \qquad (11)$$

, where $\alpha = 0.95\,, \varphi = \varphi^{+} - \varphi^{-}$

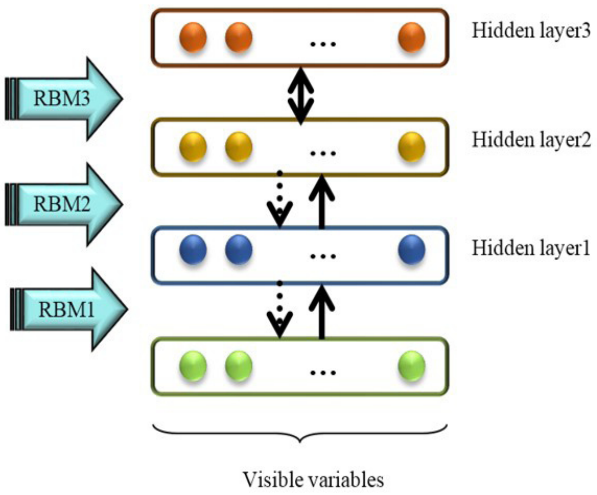


Figure 2-8: The schematic of DBN

6.2. Auto-encoder

The same as deep belief network, an auto-encoder is a kind of unsupervised learning algorithm applied layer by layer. The auto-encoder structure is based on two main parts, encode input data and decode the hidden units to reconstruct input data again (**Figure 2-9**). Thus, the number of output units should be equal to the number of input variables, which can be defined as:

$$\Psi\colon X' = f_2(W'Z + b') \qquad (12)$$

$$\emptyset\colon Z = f_1(WX + b) \qquad (13)$$

$$Error = \|X - X'\|^2 = \|X - f_2(W'.f_1(WX + b) + b'\|^2 \qquad (14)$$

The training algorithm of auto-encoder is utilized layer by layer, the same as RBM, thus it has been widely used as a building block to pre-train deep neural network [69-79].

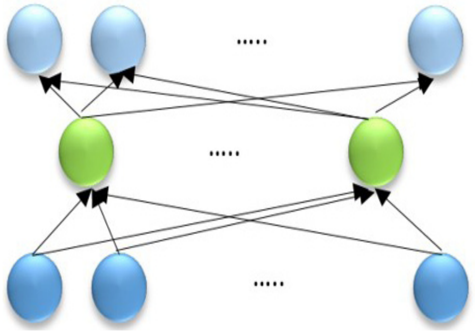


Figure 2-9. The simplest scheme of auto-encoder

6.3. Convolutional neural network

Convolutional neural network (CNN) is a kind of artificial neural network in which made up of neurons that have learnable weights and biases. In the image with the small number of pixels, using fully connected network could be manageable, but in the large image, network parameters (*e.g.* weights) would add up quickly and the network would quickly lead to over-fitting.

Unlike a feed forward neural network used only fully connected layers, convolutional neural network architectures have three different type of layers, convolutional layer, pooling layer and fully connected layer. The convolutional layers, ConvNet, utilized to extract feature from input neuron have *K* filters (or kernels) in which its size is smaller than the dimension of the image. On the other hand, ConvNet can compute the output of neurons connected to local regions in the input. After that, each map is then subsampled or down sampled typically with average, sum or max pooling over p × p neighbor pixels in which p is changed between 2 for small input data (*e.g.* MINIST images) and 5 for larger inputs. Fully connected layer is the same as feed forward neural network layers (**Figure 2-10**) [80,81].

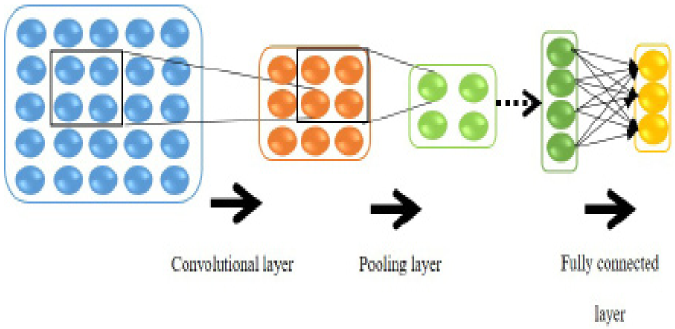


Figure 2-10: A schematic of convolutional neural network.

6.4. Drop-out

Drop-out is another popular and effective technique to improve the generalization error of large neural networks in order to reduce over-fitting problem. Unlike the auto-encoder and DBN that are used as unsupervised methods for fine tuning network initial parameters, drop out is utilized as a supervised network with end to end back propagation [82,83]. The key point in dropout technique is discarding random units from visible and hidden layer during training. These neurons are temporarily removed from the network (**Figure 2-11**) [84].

As described in the previous section, in normal forward neural network, input variables are multiplied by their weights and summed with their biases. After that, the hidden variables are computed by applying activation function on calculated values. This procedure is repeated for other layers. With dropout technique, before starting training algorithm, each input variables are multiplied to the Bernoulli dropout probability. For hidden layers, these steps are repeated. These procedures are given by:

$$r_j = Bernoulli(p) \qquad (15)$$

$$\widetilde{x_j} = r_j * x_j \qquad (16)$$

$$H_i = f(W * \tilde{X} + B) \qquad (17)$$

where p is the probability of j^{th} visible units; r_j is the j^{th} *Bernoulli* dropout probability, and f , W and B are activation function, weight matrix and bias vector, respectively [83].

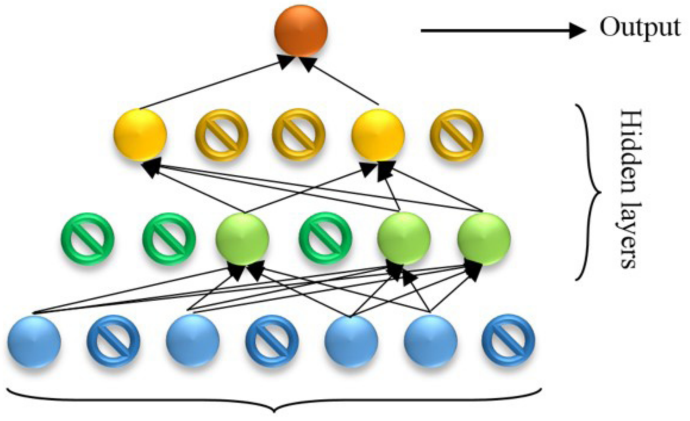


Figure 2-11: An example of a thinned neural network by applying dropout technique

6.5. Hessian free optimization

Hessian free optimization is a kind of *2nd* Newton optimization algorithm introduced by Martens (2010) which is used to optimize local quadratic approximations of objective function to make updates, iteratively [85]. A major advantage of hessian free optimization is eliminating gradient problem of back propagation [86]. The model of objective function is computed as:

$$f(\theta + \varepsilon) \cong M_\theta(\varepsilon) = f(\theta) + \nabla f(\theta)^T \varepsilon + \frac{1}{2}\, \varepsilon^T B \varepsilon \qquad (18)$$

where $\nabla f(\theta)$ is gradient of f, and B is an approximation to the Hessian matrix which quantifies curvature ($B = H + \mu I$). At the iteration (k+1) the new parameters are computed by:

$$\theta_{k+1} = \theta_k + \alpha_k \varepsilon_k^* \qquad \alpha_k \in [0,1] \qquad (19)$$

$$\varepsilon_k^* = -B^{-1} \nabla f(\theta_{k-1}) \qquad (20)$$

Based on the above equations, the 3rd equation must be computed for $f(\theta + \varepsilon)$ though sometimes unfeasible. Conjugate gradient algorithm is applied as the solution of hessian free optimization method to minimize the quadratic objective defined as [87]:

$$q(\varepsilon) = \varepsilon_k^* = \frac{1}{2}\, \varepsilon^T B \varepsilon + \nabla f(\theta_{k-1})^T \varepsilon \qquad (21)$$

6.5.1. Rectified linear units

Rectified linear unit (ReLU) is a kind of activation function widely used as an alternative to sigmoid function in deep neural networks or deep belief networks. In this method, the hidden units are sampled from [82]:

$$h = \max(0, x) \qquad (22)$$

where x is the input unit. In some cases, the Gaussian noise (with sigmoid variant) is added to ReLU which makes up NReLU in which the hidden variables are sampled from:

$$h = \max(0, x + \varepsilon) \tag{23}$$

where ε is extracted from normal distribution with a mean of zero and variance of $\frac{1}{1+e^{-x}}$.

6.6. Conditional restricted boltzmann machine

Probabilistic model has recently been enriched by conditional restricted Boltzmann machine (CRBM) and is widely used in various fields of research (e.g. classification and collaborative filtering). Though, there are two major problems challenged in RBM: it is not applicable for conditional model, and the output space is structured arbitrarily. In this method, the energy is defined as:

$$E(v,h,u) = -v^T W^{vh} h - v^T b^v - u^T w^{uv} v - u^T W^{uh} h - h^T b^h \tag{24}$$

Thus, the free energy is given by:

$$F(v,u) = -\log(\sum_h \exp(-E(v,h,u))\Big) \tag{25}$$

The CRBM could be used to train based on another energy-based model (88). In this model, the probability distribution is defined as:

$$P(v|u) = \frac{\exp(-F(v,u))}{\sum_{v'} \exp(-F(v',u))} \tag{26}$$

7. Conclusion

This study was attempted to present the most comprehensive literature review of QSAR studies, including molecular descriptors as input model, data mining approaches utilized in drug discovery, proposed neural network models in QSAR studies containing each proposed method's merits and drawbacks, and finally deep learning algorithms. It can be concluded that in biological activity prediction using neural network, there are two major issues, which are the consideration of high throughput virtual screening based on thousands of molecules and limited number of compounds, usually less than 100 molecules.

In the first issue, a large number of compounds with thousands of descriptors leads to use of deep learning architecture. On the other hand, a large number of input data requires a network with a large number of computational elements. That's why in QSAR the focus is on finding the best model based on deep architecture to avoid the situation of getting stuck in local minima and being prone to over-fitting.

In the second issue, a different situation is encountered: the small number of compounds commonly with thousands of descriptors ends up in opening a new problem for using deep learning architecture, redundancy and over-fitting. Therefore, Data mining algorithms to reduce number of descriptors were suggested.

8. References

1. Leelananda SP, Lindert S. Computational methods in drug discovery. Beilstein Journal of Organic Chemistry. 2016;12(1):2694-718.

2. Young DC. Computational drug design: a guide for computational and medicinal chemists: John Wiley & Sons; 2009.

3.Huang B. MetaPocket: a meta approach to improve protein ligand binding site prediction. OMICS A Journal of Integrative Biology. 2009;13(4):325-30.

4. Hendlich M, Rippmann F, Barnickel G. LIGSITE: automatic and efficient detection of potential small molecule-binding sites in proteins. Journal of Molecular Graphics and Modelling. 1997;15(6):359-63.

5. Levitt DG, Banaszak LJ. POCKET: A computer graphies method for identifying and displaying protein cavities and their surrounding amino acids. Journal of molecular graphics. 1992;10(4):229-34.

6. Unterthiner T, Mayr A, Klambauer G, Steijaert M, Wegner JK, Ceulemans H, et al. Deep Learning for Drug Target Prediction. 2014.

7. Ghasemi F, Mehri A, Peña-García J, den-Haan H, Pérez-Garrido A, Fassihi A, et al. Improving Activity Prediction of Adenosine A2B Receptor Antagonists by Nonlinear Models. Bioinformatics and Biomedical Engineering: Springer; 2015. p. 635-44.

8. Todeschini R, Consonni V. Frontmatter: Wiley Online Library; 2000.

9. Hiller S, Golender V, Rosenblit A, Rastrigin L, Glaz A. Cybernetic methods of drug design. I. Statement of the problem—the perceptron approach. Computers and Biomedical Research. 1973;6(5):411-21.

10. Ataide Martins JP, Rougeth de Oliveira MA, Oliveira de Queiroz MS. Web-4D-QSAR: A web-based application to generate 4D-QSAR descriptors. Journal of computational chemistry. 2018;39(15):917-24.

11. Gawehn E, Hiss JA, Schneider G. Deep Learning in Drug Discovery. Molecular Informatics. 2015.

12. Ewing TJ, Makino S, Skillman AG, Kuntz ID. DOCK 4.0: search strategies for automated molecular docking of flexible molecule databases. Journal of computer-aided molecular design. 2001;15(5):411-28.

13. Miller MD, Kearsley SK, Underwood DJ, Sheridan RP. FLOG: a system to select 'quasi-flexible'ligands complementary to a receptor of known three-dimensional structure. Journal of computer-aided molecular design. 1994;8(2):153-74.

14. Itskowitz P, Tropsha A. k nearest neighbors QSAR modeling as a variational problem: theory and applications. Journal of chemical information and modeling. 2005;45(3):777-85.

15. Ajmani S, Jadhav K, Kulkarni SA. Three-dimensional QSAR using the k-nearest neighbor method and its interpretation. Journal of chemical information and modeling. 2006;46(1):24-31.

16. Nigsch F, Bender A, van Buuren B, Tissen J, Nigsch E, Mitchell JB. Melting point prediction employing k-nearest neighbor algorithms and genetic parameter optimization. Journal of chemical information and modeling. 2006;46(6):2412-22.

17. Zilian D, Sotriffer CA. SFCscore RF: a random forest-based scoring function for improved affinity prediction of protein–ligand complexes. Journal of chemical information and modeling. 2013;53(8):1923-33.

18. Ma J, Sheridan RP, Liaw A, Dahl GE, Svetnik V. Deep Neural Nets as a Method for Quantitative Structure–Activity

Relationships. Journal of chemical information and modeling. 2015;55(2):263-74.

19. Kuz'min VE, Polishchuk PG, Artemenko AG, Andronati SA. Interpretation of QSAR models based on random forest methods. Molecular Informatics. 2011;30(6-7):593-603.

20. Shahlaei M, Sabet R, Ziari MB, Moeinifard B, Fassihi A, Karbakhsh R. QSAR study of anthranilic acid sulfonamides as inhibitors of methionine aminopeptidase-2 using LS-SVM and GRNN based on principal components. European journal of medicinal chemistry. 2010;45(10):4499-508.

21. Shahlaei M, Fassihi A. QSAR analysis of some 1-(3, 3-diphenylpropyl)-piperidinyl amides and ureas as CCR5 inhibitors using genetic algorithm-least square support vector machine. Medicinal Chemistry Research. 2013;22(9):4384-400.

22. Pérez-Sánchez H, Cano G, García-Rodríguez J. Improving drug discovery using hybrid softcomputing methods. Applied Soft Computing. 2014;20:119-26.

23. Trott O, Olson AJ. AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. Journal of computational chemistry. 2010;31(2):455-61.

24. Schneider G, Böhm H-J. Virtual screening and fast automated docking methods. Drug Discovery Today. 2002;7(1):64-70.

25. www.kaggle.com.

26. Wang Y, Zeng J. Predicting drug-target interactions using restricted Boltzmann machines. Bioinformatics. 2013;29(13):i126-i34.

27. Timón I, Soto J, Pérez-Sánchez H, Cecilia JM. Parallel implementation of fuzzy minimals clustering algorithm. Expert Systems with Applications. 2016;48:35-41.

28. Dahl GE, Jaitly N, Salakhutdinov R. Multi-task neural networks for QSAR predictions. arXiv preprint arXiv:14061231. 2014.

29. Lowe R, Mussa HY, Mitchell JB, Glen RC. Classifying molecules using a sparse probabilistic kernel binary classifier. Journal of chemical information and modeling. 2011;51(7):1539-44.

30. Erić S, Kalinić M, Popović A, Zloh M, Kuzmanovski I. Prediction of aqueous solubility of drug-like molecules using a novel algorithm for automatic adjustment of relative importance of descriptors implemented in counter-propagation artificial neural networks. International journal of pharmaceutics. 2012;437(1):232-41.

31. Hinton G, Deng L, Yu D, Dahl GE, Mohamed A-r, Jaitly N, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. Signal Processing Magazine, IEEE. 2012;29(6):82-97.

32. Cerón-Carrasco JP, Coronado-Parra T, Imbernón-Tudela B, Banegas-Luna AJ, Ghasemi F, Vegara-Meseguer JM, et al. Application of Computational Drug Discovery Techniques for Designing New Drugs against Zika Virus. Drug Designing: Open Access. 2016:1-2.

33. Bengio Y. Learning deep architectures for AI. Foundations and trends® in Machine Learning. 2009;2(1):1-127.

34. Cereto-Massagué A, Ojeda MJ, Valls C, Mulero M, Garcia-Vallvé S, Pujadas G. Molecular fingerprint similarity search in virtual screening. Methods. 2015;71:58-63.

35. Durant JL, Leland BA, Henry DR, Nourse JG. Reoptimization of MDL keys for use in drug discovery. Journal of chemical information and computer sciences. 2002;42(6):1273-80.

36. Kim S, Thiessen PA, Bolton EE, Chen J, Fu G, Gindulyte A, et al. PubChem substance and compound databases. Nucleic acids research. 2015;44(D1):D1202-D13.

37. Tovar A, Eckert H, Bajorath J. Comparison of 2D fingerprint methods for multiple-template similarity searching on compound activity classes of increasing structural diversity. ChemMedChem: Chemistry Enabling Drug Discovery. 2007;2(2):208-17.

38. Sheridan RP, Miller MD, Underwood DJ, Kearsley SK. Chemical similarity using geometric atom pair descriptors. Journal of chemical information and computer sciences. 1996;36(1):128-36.

39. Bender A, Mussa HY, Glen RC, Reiling S. Similarity searching of chemical databases using atom environment descriptors (MOLPRINT 2D): evaluation of performance. Journal of chemical information and computer sciences. 2004;44(5):1708-18.

40. Rogers D, Hahn M. Extended-connectivity fingerprints. Journal of chemical information and modeling. 2010;50(5):742-54.

41. Broomhead DS, Lowe D. Radial basis functions, multi-variable functional interpolation and adaptive networks. DTIC Document; 1988.

42. Schneider G, Wrede P. Artificial neural networks for computer-based molecular design. Progress in biophysics and molecular biology. 1998;70(3):175-222.

43. Melssen W, Smits J, Buydens L, Kateman G. Using artificial neural networks for solving chemical problems: Part II. Kohonen self-organising feature maps and Hopfield networks. Chemometrics and intelligent laboratory systems. 1994;23(2):267-91.

44. Schilling RJ, Carroll JJ, Al-Ajlouni AF. Approximation of nonlinear systems with radial basis function neural networks. IEEE Transactions on neural networks. 2001;12(1):1-15.

45. Davis AM, Teague SJ, Kleywegt GJ. Application and Limitations of X-ray Crystallographic Data in Structure-Based Ligand and Drug Design. Angewandte Chemie International Edition. 2003;42(24):2718-36.

46. Rose VS, Croall IF, Macfie HJ. An Application of Unsupervised Neural Network Methodology Kohonen Topology-Preserving Mapping to QSAR Analysis. Quantitative Structure-Activity Relationships. 1991;10(1):6-15.

47. Xue L, Bajorath J. Molecular descriptors in chemoinformatics, computational combinatorial chemistry, and virtual screening. Combinatorial Chemistry & High Throughput Screening. 2000;3(5):363-72.

48. Whitley DC, Ford MG, Livingstone DJ. Unsupervised forward selection: a method for eliminating redundant variables. Journal of chemical information and computer sciences. 2000;40(5):1160-8.

49. Kubiny H. Variable selection in QSAR studies. I. An evolutionary algorithm. Quantitative Structure-Activity Relationships. 1994;13(3):285-94.

50. Vendrame R, Braga R, Takahata Y, Galvao D. Structure-activity relationship studies of carcinogenic activity of polycyclic aromatic hydrocarbons using calculated molecular descriptors with principal component analysis and neural network methods. Journal of chemical information and computer sciences. 1999;39(6):1094-104.

51. Barlow TW. Self-organizing maps and molecular similarity. Journal of molecular graphics. 1995;13(1):24-7.

52. Chen S, Cowan CF, Grant PM. Orthogonal least squares learning algorithm for radial basis function networks. IEEE Transactions on neural networks. 1991;2(2):302-9.

53. van Nostrum CF, Bosman AW, Gelinck GH, Schouten PG, Warman JM, Kentgens AP, et al. Supramolecular Structure, Physical Properties, and Langmuir-Blodgett Film Formation of an Optically Active Liquid-Crystalline Phthalocyanine. Chemistry–A European Journal. 1995;1(3):171-82.

54. Bradbury SP. Predicting modes of toxic action from chemical structure: an overview. SAR and QSAR in Environmental Research. 1994;2(1-2):89-104.

55. Yasri A, Hartsough D. Toward an optimal procedure for variable selection and QSAR model building. Journal of Chemical Information and Computer Sciences. 2001;41(5):1218-27.

56. McCulloch WS, Pitts W. A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics. 1943;5(4):115-33.

57. Aoyama T, Suzuki Y, Ichikawa H. Neural networks applied to pharmaceutical problems. III. Neural networks applied to quantitative structure-activity relationship (QSAR) analysis. Journal of medicinal chemistry. 1990;33(9):2583-90.

58. Sun H. A naive Bayes classifier for prediction of multidrug resistance reversal activity on the basis of atom typing. Journal of medicinal chemistry. 2005;48(12):4031-9.

59. Werbos PJ. The roots of backpropagation: from ordered derivatives to neural networks and political forecasting: John Wiley & Sons; 1994.

60. Suresh H, Puttamadappa C. Removal OF EMG and ECG artifacts from EEG based on real time recurrent learning algorithm. International journal of physical sciences. 2008;3(5):120-5.

61. Sutskever I, Martens J, Dahl G, Hinton G, editors. On the importance of initialization and momentum in deep learning. Proceedings of the 30th international conference on machine learning (ICML-13); 2013.

62. Peterson KL. Counter-propagation neural networks in the modeling and prediction of Kovats indexes for substituted phenols. Analytical Chemistry. 1992;64(4):379-86.

63. Hecht-Nielsen R. Applications of counterpropagation networks. Neural networks. 1988;1(2):131-9.

64. Wu C, Shivakumar S. Back-propagation and counter-propagation neural networks for phylogenetic classification of ribosomal RNA sequences. Nucleic acids research. 1994;22(20):4291-9.

65. Lohninger H. Evaluation of neural networks based on radial basis functions and their application to the prediction of boiling points from structural parameters. Journal of Chemical Information and Computer Sciences. 1993;33(5):736-44.

66. Deng L, Yu D. Deep Learning. Signal Processing. 2014;7:3-4.

67. Deng L, Hinton G, Kingsbury B, editors. New types of deep neural network learning for speech recognition and related applications: An overview. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing; 2013: IEEE.

68. Erhan D, Bengio Y, Courville A, Manzagol P-A, Vincent P, Bengio S. Why does unsupervised pre-training help deep learning? The Journal of Machine Learning Research. 2010;11:625-60.

69. Bengio Y, Lamblin P, Popovici D, Larochelle H. Greedy layer-wise training of deep networks. Advances in neural information processing systems. 2007;19:153.

70. Hinton G. A practical guide to training restricted Boltzmann machines. Momentum. 2010;9(1):926.

71. Geman S, Geman D. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. Pattern Analysis and Machine Intelligence, IEEE Transactions on. 1984(6):721-41.

72. Hinton GE. Training products of experts by minimizing contrastive divergence. Neural computation. 2002;14(8):1771-800.

73. Salakhutdinov R, Hinton GE, editors. Deep Boltzmann Machines. AISTATS; 2009.

74. Hinton GE, Osindero S, Teh Y-W. A fast learning algorithm for deep belief nets. Neural computation. 2006;18(7):1527-54.

75. Tieleman T, Hinton G, editors. Using fast weights to improve persistent contrastive divergence. Proceedings of the 26th Annual International Conference on Machine Learning; 2009: ACM.

76. Tieleman T, editor Training restricted Boltzmann machines using approximations to the likelihood gradient. Proceedings of the 25th international conference on Machine learning; 2008: ACM.

77. Breuleux O, Bengio Y, Vincent P. Quickly generating representative samples from an rbm-derived process. Neural Computation. 2011;23(8):2058-73.

78. Ghasemi FF, Afshin;Preze Sunchez, Horacio Emilio;Mehridehnavi, Alireza. The role of Different Sampling Methods in Improving Biological Activity Prediction Using Deep Belief Network Journal of computational chemistry. 2016.

79. Deng L, Seltzer ML, Yu D, Acero A, Mohamed A-r, Hinton GE, editors. Binary coding of speech spectrograms using a deep auto-encoder. Interspeech; 2010: Citeseer.

80. Krizhevsky A, Sutskever I, Hinton GE, editors. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems; 2012.

81. Kalchbrenner N, Grefenstette E, Blunsom P. A convolutional neural network for modelling sentences. arXiv preprint arXiv:14042188. 2014.

82. Dahl GE. Deep learning approaches to problems in speech recognition, computational chemistry, and natural language text processing: University of Toronto; 2015.

83. Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR. Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:12070580. 2012.

84. Srivastava N, Hinton GE, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research. 2014;15(1):1929-58.

85. Martens J, editor Deep learning via Hessian-free optimization. Proceedings of the 27th International Conference on Machine Learning (ICML-10); 2010.

86. Kiros R. Training neural networks with stochastic hessian-free optimization. arXiv preprint arXiv:13013641. 2013.

87. Martens J, Sutskever I. Training deep and recurrent networks with hessian-free optimization. Neural networks: Tricks of the trade: Springer; 2012. p. 479-535.

88. Mnih V, Larochelle H, Hinton GE. Conditional restricted Boltzmann machines for structured output prediction. arXiv preprint arXiv:12023748. 2012.