# Particle Swarm Optimization and Neural Network Application for QSAR

Zhiwei Wang
*ECE Department, IUPUI*

Gregory L. Durst
*Eli Lilly and Company*

Russell C. Eberhart
*ECE Department, IUPUI*

Donald B. Boyd
*Chemistry Department, IUPUI*

Zina Ben Miled
*ECE Department, IUPUI*

## Abstract

*A successful approach to building QSAR models was proposed by other researchers. It uses binary particle swarm optimization (BPSO) for feature selection in the first stage, and a back propagation neural network in the second stage to generate a QSAR model based on the features selected in the first stage.*

*This paper starts by re-establishing the results of this approach on an extended number of data sets. A new method is then proposed that addresses the limitation of back propagation. The new approach uses particle swarm optimization (PSO) in the second stage for training and bootstrap aggregation (Bagging) in order to overcome the instability of PSO. The proposed approach yields robust QSAR models, while reducing the variability due to the choice of the back propagation parameters.*

## 1. Introduction

A structure-activity study can indicate which features of a given molecule correlate with its activity, thus making it possible to synthesize new and more potent compounds with enhanced biological activities. QSAR analysis is based on the assumption that the behavior of compounds is correlated to the characteristics of their structure[1]. In general, a QSAR model is represented as follows:

$$BioActivity = C_0 + (C_1 \times P_1) + \cdots + (C_n \times P_n) \qquad (1)$$

where the parameters $P_1$ through $P_n$ are a set of measured (or computed) properties of the compounds and $C_0$ through $C_n$ are the calculated coefficients of the QSAR model.

Several approaches have been previously proposed for the development of QSAR models. Linear regression has been one of the more successful techniques used to construct QSAR models [1] [2]. However, even with moderate numbers of features this technique can result in over-fitting [1]. In order to avoid over-fitting, linear regression is often used in combination with principal component analysis (PCA) [3].

Recently, neural networks and genetic algorithms were found to be efficient in constructing QSAR models [4] [ 5 ]. The advantage of using a non-linear method compared to a linear method such as linear regression is that more complex and non-linear QSAR models can be derived, which in turn can better reflect the possible relationship between the features of the molecule and its activity. In this paper, we propose to investigate the use of PSO and neural networks [6] in the construction of QSAR models.

A successful and scalable approach to generating QSAR models has been previously proposed by Agrafiotis and Cedeno [7]. This approach uses Binary Particle Swarm Optimization (BPSO) for feature selection followed by a neural network which is trained using back propagation (BP) for the construction of the QSAR model. The effectiveness of this approach was demonstrated on three data sets. In [7], the BPSO-BP method was compared with simulated annealing and it was shown that BPSO was capable of discovering a better and more diverse set of solutions than simulated annealing. The major disadvantage of BPSO-BP is the difficulty in choosing parameters for the back propagation that can ensure efficient network training. For example, the tests described in Section 3 show how an inadequate choice for the values of the weight updating parameters (e.g.., the learning rate) can result in poorly-trained QSAR models. We address this limitation by using PSO instead of back propagation as a training technique for the neural network that constructs the QSAR models. However, during our investigation, we established that while PSO effectively addresses the issues related to the neural network parameters, it yields QSAR models that may be unstable. This is not unique to models developed by PSO. All neural network models have the potential to exhibit instability. In order to capitalize on this instability, bootstrap aggregation (Bagging) [8] is used. Bagging is a technique that combines the "opinions" of multiple

models in such a way that the aggregated results are more predictive and robust. The model instability is key to obtaining the various "opinions."

## 2. Background

This section briefly presents an overview of PSO, Binary PSO (BPSO) and neural network technology. The benchmark data sets and a description of the process underlying the experimental validation for the proposed approach are also described.

### 2.1 PSO

PSO is a non-linear method which falls under the class of evolutionary computation techniques. Particle swarms explore the search space through a population of particles, which adapt by returning to previously successful regions [6]. The movement of the particles is stochastic; however it is influenced by the particle's own memories as well as the memories of its peers. Each particle keeps track of its coordinates in the problem space. PSO also keeps track of the best solution for all the particles (*gbest*) achieved so far, as well as the best solution (*pbest)* achieved so far by each particle. At the end of a training iteration, PSO changes the velocity of each particle toward its *pbest* and the current *gbest* value. The individual velocity is updated by using the following equation:

$$v_i(t+1) = \omega v_i(t) + \eta_1 r[p_i - x_i(t)] + \eta_2 r[p_{b(i)} - x_i(t)] \quad (2)$$

where $v_i$ is the current velocity of the $i^{th}$ particle, $p_i$ is the position with the best fitness value visited by the $i^{th}$ particle, and *b(i)* is the particle with the best fitness among all the particles. Each particle is updated by using the following equation:

$$x_i(t+1) = x_i(t) + v_i(t) \quad (3)$$

In the proposed approach, PSO is used to evolve the weights of a neural network that generates a QSAR model. PSO is initialized so that each dimension of the particle represents a weight of the link connecting two processing elements (PEs) in the network. PSO tries to minimize the error between the target values and predicted values of the biological activities of the compounds. At the end of each iteration, the smallest fitness value is remembered by PSO, and the corresponding particle is retained as *gbest*.

### 2.2 BPSO

The PSO technique described above is the real valued PSO, whereby each dimension can take on any real valued number. On the other hand, in Binary PSO (BPSO), the technique described in this section, each dimension of the particle can only take on the discrete values of 0 or 1.

In the proposed approach, BPSO is used in the first stage for feature selection. The input presented to the network consists of a matrix where the rows represent chemical compounds and the columns correspond to molecular descriptors. Each compound has a value for a given descriptor. An ideal QSAR model will be able to accurately predict the biological activity of the compounds based on their values for a subset of the descriptors. In the remainder of the paper, the terms "descriptors" and "features" will be used interchangeably.

The $x_{ij}^{th}$ dimension of the $i^{th}$ particle can only take on the values 0 or 1 indicating whether the $j^{th}$ feature is selected or not. The dimensionality of the particle is equal to the total number of features.

In Equation 3, $x_i$ is the current position of the $i^{th}$ particle. Initially $x_{ij}$ is a real number. After the update step, $x_{ij}$ is converted to a binary value using probabilistic selection, which is the fractional value of $x_{ij}$ and is treated as a probability threshold that determines the subset membership. Each feature (i.e., each dimension of the particle) is assigned a slice of a roulette wheel whose size is proportional to its value $x_{ij}$. The subset of selected features is obtained by spinning the wheel and selecting the features to which the marker points. Only a predefined number of features can be selected. A total of $k$ spins are performed, which enables $k$ features to be selected. The selected dimensions are set to 1, the remaining dimensions are set to 0. The actual probabilities, $p_{ij}$, for each dimension are computed as follows:

$$p_{ij} = \frac{x_{ij}^{\ a}}{\sum_{j=1}^{n} x_{ij}^{\ a}} \quad (4)$$

where $x_{ij}$ is the fractional coordinate obtained by applying Equation 3, and $a$ is a scaling factor referred to as a selection pressure. In this paper, $a$ is set to 2 [7].

As will be discussed in Section 3, the computational advantage of BPSO is that the near-optimal solutions could be found much faster than by using a random search. This feature allows BPSO to perform feature selection efficiently in data sets with large numbers of features.

### 2.3 Neural networks and back propagation

A neural network has two elementary components: processing elements (PEs) and connection weights. Back propagation is one of the methods that can be used to update the weights of a neural network during training. This is the training technique that was used in [7] for the successful construction of QSAR models.

Weight adjustment between PEs in back propagation is carried out according to the difference between the target value and the output value of the neural network. In back

propagation, the difference of the error is measured by the mean square error, as shown below:

$$E = \sum_{k=1}^{m} \sum_{j=1}^{q} (t_{kj} - z_{kj})^2 \qquad (5)$$

where $t_{kj}$ is the $j^{th}$ target value of the $k^{th}$ compound, and $z_{kj}$ is the output.

The weights are adjusted toward the gradient direction that produces a better fitness [6] as shown in the following equation:

$$w_{ji}^{new} = w_{ji}^{old} + \alpha \sum_{k} \delta_{kj} y_{ki} + \beta \Delta w_{ji}^{old} \quad (6)$$

where $j$, $i$ are the indices of the adjacent layers, $w_{ji}$ is the weight from the $i^{th}$ PE in the previous layer to the $j^{th}$ PE in the current layer and $\Delta w_{ji}^{old}$ is the previous weight change. The variable $y_{ki}$ represents the $i^{th}$ output for the $k^{th}$ pattern. The parameters $\alpha$ and $\beta$ are positive constants called learning rate and momentum rate respectively. They control the amount of weight adjustments during the weight update process [6].

## 2.4 Data sets

Four data sets were used in this study. The Selwood data set [9] has 31 compounds with 53 features for each compound, and a set of corresponding antifilarial antimycin activities. The activity is measured as $-[log(EC_{50})]$, where $EC_{50}$ is the concentration of an analog needed to reduce the concentration of adenine in cell lysate by 50%. This data set has been used by many others to construct QSAR models using a variety of techniques [4,9,10,11,12]. It was also used to test feature selection by using BPSO for feature selection and a neural network for building the QSAR models that are trained by using back propagation [7]. This approach was compared to simulated annealing and was found in [7] to be able to identify a better and more diverse set of solutions.

The BEN data set [13] has 57 compounds with 42 features for each compound. The biological activity is expressed as the binding affinities for the benzodiazepine $GABA_A$ receptor preparations. This data set was also used in [7].

The Breneman data set [14] has 64 compounds with 428 features for each compound. Among the data sets that are used in this paper, the Breneman data set has the largest number of features. In a previous study by other researchers [15], this HIV related data set was used to perform feature reduction. The model proposed in [15] reduced the number of features to 35, and the results proved to be better than the full feature set [15].

The dihydrofolate reductase (DHFR) data set [16] has 256 compounds, with 13 features for each compound. The biological activity is measured by the concentration that inhibits the dihydrofolate reductase enzyme. This data set was previously used by others to construct QSAR models by using neural network and multiple linear regression methods [16].

In the implementation of back propagation, each input can only take on a real value between 0 and 1 [6]. Therefore, the feature values and the biological activities in the data sets have to be scaled to values between 0 and 1. However, because networks cannot train when the output value is 0, the biological activity had to be scaled between 0.001 and 1.

When PSO is used to update the weights of a neural network, the input data does not have to be scaled [6]. This finding was verified on the data sets used in this paper. However, as previously mentioned, scaling is necessary for back propagation, and therefore, for comparison purposes, scaling was also used with PSO.

## 2.5 Evaluation Method

Leave-n-out testing, which is regarded as an indication of the generalization ability of the QSAR model [2], was used to assess the quality of the models obtained by the various approaches. Each time, only a subset of the compounds is used in the training process. For each data set, approximately 10%-15% of the compounds were left out for testing. The experiment is carried out multiple times until each compound is left out once. The predictive ability of the network is measured by the average error of the leave-n-out compounds. Equation 7 shows how this average error is calculated.

$$AverageError = \frac{1}{n} \sum_{i} |T_i - O_i| \qquad (7)$$

where $T_i$ is the target value of the $i^{th}$ compound, and $O_i$ is the output value of the $i^{th}$ compound. The overall average error is calculated for all the compounds (i.e., both testing and training compounds), and the average testing error is the average error for the $n$ compounds that are left out for testing.

The correlation coefficient is also an index that can be used to measure the quality of the QSAR models generated by the various approaches. Equation 8 shows the Pearson correlation coefficient

$$r = \frac{n(\sum x_i y_i) - (\sum x_i)(\sum y_i)}{\sqrt{\left[n(\sum x_i^2) - (\sum x_i)^2\right] \times \left[n(\sum y_i^2) - (\sum y_i)^2\right]}} \qquad (8)$$

where $n$ is the number of test patterns, and $x$ and $y$ are the measured and predicted biological activities of the compounds, respectively.

Since the test data set contains only a small number of compounds, the correlation coefficients of the test set are not significant enough to represent the relationship between the target values and the predicted values.

Therefore, only the overall correlation coefficient (i.e. the correlation coefficient for all the compounds: testing and training) is calculated for each training set.

The reported testing correlation coefficient is calculated by cross validating all the predictions for all the test data sets (i.e., considering the correlation coefficient between the biological activity and the predicted activity for every compound throughout the data set). Therefore, for each data set, only one testing correlation coefficient is reported.

## 3. Results

For all the experiments conducted in this paper the number of features was fixed. The number of features for the Selwood and BEN data sets was set according to the results presented in [7]. The number of features for the Breneman and DHFR data sets was set to 3. Exploring the optimal number of features for each data set is the subject of future work.

### 3.1 BPSO-BP

The approaches investigated in this paper have a general architecture that consists of two stages. In the first stage BPSO is used for feature selection and in the second stage a neural network is used to generate a QSAR model based on the features selected in the first stage. The first approach, which is discussed in this subsection, will be referred to as BPSO-BP. This approach uses back propagation to train the neural network in the second stage. The second approach, which is discussed in the next subsection, will be referred to as BPSO-PSO.

BPSO-BP was initially proposed in [7] and it is used in this paper as a base case for comparison. In [7], BPSO-BP was shown to successfully generate highly predictive QSAR models. In this paper, this result is established for an extended number of data sets. Furthermore, the difficulty in selecting an adequate learning rate for back propagation is demonstrated.

BPSO-BP consists of two nested loops. BPSO is the outer loop, and each iteration of this loop generates a set of selected features. The neural network with back propagation is the inner loop. The neural network takes the selected features as input, and is trained for a predefined number of iterations. The model fitness is fed back to the BPSO stage to guide the feature selection in the outer loop.

The population size in BPSO was set to 10 particles, and BPSO training was carried out for 50 iterations (the outer loop). The neural network was trained for 300 iterations (the inner loop) for each BPSO iteration.

Table 1 shows the leave-three-out result for the Selwood data set. The lowest average testing error corresponds to the case where compounds 25 through 27 were left out and the highest average testing error corresponds to the case where compounds 1 through 3 were left out. The overall correlation coefficient is 0.8899, which is comparable to the result obtained in [9] (0.90) by using linear regression, and to the one obtained in [7] (0.912). The features selected by BPSO-BP in this paper are similar to those selected by the methods presented in [7]. For example, features 3, 4, 49 correspond to the first ranking set of features selected in [7], and 31, 34, 49 correspond to the second ranking set of features. These two sets were both selected as shown in Table 1. Furthermore, feature 49 is the most frequently selected feature in both Table 1 and in [7]. All the features selected in Table 1 were also selected in [7] except for feature 44 which was not selected in [7]. The feature set that contained feature 44 produced the highest testing error in Table 1. This may be indicative of the fact that the method failed to train in this case. The testing correlation coefficient is 0.6902, which is lower than the overall correlation coefficient (0.8899).

**Table 1. BPSO-BP for the Selwood data set**

| Left out Compounds | Selected features | Overall avg error | $R_{overall}$ | Avg testing error |
|---|---|---|---|---|
| *1-3* | *3, 4, 44* | *0.1122* | *0.8139* | *0.3253* |
| 4-6 | 3, 6, 49 | 0.0948 | 0.9062 | 0.1152 |
| 7-9 | 32, 49, 50 | 0.0943 | 0.8971 | 0.1428 |
| 10-12 | 4, 34, 49 | 0.1071 | 0.8836 | 0.1384 |
| 13-15 | 3, 4, 49 | 0.0998 | 0.9021 | 0.2085 |
| 16-18 | 31, 34, 49 | 0.1048 | 0.8794 | 0.2254 |
| 19-21 | 31, 34, 49 | 0.0950 | 0.8904 | 0.1749 |
| 22-24 | 34, 49, 51 | 0.0934 | 0.9081 | 0.1093 |
| *25-27* | *35, 49, 51* | *0.0897* | *0.9112* | *0.0651* |
| 28-31 | 35, 49, 51 | 0.0897 | 0.9072 | 0.1200 |
| **Avg** | **NA** | **0.0981** | **0.8899** | **0.1625** |
| **$R_{testing}$ = 0.6902** | | | | |

It should be noted that low average testing error and overall average error, and high overall correlation coefficient and testing correlation coefficient are desired. The minimum value for the average testing error and overall average error is zero, whereas the maximum value for the overall correlation coefficient and testing correlation coefficient is one.

For the BEN data set, the lowest average testing error (0.0639) corresponds to the case where compounds 41 through 44 were left out, and the highest average testing error (0.1862) corresponds to the case where compounds 29 through 32 were left out. The overall correlation coefficient (0.9357) is comparable to the one obtained in [7] (0.951). Features 0 and 1 are the most frequently selected features in both [7] and in this paper.

The results obtained for the Selwood and BEN data sets together with the results reported in [7] show that BPSO can successfully identify the key features needed to construct predictive QSAR models. BPSO accomplishes this result by using 10 particles and 50 iterations, which adds up to 500 searches out of 23,426 and 52,426 possible solutions for the Selwood and BEN data sets, respectively. The quick convergence of BPSO allows it to process a large number of features efficiently.

For the Breneman data set, the lowest average testing error (0.0427) corresponds to the case where compounds 41 through 44 were left out, and the highest average testing error (0.5189) corresponds to the case where compounds 33 through 36 were left out. This latter case is an obvious case of failure to train, which affects the average testing error of the model.

Also, as opposed to the previous two data sets, the model generated for the Breneman data set seems to be less robust. A new set of features was selected every time a different set of compounds was left out. Furthermore, only few of the features were selected more than twice.

For the DHFR data set, the lowest average testing error (0.0253) corresponds to the case where compounds 61 through 80 were left out, and the highest average testing error (0.1328) corresponds to the case where compounds 241 through 256 were left out. For this data set, 12 out of the 13 times, BPSO selected the feature set 8, 9, and 10.

While the above results demonstrate that BPSO-BP can construct predictive QSAR models, the choice of the appropriate network parameters, as illustrated by the following experiments on the BEN data set, is critical for adequate neural network training.

In the first experiment, six runs of the QSAR modeling by using BPSO-BP were carried out. Compounds 13 through 16 of the BEN data set were left out. The learning rate value was set to 1. In the six runs, the number of training iterations was increased from 100 to 600. Figure 1 shows the overall average error (the lower line) and the average testing error (the upper line). As the number of training iterations increases, the training error decreases from 0.0940 for 100 iterations to 0.0622 for 600 iterations. However, when 500 training iterations are used, the overall average error jumps to 0.1006, which is the highest value of the overall average error among the six runs. The average testing error oscillates as the number of iteration changes. The large value of the learning rate is responsible for the oscillation. As will be shown next, this oscillation can be avoided by selecting a smaller learning rate. However, this will be at the cost of an increase in computational time.

Figure 2 shows the results of the second experiment in which another six runs of QSAR modeling by using BPSO-BP were carried out. In this case, the learning rate was set to 0.01. Figure 3 shows the overall average error (the lower line) and the average testing error (the upper line) of this experiment. As the number of training iterations increases, both the training error and testing error decreases. However, the convergence progresses very slowly. With 600 iterations, the overall average error is 0.1200 and the average testing error is 0.1460, these values are considerably higher than their values at convergence (i.e. 0.0668 and 0.0709, respectively).

Choosing the appropriate learning rate is often difficult and data set dependent.

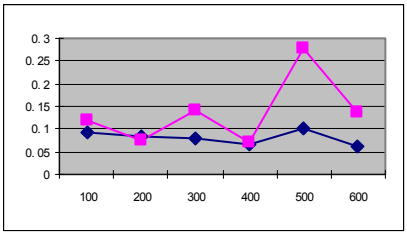


**Figure 1. Average testing and overall average errors with constant learning rate equal to 1**

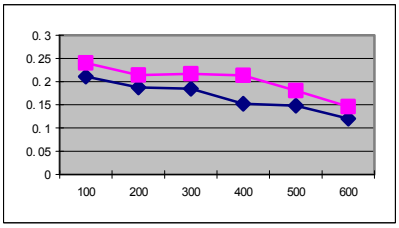


**Figure 2. Average testing and overall average errors with constant learning rate equal to 0.01**

### 3.2 BPSO-PSO

The prediction ability of the previous approach was shown to be sensitive to the learning rate, a back propagation training parameter. In this section, an approach that reduces the dependency of the prediction ability on the network parameters is proposed. This approach was earlier denoted by BPSO-PSO and has a similar first stage as the BPSO-BP. Both methods use BPSO for feature selection. However, they differ in the second stage. In BPSO-PSO, PSO instead of back propagation is used in the second stage to train the weight values of the neural network.

In the second stage of BPSO-PSO, each particle represents a set of weights, and the mean square error produced by the neural network is used as the fitness measure of PSO. The updated particles are fed to the neural network as new weights. It is expected that, after a certain number of iterations, the near-optimal weights can be obtained.

Unlike in back propagation where the learning rate governs the amount of adjustment to the weights in every iteration, in PSO the search for a solution is guided by the best solution that has been achieved so far. In addition, no data set dependent parameters need to be tuned to ensure convergence.

In this implementation, BPSO is carried out for 50 iterations with 10 particles, and PSO is carried out for 100 iterations with 20 particles for each BPSO iteration.

For the Selwood data set, the lowest average testing error (0.0604) corresponds to the case where compounds 7 through 9 were left out, and the highest average testing error (0.3246) corresponds to the case where compounds 25 through 27 were left out. The overall correlation coefficient (0.8768) is close to the result obtained in [9] (0.90) by using linear regression and to the one obtained in [7] (0.912) by using BPSO-BP. The features selected in this case are similar to those selected by BPSO-BP. Furthermore, all the features selected in this case were also selected in [7] except for feature 32. For both BPSO-BP and BPSO-PSO, feature 49 is the most frequently selected feature. The testing correlation coefficient is 0.7024, which is slightly higher than the testing correlation coefficient (0.6902) obtained by using BPSO-BP.

For the BEN data set, the lowest average testing error (0.0478) corresponds to the case where compounds 9 through 12 were left out, and the highest average testing error (0.3442) corresponds to the case where compounds 21 through 24 were left out. The overall correlation coefficient (0.9092) is comparable to that obtained in [7] (0.951) by using BPSO-BP. Also, the selected features are similar to those selected in [7]. The testing correlation coefficient (0.7612) is slightly lower than the one obtained by using BPSO-BP (0.7987).

For the Breneman data set, the lowest average testing error (0.0254) corresponds to the case where compounds 25 through 28 were left out, and the highest average testing error (0.3744) corresponds to the case where compounds 1 through 4 were left out. Similar to the BPSO-BP case, this approach also selects completely different features each time a new set of compounds are left out. Furthermore, there is little overlap between the features selected by BPSO-BP and those selected in this case by BPSO-PSO. Whether this indicates a very diverse solution space for the Breneman data set, or that some of the features are highly correlated is the subject of current investigation. The testing correlation coefficient (0.6942) is slightly higher than the one obtained by using BPSO-BP (0.6427).

For the DHFR data set, the lowest average testing error (0.0430) corresponds to the case where compounds 81 through 100 were left out, and the highest average testing error (0.1396) corresponds to the case where compounds 241 through 256 were left out. Interestingly, this approach selects exactly the same set of features every time. The selected features in this case also overlap with the ones selected by BPSO-BP. Features 1, 8 and 9 were selected every time for the DHFR data set with BPSO-PSO. These same features were selected most of the time except for one case where features 1, 8, and 9 were selected for the DHFR data set with BPSO-BP. The testing correlation coefficient (0.7829) obtained by using BPSO-PSO is slightly higher than the one obtained by using BPSO-BP (0.7384).

### 3.3 Discussion

Two methods for QSAR modeling are discussed in this paper. The first method, BPSO-BP, was already proposed in the literature [7] and BPSO-PSO is introduced in this paper. The results obtained with four different data sets in each case are presented. Table 2 summarizes the findings which indicate that the two approaches are comparable.

**Table 2. Average errors and correlation coefficients**

| | | Selwood | BEN | Breneman | DHFR |
|---|---|---|---|---|---|
| BPSO-BP | Overall avg error | 0.0981 | 0.0711 | 0.0953 | 0.0891 |
| | $R_{overall}$ | 0.8899 | 0.9357 | 0.8668 | 0.7613 |
| | Avg testing error | 0.1625 | 0.1249 | 0.1483 | 0.0925 |
| | $R_{testing}$ | 0.6902 | 0.7987 | 0.6427 | 0.7384 |
| BPSO- PSO | Overall avg error | 0.1032 | 0.0847 | 0.0963 | 0.0869 |
| | $R_{overall}$ | 0.8768 | 0.9092 | 0.8621 | 0.7892 |
| | Avg testing error | 0.1637 | 0.1472 | 0.1532 | 0.0934 |
| | $R_{testing}$ | 0.7024 | 0.7612 | 0.6942 | 0.7829 |

Compared to BPSO-BP, BPSO-PSO produced less stable results. While back propagation uses a predefined way of updating the weights, the weight updating process in PSO is influenced by random factors. Also PSO explores a larger candidate solution space than back propagation. Both of these aspects of PSO may lead to less stable results. To illustrate this, consider the case where compounds 1-3 are left out in the Selwood data set. Ten BPSO-BP models and ten BPSO-PSO models were built, and the standard deviations of the ten results were calculated as shown in Table 3. The last row of Table 3 is the standard deviation of the corresponding column. Compared to BPSO-BP, BPSO-PSO has higher standard deviation of the overall average error, the average testing error and the overall correlation, which indicates a less robust QSAR model generation process. One approach to addressing this instability is to run multiple instances of BPSO-PSO and retain the QSAR model with the highest

fitness score. The Bagging technique discussed in the next section builds on this idea.

**Table 3: Standard deviation of BPSO- BP and BPSO-PSO results**

| | BPSO-BP | | | BPSO-PSO | | |
|---|---|---|---|---|---|---|
| | Overall avg error | $R_{overall}$ | Avg testing error | Overall avg error | $R_{overall}$ | Avg testing error |
| 1 | 0.1323 | 0.7659 | 0.3555 | 0.1229 | 0.8154 | 0.2931 |
| 2 | 0.1315 | 0.7639 | 0.3533 | 0.1067 | 0.8830 | 0.1317 |
| 3 | 0.1272 | 0.7667 | 0.3717 | 0.1150 | 0.8268 | 0.2916 |
| 4 | 0.1324 | 0.7663 | 0.3553 | 0.0977 | 0.8873 | 0.1696 |
| 5 | 0.1373 | 0.7498 | 0.3762 | 0.1290 | 0.7012 | 0.3472 |
| 6 | 0.1326 | 0.7678 | 0.3702 | 0.1213 | 0.7773 | 0.3682 |
| 7 | 0.1352 | 0.7636 | 0.3763 | 0.1205 | 0.7970 | 0.3432 |
| 8 | 0.1327 | 0.7673 | 0.3546 | 0.1286 | 0.7573 | 0.3872 |
| 9 | 0.1324 | 0.7674 | 0.3546 | 0.0931 | 0.9007 | 0.1226 |
| 10 | 0.1320 | 0.7664 | 0.3568 | 0.1298 | 0.7657 | 0.3604 |
| | **0.0022** | **0.0065** | **0.0098** | **0.0131** | **0.0523** | **0.1019** |

# 4. Bagging method for QSAR

The bootstrap aggregation (Bagging) [8] method aggregates the results from different models. The success of this technique relies on the instability of the prediction method [8]. If slight perturbation of the training process results in significant changes in the outcome, then bagging can improve the robustness of the QSAR models.

In order to implement bagging, the data set is split into two sets: the training set and the test set. The training set is used to train the bagging model, and the test set is retained to test the quality of the bagging model. To provide enough data information for training, usually a large portion of the data is used as the training set and a small portion as the test set. Multiple neural network models are built based on the training data set. Each model is referred to as a bag. Each time, a different small portion of data is left out from the training set, forming a sub-training set. The sub-training set is constructed by randomly sampling with replacement a predefined percentage of compounds from the training set. Therefore, some compounds may appear in the sub-training set more than once. This characteristic creates more instability in the models generated across the bags, which is a desirable feature as explained in [8]. The sub-training set is the data set that is fed into the BPSO-neural network system during training. The validation set is used to establish the predictability of the neural network, (i.e., the quality of the model). The bagging model is constructed by averaging the output of all the models generated by using the above process.

## 4.2 Results and discussion

The bagging technique was implemented for BPSO-PSO. As suggested in [8], 20 bags were built for each QSAR model. Also, as suggested in [15], 60% of the entire data set was randomly picked as the training data set. Furthermore, out of the data set used for training, 10% was left out for testing as in the experiments discussed in Section 3.

**Table 4: Standard deviation with and without bagging**

| | | Standard Deviation of the Overall avg error | Standard Deviation of $R_{overall}$ | Standard Deviation of the Avg testing error |
|---|---|---|---|---|
| Bagging | Selwood | 0.0066 | 0.0095 | 0.0162 |
| | BEN | 0.0048 | 0.0117 | 0.0250 |
| | Breneman | 0.0507 | 0.0279 | 0.0761 |
| | DHFR | 0.0087 | 0.0136 | 0.0092 |
| Without Bagging | Selwood | 0.0257 | 0.0840 | 0.0842 |
| | BEN | 0.0379 | 0.0788 | 0.1136 |
| | Breneman | 0.1302 | 0.2689 | 0.1980 |
| | DHFR | 0.0202 | 0.0494 | 0.0745 |

To illustrate the importance of bagging in generating robust QSAR models, ten bagging models and ten models without bagging were built for each data set using BPSO-PSO, and the overall average error, overall correlation coefficient and average testing error were obtained for each model. The standard deviation for all three parameters and for each model was calculated and is listed in Table 4. For each data set, models with bagging have smaller standard deviation for overall average error, overall correlation coefficient and average testing error. A smaller standard deviation indicates a more robust QSAR model generation process.

It can be argued that bagging increases computational time and places the BPSO-PSO with bagging at a disadvantage compared to BPSO-BP with a small learning rate which was itself computationally intensive. However, the computation needed to generate the bags can be done in parallel whereas the computation underlying back propagation is inherently sequential and cannot be parallelized.

# 6. Conclusions and Future Work

Two approaches for constructing QSAR models and selecting relevant features have been discussed in this paper. The approaches are based upon computational intelligence tools such as PSO and neural networks. Four

data sets have been used to test each of the two approaches.

Both BPSO-BP and BPSO-PSO produce QSAR models with comparable predictive capabilities. However, the limitation of BPSO-BP lies in the difficulty of determining some of the back propagation neural network parameters. For instance, a large value of the learning rate for back propagation enables the network to converge rapidly. However, oscillation may occur around the most predictive QSAR model. A small value of the learning rate avoids this oscillation. However, this is at the cost of a longer computational time. BPSO-PSO addresses this problem by using PSO to update the weights in the neural network. However, due to the random nature of PSO, BPSO-PSO produces less robust QSAR models compared to BPSO-BP. For this purpose, bagging is used to minimize the instability of the QSAR models. The BPSO-PSO approach with bagging produces robust models that are as predictive as the BPSO-BP approach.

The number of features selected for the QSAR models was fixed in each of the experiments conducted in this paper. Determining the appropriate number of features is the subject of future work.

## Acknowledgment

This work was supported in part by Eli Lilly and Company. We also would like to thank professors C. M. Breneman and N. Sukumar (Rensselaer Polytechnic Institute) for supplying their data set.

## References

[1] D. Rogers, and A. J. Hopfinger. Application of Genetic Function Approximation to Quantitative Structure-Activity Relationships and Quantitative Structure-Property Relationships. *J. Chem. Inf. Comput. Sci.,* 1994, 34, 854-866.

[2] C. Hansch, A. Leo, R. Stephen, and Eds. Heller. Exploring QSAR, Fundamentals and Applications in Chemistry and Biology. *ACS professional Reference Book.,* American Chemical Society, Washington, D.C., 1995.

[3] W.D. Glen, W. J. Dunn, and R. D. Scott. Principal Components Analysis and Partial Least Squares Regression. *Tetrahedron Comput. Methodol.,*1989, 2, 349-376.

[4] J. Wikel, and E. Dow. The Use of Neural-Networks for Variable Selection in QSAR. *Bioorg. Med. Chem. So*c., 1988, 110, 5959-5967.

[5] R. Leardi, R. Bogga, and M. Terrille. Genetic Algorithms as a Strategy for Feature selection. *J. Chemometrics,* 1992, 6, 267-281.

[6] R. C. Eberhart, and Y Shi. *Computational Intelligence: Concepts to Implementations,* Morgan Kaufmann Publishers. San Francisco, (in press).

[7] D. K. Agrafiotis, and W. Cedeno. Feature Selection for Structure-Activity Correlation Using Binary Paricle Swarms. *J. Med. Chem*., 2002, 45, 1098.

[8] L. Breiman, Bagging predictors, Technical Report No. 421, Department of Statistics, University of California, Berkeley, 1994.

[9] D. L. Selwood, D.J. Livingtone, J.C. Comley, A.B. O'Dowd, A.T. Hudson, P. Jackson, K.S. Jandu; V.S. Rose, and J.N Stables. Structure-activity relationships of antifilarial antimycin analogs: a multivariate pattern recognition study. *J.* Med. Chem**.,** 1990, 33, 136.

[10] J. McFarland, and D. Gans. On Identifying Likely Determinants of Biological Activity in High Dimensional QSAR Problem, *Quant. Structure-Act. Relat.,*1994, 13, 11.

[11] H. Kubinyi; Variable Selection in QSAR Studies. II. A Highly Efficient Combination of Systematic Search and Evolution, *Quant. Structure-Act. Relat*., 1994, 13, 393.

[12] C. Waller, and M. Bradley. Development and Validation of a Novel Variable Selection Technique with Application to Multidimensional Quantitative Structure-Activity Relationship Studies, *J Chem. Inf. Comput. Sci.,*1999, 39, 345.

[13] D. J. Maddalena, and Graham A. R. Johnston. Prediction of Receptor Properties and Binding Affinity of Ligands to Benzodiazepine/GABAA Receptors Using Artificial Neural Networks. *J. Med. Chem.,* 1995, 38, 715.

[14] C. M. Breneman, N. Sukumar, K. P. Bennett, M. J. Embrechts, M. Sundling, and L. Lockwood. Wavelet Representations of Molecular Electronic Properties: Applications in ADME, QSPR, and QSAR, *Proceedings of the American Chemistry Society National Meeting,* Washington D.C, 2000.

[15] M. J. Embrechts, F. Arciniegas, M. Ozdemir, C. M. Breneman, K. Bennett, and L. Lockwood. Bagging Neural Network Sensitivity Analysis for Feature Reduction for In-Silico Drug Design. *2001 INNS - IEEE International Joint Conference on Neural Networks*, IEEE Press, Washington D.C., 2001, 4, 2478.

[16] T. A. Andrea, and H. Kalayeh, Applications of Neural Networks in Quantitative Structure-Activity Relationships of Dihydrofolate Reductase Inhibitors. *J. Med. Chem.,* 1991, 34, 2824.