A Comparison of Particle Swarms Techniques for the Development of Quantitative Structure-Activity Relationship Models for Drug Design

Walter Cedeño & Dimitris Agrafiotis
Johnson & Johnson Pharmaceutical R&D
665 Stockton Drive
Exton, PA 19341
wcedeno@prdus.jnj.com
dagrafio@prdus.jnj.com

Abstract

The development of quantitative structure-activity relationship (QSAR) models for computer-assisted drug design is a well-known technique in the pharmaceutical industry. QSAR models provide medicinal chemists with mechanisms for predicting the biological activity of compounds using their chemical structure or properties. This information can significantly reduce the time to discover a new drug. This work compares and contrasts particle swarms to simulated annealing and artificial ant systems techniques for the development of QSAR models based on artificial neural networks and k-nearest neighbor and kernel regression. Particle Swarm techniques are shown to compared favorably to the other techniques using three classical data sets from the QSAR literature.

1. Introduction

The design of new drugs with the suitable physicochemical properties is a challenge faced by the pharmaceutical industry on a daily basis. Techniques that can reduce the time to design new drugs and improve the quality of drug candidates can significantly decrease the cost to bring new drugs to the market.

The increasing amount of information available in digital form has prompted scientist to develop novel data mining methodologies to help process and interpret large volumes of data faster and with greater reliability. Artificial intelligence methods, such as artificial neural networks (ANN) [1], classification and regression trees (CART) [2], and k-nearest neighbor classifiers (KNN) [3], have been used extensively for this purpose [4][5].

These methods are used in drug design to correlate some measure of biological activity with a set of physicochemical, structural and/or electronic properties, known as descriptors, of the compounds under investigation. It is assumed that the biological activity of a compound is related to its chemical structure, and can therefore be inferred from a carefully chosen set of molecular descriptors. The key challenge is to determine which set of descriptors correlates best with biological activity. Since it is not possible to know in advance which molecular features are most relevant to the problem at hand, a comprehensive set of descriptors is usually employed, chosen based on experience, software availability, and computational cost.

Quantitative structure-activity relationship (QSAR) models for computer-assisted drug design are a well-known technique in the pharmaceutical industry to correlate biological activity with compounds properties, known as features. QSAR models based in ANN, CART, or KNN provide medicinal chemists with mechanisms for predicting the biological activity, such as drug potency and toxicity, of compounds using their chemical structure or properties. This information can significantly reduce the time to discover a new drug. However, is well known, both in the chemical and statistical fields, that the number of features used in a QSAR model can greatly affect its accuracy. The presence of noise and irrelevant or redundant features can cause the method to learn the idiosyncrasies of the individual samples and lose sight of the broad picture that is essential for generalization beyond the training set [6]. This problem is compounded when the number of features is also relatively small, as is often the case in molecular design. If the number of features is comparable to the number of training patterns, the parameters of the model may become unstable and unlikely to replicate if the study were to be repeated.

The large number of descriptors available, which can be used as features for the QSAR model, also increases the risk of chance correlations [7]. Comparing the results against randomly generated results is commonly used to verify that the risk of chance correlations is low or does not exist.

Two techniques, feature selection and feature weighting, are often used to remedy this situation and improve the accuracy of a classification or regression technique. In feature selection, the goal is to select a subset of the features that can best predict the biological activity of compounds. Feature selection is often used to create QSAR models based on ANN. In feature weighting the goal is not only to select a subset of features, but also to define their relative influence for predicting the biological activity of compounds. Feature weighting is often used to create QSAR models based on KNN. In this work, we compare and contrast the use particle swarms to artificial ants and simulated annealing techniques for the development of QSAR models based on ANN and KNN.

The following section provides an overview of feature selection and feature weighting. Section 3 presents an overview of the KNN regression technique. Section 4 presents an overview of ANN for regression. Section 5 presents the background for particle swarms. Section 6 describes our implementation of binary particle swarms for feature selection. Section 7 describes niching particle swarms, a technique to encourage the formation of niches during the search. Section 8 provides the background for simulated annealing. Section 9 provides the background for artificial ant systems. Section 10 provides the results from comparing the various techniques. Finally, Section 11 provides our conclusions for this work.

2. Feature Selection and Feature Weighting

Feature selection is often used in QSAR to find the best set of compound properties that can improve the accuracy of the regression technique. Feature selection works by identifying a small subset of necessary and sufficient features that can be used as input to the underlying predictor. Feature selection algorithms can be divided into three main categories [8]: 1) those where the selection is embedded within the basic regression algorithm, 2) those that use feature selection as a filter prior to regression, and 3) those that use feature selection as a wrapper around the regression. The latter has a long history in statistics and pattern recognition, and is the method of choice for QSAR.

Feature selection can be viewed as a heuristic search, where each state in the search space represents a particular subset of the available features. In all but the simplest cases, an exhaustive search of the state space is impractical, since it involves $n!/(n-r)!r!$ possible combinations, where n is the total number of available features and r is the number of features selected. Several search algorithms have been applied to this problem, ranging from simple greedy approaches such as forward selection or backward elimination [9], to more elaborate methodologies such as simulated annealing [10], evolutionary programming [11], genetic algorithms [12][13][14][15], artificial ants [16], and more recently particle swarms [17][18].

Feature selection is a special case of a general technique known as feature weighting [19][20]. In feature weighting a weight value is associated with each feature. The weight value is usually a number in the interval [0, 1] and denotes the contribution of the feature in the learning algorithm. In feature selection weight values are 1 or 0 indicating whether the feature is used or not, no partial feature information is allowed. The aim of feature weighting is to find the relative importance of each feature. In some cases, using partial information from each feature has been shown to benefit the learning algorithm [21][22][23].

3. *K*-Nearest Neighbor and Kernel Regression

The k-nearest neighbor method (KNN) is an intuitive method used extensively for classification. Given a pattern to classify, KNN works by selecting the k most similar patterns from a set of well-known classified data (training data) and choosing the class with the most representatives in the set. Similarity between elements is typically measured using the Euclidean distance in some appropriate feature space or some other suitable metric. KNN is a lazy algorithm, i.e., it defers data processing until needed. The algorithm uses local information and adapts well to changes in the training data. Two main drawbacks of KNN are its susceptibility to noise and the curse of dimensionality. These can be alleviated using normalization and feature weighting to calculate the distance between patterns according to the equation

$$d(p,q) = \sqrt{\sum_{i=1}^{n} w_i (p_i - q_i)^2} \qquad (1)$$

where n is the number of features, p_i and q_i are the i-th feature values for patterns p and q respectively, and w_i the weight for the i-th feature. The implementation of KNN used in this work is based on k-d trees in order

to reduce the computational cost associated with calculating distances [24].

Kernel regression is a closely related non-parametric methodology that uses local information to obtain a prediction. The main difference from KNN is that kernel regression is used for predicting an unknown value where as KNN is used for selecting the *k* patterns to use for prediction. In this work, we want to predict the biological activity for a compound *q* based on the weighted average of the biological activity of known compounds in the neighborhood of *q*. A kernel function is used to give more weight to compounds that are closer more similar to *q* in descriptor space.

The patterns to use for prediction are the *k* most similar chemical compounds with known biological activity. Similarity between compounds is measured using a suitable set of molecular features. We use KNN to select the *k* most similar compounds to a compound *q* and apply kernel regression using the kernel function

$$f(p,q) = \frac{1}{1 + d(p,q)} \tag{2}$$

where *d(p,q)* is the KNN distance as given in equation 1. The prediction for *q* is obtained by

$$y'(q) = \frac{\sum_{i=i}^{k} y_i f(p_i, q)}{\sum_{i=i}^{k} f(p_i, q)} \tag{3}$$

where *k* is the number of nearest neighbors selected for *q*, p_i is the *i*-th nearest neighbor of *q*, y_i is the known response value of p_i, and *f(p, q)* the kernel function given in equation 2.

4. Artificial Neural Networks

Artificial Neural Networks attempts to mimic biological neural systems by modeling the low-level structure of the brain. ANN have been used extensively as a regression technique due to their ability to model non-linear systems. Our analysis was based on three-layer, fully connected multilayer perceptrons, trained with the standard error back-propagation algorithm. The logistic transfer function

$$f(x) = 1/(1 + e^{-x}) \tag{4}$$

was used for both hidden and output layers. During feature selection, each network was trained for 200 epochs, using a linearly decreasing learning rate from 1.0 to 0.01 and a momentum of 0.8. During each epoch, the training patterns were presented to the network in a randomized order. To minimize the risk of back-propagation getting trapped in local minima in synaptic weight space, each model was trained 3 times, and the model with the lowest training error was retained.

5. Particle Swarms

Particle swarms (PS) is a relatively new optimization paradigm introduced by Kennedy and Eberhart [25]. The method is based on the observation that social interaction, which is believed to play a crucial role in human cognition, can serve as a valuable heuristic in identifying optimal solutions to difficult optimization problems. Particle swarms explore the search space using a population of individuals, each with an individual, initially random, location and velocity vector. The particles then "fly" over the state space, remembering the best solution encountered. Fitness is determined by an application-specific objective function *f(x)*. During each iteration, the velocity of each particle is adjusted based on its momentum and the influence of the best solutions encountered by itself and its neighbors. The particle then moves to a new position, and the process is repeated for a prescribed number of iterations. In the original PS implementation [26], the trajectory of each particle is governed by the equations:

$$v_i(t+1) = v_i(t) + \eta_1 r(p_i - x_i(t)) + \eta_2 r(p_{b(i)} - x_i(t)) \tag{5}$$

and

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{6}$$

where x_i and v_i are the current position and velocity of the *i*-th particle, p_i is the position of the best state visited by the *i*-th particle, *b(i)* is the particle with the best fitness in the neighborhood of *i*, and *t* is the iteration number. The parameters η_1 and η_2 are called the *cognitive* and *social learning rates,* and determine the relative influence of the memory of the individual versus that of its neighborhood. In the psychological metaphor, the cognitive term represents the tendency of organisms to repeat past behaviors that have proven successful or have been reinforced by their environment, whereas the social term represents the tendency to emulate the successes of others, which is fundamental to human sociality. In effect, these terms

introduce a tendency to sample regions of space that have demonstrated promise. *r* is a random number whose upper limit is a constant parameter of the system, and is used to introduce a stochastic element in the search process.

Kennedy defined four models of PS. The full model, which places equal influence to the cognitive and social influence, the social-only model, which involves no cognitive learning, the cognitive-only model, which has no social component, and the selfless model, which is a social-only model in which the individual is excluded from consideration in determining its neighborhood's best. The neighborhood represents a subset of the population surrounding a particular particle. The neighborhood size defines the extent of social interaction, and can range from the entire population to a small number of neighbors on either side of the particle (i.e. for the *i*-th particle, a neighborhood size of 3 would represent particles *i-1*, *i*, and *i+1*).

The present work employs Shi and Eberhart's [27][28] variant of the PS algorithm, which makes use of an inertia weight, *w*, to dampen the velocities during the course of the simulation, and allow the swarm to converge with greater precision:

$$v_i(t+1) = \omega v_i(t) + \eta_1 r(p_i - x_i(t)) + \eta_2 r(p_{b(i)} - x_i(t)) \tag{7}$$

Larger values of ω induce larger transitions and thus enable global exploration, whereas lower values facilitate local exploration and fine-tuning of the current search area.

This work examines the use of particle swarms as a wrapper around ANN and KNN with kernel regression. When used with ANN, the location vector for each particle corresponds the subset of features used for the regression model. A new technique, called binary particle swarms, is introduced to convert location values to 0 or 1. For KNN, the location vector for each particle corresponds to the feature weights. The swarm searches for the best subset of features and corresponding weights that minimize the regression error in the training data. All weights are limited to values in the range [0, 1]. Location vectors that move outside this range are normalized at run time. In the present study, only a fixed number of features with the highest weight values were used to construct the actual models.

6. Binary Particle Swarms

The particle swarm algorithm, which was originally intended for searching multidimensional continuous spaces, can be adapted to the discrete problem of feature selection by viewing the location vectors of the particles as probabilities and employing roulette wheel selection to construct candidate subsets. In this scheme, the elements of the location vectors x_{ij} and p_{ij} can only take the values 0 and 1 to indicate whether the *jth* feature is selected in the *ith* particle (subset). A discretization step is introduced following the application of equation 6, which converts the fractional coordinates, x_{ij}, to binary values using probabilistic selection. During this step, the fractional values of x_{ij} are treated as probability thresholds to determine subset membership. Two possibilities can be envisioned to prevent overfitting: (1) to select each feature on the basis of its own probability and employ an objective function that penalizes solutions containing a large number of features, such as Rao's lack-of-fit [29]; (2) to select a predefined number of features on the basis of the ratio of the number of training patterns to the number of freely adjustable parameters in the model. In the latter case, which is the one employed in this work, the features comprising the model are determined by roulette wheel selection. In this method, each feature is assigned a slice of a roulette wheel whose size is equal to the probability assigned to that feature. The subset is assembled by spinning the wheel and selecting the features under the wheel's marker. This process is repeated *k* times, where *k* is the number of desired features in the model (duplicates are excluded).

The actual probabilities, P_{ij}, are computed by equation 8:

$$P_{ij} = x_{ij}^{\alpha} \Big/ \sum_{j=1}^{n} x_{ij}^{\alpha} \tag{8}$$

where x_{ij} is the fractional coordinates obtained by applying equation 6 (confined in the interval [0, 1]) and α is a scaling factor referred to as selection pressure. If α is greater than 1, the selection tends to emphasize highly fit individuals, whereas if it is less than 1, the differences between the individuals are attenuated and less fit individuals have an increased chance of being selected. To eliminate redundant computation, the program caches all visited states and their fitness into a lookup table, implemented as a sorted vector of pointers with $O(n \log n)$ insertion and retrieval time.

7. Niching Particle Swarms

Encouraging niches in Partcle Swarms have been used successfully in applications of drug design and

multimodal function optimization. NichePSO [30] locates niches through the growing of subswarms from an initial population of particles using a cognitive model PS. Niching Particle Swarms [18] (NPS) introduces a new neighborhood operator to encourage inetraction among nearby particles. NPS has been successfully applied to drug design in the past and it's the technique of choice here.

In NPS, niching (speciation) is introduced in PS by encouraging social interaction among similar particles. During each iteration, the neighborhood best for each particle is selected from a group of most similar particles taken at random from the population. First, the algorithm selects k groups of m particles chosen at random with replacement from the population. Then the most similar member to the current particle from each group is selected. Finally, the best particle among the most similar ones is selected as the neighborhood best. Similarity is typically defined by the Euclidean distance in feature space. The parameters m and k are called the neighborhood size and neighborhood sample size respectively, and are used to balance exploration versus exploitation. Large values of m encourage the selection of close neighbors, while large values of k encourage the selection of better particles.

8. Simulated Annealing

Simulated annealing (SA) is a global, multivariate optimization technique based on the Metropolis Monte-Carlo search algorithm [31]. The method starts from an initial random state, and walks through the state space associated with the problem of interest by generating a series of small, stochastic steps. As with particle swarms, an objective function maps each state into a value that measures its energy or fitness. While downhill transitions are always accepted, uphill transitions are accepted with a probability that is inversely proportional to the energy difference between the two states. This probability is computed using Metropolis' acceptance criterion

$$p = e^{-\Delta E/(KT)}, \tag{9}$$

where K is a constant used for scaling purposes and T is an artificial temperature factor that controls the ability of the system to overcome energy barriers. The temperature is systematically adjusted during the simulation in a manner that gradually reduces the probability of high energy transitions. In this case a Gaussian cooling schedule with a half-width of 5 deviation units was used.

To circumvent the problem of assigning the appropriate value for K and to ensure that the transition probability is properly controlled, an adaptive approach is used. In this approach, K is not a true constant, but rather it is continuously adjusted during the course of the simulation on the basis of a running estimate of the mean transition energy [32][33]. In particular, at the end of each transition, the mean transition energy is updated, and the value of K is adjusted so that the acceptance probability for a mean uphill transition at the final temperature is 0.1%. In general, schedules that involve more extensive sampling at lower temperatures seem to perform best, although it is also important that sufficient time must be spent at higher temperatures so that the algorithm does not get trapped into local minima.

In the problem at hand, a state may represent the set of feature weights used in KNN or the set of selected features used in ANN. In both cases, the objective is to minimize the regression error in the training data. For KNN, two stochastic steps were evaluated: 1) after each step a feature is randomly selected and a new random value, in the interval [0, 1], is assigned to the weight, and 2) after each step all feature weights are adjusted by a random value in the interval [-0.25, 0.25]. All weight values are kept in the interval [0, 1]. For ANN, a simple stochastic step replaces a selected feature with one chosen at random from the set of available features.

9. Artificial Ant Systems

The algorithms based on artificial ant systems (AAS) are inspired by the social behavior of ants in a colony. Ants use deposits of pheromone as a communication agent and are able to find the shortest path between a food source and their nest [34]. As ants travel in search of food they deposit pheromone on the ground to mark their path. Ants move at random in search of food, when they detect pheromone trails they follow one of them with a probability proportional to the amount of pheromone on the trail. As multiple ants follow the same trail, depositing their own pheromone, they reinforce the trail making it more attractive to other ants. Initially, all paths to a food source are equally probable. In time, the shorter paths encounter more ants making round trips to the food source and receive more pheromone. Thus, short paths become increasingly more attractive to the ants. Eventually, all ants follow the shortest trail.

The application of AAS to drug design is based on the development QSAR models based on ANN. For a detailed description of this application, see the work of Izrailev and Agrafiotis [16].

10. Results and Discussion

The methods were tested on three well-known data sets: antifilarial activity of antimycin analogues (AMA) [9], binding affinities of ligands to benzodiazepine/$GABA_A$ receptors (BZ) [35], and inhibition of dihydrofolate reductase by pyrimidines (PYR) [36]. These data sets have been the subject of extensive QSAR studies, and have served as a test bed for many feature selection algorithms.

Table 1: **Data set size and ANN topology used.**

Data Set	*N*	*M*	*F*	*H*
AMA	31	53	3	3
BZ	57	42	6	2
PYR	74	27	6	2

Table 1 summarizes the number of samples (*N*), number of features (*M*) in the original data set, number of features used in the models (*F*), and number of hidden neurons (*H*) for the data sets. In all three cases, the descriptor data were normalized to [0, 1] prior to modeling with ANN and KNN with kernel regression.

There are multiple observations that need to be point out from these data sets. First, using 10000 objective function evaluations for each run allows us to compare the techniques under three scenarios. Given the number of features selected for AMA, BZ, and PYR data sets there are a total of 23426, 5245786, and 296010 possible feature subsets respectively. Given the feature space size and the number of objective function evaluations used, any of the techniques could examine up to 42.7%, 0.2%, and 3.4% of the possible subsets for the AMA, BZ, and PYR data sets respectively. Obviously, we have a scenario were the number of possible evaluations is relatively high compared to the number of possible subsets as is the case for the AMA data set. In the other scenarios we have a low percentage of the subsets being examined, the case with BZ data set, and somewhere in between, the case with PYR data set. Techniques applied in conjunction with KNN have the additional search space complexity of finding the weights for the within the feature space in order to obtain better results.

All programs were implemented in the C++ programming language. All calculations were carried out on a Dell Inspiron 8100 laptop computer equipped with a 1.133 GHz Pentium IV Intel processor running Windows 2000 Professional.

Following common practice, the cross-validated correlation coefficient, R_{CV}, resulting from leave-one-out (LOO) cross-validation was used to define the quality of the resulting models. This value is based on the training correlation coefficient *R* given by equation 10.

$$R = \frac{N\sum_{i=1}^{N} y_i \tilde{y}_i - \sum_{i=1}^{N} y_i \sum_{i=1}^{N} \tilde{y}_i}{\sqrt{\left[N\sum_{i=1}^{N} y_i^2 - \left(\sum_{i=1}^{N} y_i\right)^2\right]\left[N\sum_{i=1}^{N} \tilde{y}_i^2 - \left(\sum_{i=1}^{N} \tilde{y}_i\right)^2\right]}} \quad (10)$$

where *N* is the number of training patterns, and y_i and $\tilde{y}_i$ are the measured and predicted activities of the i-th compound, respectively.

The LOO cross-validation coefficient is obtained by systematically removing one of the patterns from the training set, building a model with the remaining cases, and predicting the activity of the removed case using the optimized weights. This is done for each pattern in the training set, and the resulting predictions are compared to the measured activities to determine their degree of correlation.

Table 2: **Top models selected by each method using ANN.**

Method	Data	Variables	$\mu(R_{CV})$	$\sigma(R_{CV})$
ANN-NPS	AMA	37,49,51	0.831	0.013
ANN-BPS	AMA	31,35,49	0.831	0.009
ANN-SA	AMA	31,37,49	0.837	0.008
ANN-AAS	AMA	31,37,49	0.838	0.010
ANN-NPS	BZ	1,4,6,9,15,23	0.906	0.012
ANN-BPS	BZ	1,4,6,9,20,21	0.900	0.019
ANN-SA	BZ	1,4,5,9,20,23	0.901	0.005
ANN-AAS	BZ	0,1,5,9,11,14	0.890	0.009
ANN-NPS	PYR	0,5,8,10,19,22	0.822	0.035
ANN-BPS	PYR	0,5,10,16,19,22	0.808	0.014
ANN-SA	PYR	1,2,3,5,19,22	0.795	0.015
ANN-AAS	PYR	1,2,3,5,19,22	0.796	0.005

The best models identified using ANN were retrained and cross-validated 50 times using LOO cross-validation, in order to establish their true learning and generalization capabilities. The best models identified using KNN were obtained after 50 runs. Since KNN does not involve any training and the patterns in the test data do not participate in the prediction of their own response values, the training *R* corresponds to the LOO cross-validated R_{CV}.

Table 2 provides a summary of the LOO cross-validation results obtained for QSAR models based on ANN. Comparing the results in Table 2 (summarized

in Figure 1) shows that NPS-ANN outperformed the other methods for the BZ and PYR data sets by a slight margin, though the results are probably statistically insignificant. Equally inconclusive are the results for the AMA data set, where NPS-ANN performed similar to PS-ANN and slightly worse than SA-ANN and ANT-ANN. However, NPS did demonstrate the ability to form niches during the search. This allows the method to escape local minimal and explore interesting areas of the search space in parallel. Although these results do not allow any definitive conclusions to be drawn, they demonstrate the potential of NPS-ANN for building QSAR models with good generalization power. We believe that niching will prove more useful with more complicated fitness landscapes that exhibit a greater number of local minima.

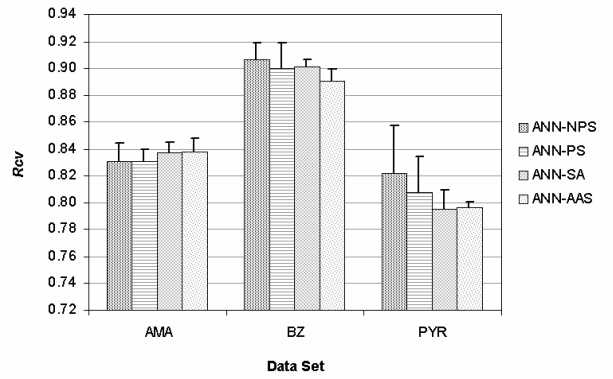

Figure 1: **LOO cross-validation values for the best models discovered by ANN based methods.**

These results provide a glimpse of the potential of NPS-ANN for QSAR modeling. The success demonstrated is, in part, due to the ability of NPS to escape local minima and maintain multiple solutions during the search. Further exploration of the effect of the niching parameters could allow us to further improve the technique. This should improve the ability to select the best set of parameters for any given data set and hopefully lead to better statistical models.

KNN-PS was compared to three other search algorithms. Two of the algorithms are based on simulated annealing, as described in a previous section, and only differing on the logic used to modify the current state. The third algorithm used is random search. All algorithms were limited to 10000 objective function evaluations. Particle swarms obtained best results with a population size of 100 and the number of steps set at 100. Simulated annealing algorithms used a schedule with 30 different temperatures. Table 3 (summarized in Figure 2) lists the best models found by each algorithm.

Table 3: **Top models selected by each method using KNN.**

Method	Data Set	Variables	K	$\mu(R_{CV})$
KNN-PS	AMA	15, 31, 5	3	0.867
KNN-SA1	AMA	49, 18, 19	1	0.858
KNN-SA2	AMA	49, 19, 18	1	0.861
KNN-RA	AMA	18, 49, 19	1	0.858
KNN-PS	BZ	9, 3, 17, 1, 13, 2	2	0.885
KNN-SA1	BZ	10, 27, 2, 9, 3, 14	2	0.873
KNN-SA2	BZ	14, 3, 2, 9, 26, 1	3	0.863
KNN-RA	BZ	22, 9, 3, 2, 14, 1	4	0.846
KNN-PS	PYR	10, 4, 19, 0, 9, 11	3	0.842
KNN-SA1	PYR	4, 10, 9, 19, 11, 0	3	0.839
KNN-SA2	PYR	10, 4, 19, 9, 0, 11	3	0.838
KNN-RA	PYR	17, 10, 19, 0, 4, 11	3	0.824

In the AMA data set, only PS was able to separate itself from the other techniques. It not only found a better model, but it did it with a different set of features and using 3 neighbors. Although not shown, the AMA results also show the importance of the weight values. Although the difference between the SA1, SA2, and RA best models is small, the slight difference between the weights produced a better model for the SA2 method. Difference between the weight values is more noticeable for the PS model, where the values are 1.000, 0.526, and 0.372 for features 15, 31, and 5 respectively. Similar observations were made for the PS method for the other data sets. This suggests that PS is naturally adept at working in continuous search spaces, which is not a surprise since the technique was created for that purpose.

For the BZ data set, the PS method obtained the best model as well. A bigger difference exist between the correlation coefficient values for all four methods. Three features, 2, 3, and 9, are present in the best models for all methods. Features 1 and 14 are present in three of the four best models. Once again, the PS method seems to adjust the weights successfully to get better solutions. This data set seems to be the most challenging for all methods. The differences on the neighborhood size used for the best models and the differences in the features selected suggests that. Better solutions might be obtained by increasing the number of objective function evaluations allowed or by trying other kernel functions.

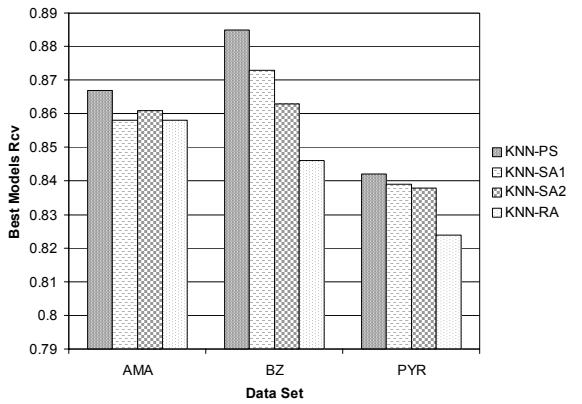

Figure 2: **LOO cross-validation values for the best models discovered by KNN based methods.**

Similar results were observed for the PYR data set. The best model was obtained by the PS method. Most methods found that features 0, 4, 9, 10, 11, and 19 are the best features to use. All methods also found the same neighborhood size. The advantage of PS over the other models appears to be in its ability to successfully find better weight values. A summary of the results can be found in Figure 2 for all methods on all three data sets.

Comparing the results obtained with ANN models to those using KNN with PS optimization is not straightforward. The PS approach appplied to KNN does feature weighting, which is a more complex problem than the feature selection technique employed with ANNs. The method not only selects the best features for the kernel regression model, but also identifies their relative contribution. The search space associated with this problem is more complex than the binary problem of classical feature selection. One can argue that the ANN is in fact performing an implicit feature weighting while training, therefore simplifying the problem to one of subset selection. Another difference has to do with the regression techniques used. ANN is a supervised learning technique and requires training prior to applying the technique, whereas KNN does not required training and relies on the appropriate selection of *k*, the neighborhood size, the kernel function, and appropriate weight values.

Comparing the results in Tables 2 and 3 shows that KNN-PS outperformed the other methods for the AMA and PYR data sets. KNN-PS did not do too well for the BZ data set. This might be due, in part, to our choice of kernel function. A different kernel function might do a better job with the BZ data set.

11. Conclusions

Although these results do not allow any definitive conclusions to be drawn, they demonstrate the potential of particle swarms for building QSAR models with good generalization power. Moreover, it allows us to compare QSAR models based on KNN with those based on ANN. KNN based models provides the researcher with a weight vector that can help in determining the relative importance of each feature. One can argue that the weight of the selected features leads to an easier interpretation of the model, which, in some cases, may be a significant advantage over ANN. On the other hand, KNN-based methods are computationally more intensive, since they require a search for the *k*-nearest neighbors before calculating the predicted value. On the other hand, since there is no training involved, KNN regression provides an easier way to include new data in the model.

For ANN based models, niching particle swarms seem to provide an advantage over binary particle swarms. The algorithm enhances binary particle swarms by encouraging niching among particles in the population. The niching particle swarms algorithm was able to identify QSAR models with slightly better generalization power as measured by LOO cross-validation in two of the three data sets tested. The results suggest that localized exploration and exploitation may lead to better results in some cases. The ability to adjust the parameters that control how much emphasis is put on niching proved to be helpful. Further empirical work is necessary in order to fully assess the effects of the niching parameters on the ability of the algorithm to locate the global minimum.

Acknowledgements

The authors would like to thank the member of the Molecular Design and Informatics group at Johnson & Johnson Pharmaceutical R&D for many useful discussions and support of this work.

References

[1] Devillers, J., Ed., Neural networks in QSAR and drug design, Academic Press, 1996.

[2] Breiman,L., Friedman, J. H., Olshen, R. A. and Stone, C. J., Classification and Regression Trees, Wadsworth Int. Group, Belmont, California, USA, 1984.

[3] Parzen, E, On the Estimation of a Probability Density Function and Mode, Ann. Math. Stat., Vol. 33, 1962, pp. 1065-1076.

[4] van de Waterbeemd, H., Ed., Chemometric methods in molecular design, in Methods and Principles in Medicinal Chemistry, Vol. 2, VCH, Weinheim, 1995.

[5] Hansch, L., and Leo , C., Exploring QSAR. Fundamentals and applications in chemistry and biology, American Chemical Society, Washington, DC, 1996.

[6] Manallack, D. T., Ellis, D. D., and Livingston, D. J., Analysis of linear and nonlinear QSAR data using neural networks, J. Med. Chem., Vol. 37, 1994, pp. 3758-3767.

[7] Topliss, J. G. and Edwards, R. P. "Chance factors in studies of quantitative structure-activity relationships", J. Med. Chem., Vol. 22, 1979, pp. 1238-1244.

[8] John, G., Kohavi, R., and Pfleger, J., Irrelevant features and the subset selection problem, in Machine learning: proceedings of the 11-th international conference, Morgan-Kaufmann, 1994, pp. 121-129.

[9] Selwood, D. L., Livingstone, D. J., Comley, J. C. W., O'Dowd, A. B., Hudson, A. T., Jackson, P., Jandu, K. S., Rose, V. S. and Stables, J. N., Structure-activity relationships of antifilarial antimycin analogues, a multivariate pattern recognition Study, J. Med. Chem., Vol. 33, 1990, pp. 136-142.

[10] Sutter, J. M., Dixon, S. L., and Jurs, P. C., Automated descriptor selection for quantitative structure-activity relationships using generalized simulated annealing, J. Chem. Info. Comput. Sci., Vol. 35, 1995, pp. 77-84.

[11] Luke, B. T., Evolutionary programming applied to the development of quantitative structure-activity relationships and quantitative structure-property relationships, J. Chem. Info. Comput. Sci., Vol. 34, 1994, pp. 1279-1287.

[12] Rogers, D. R., and Hopfinger, A. J., Application of genetic function approximation to quantitative structure-activity relationships and quantitative structure-property relationships, J. Chem. Info. Comput. Sci., Vol. 34, 1994, pp. 854-866.

[13] So, S., and Karplus, M., Evolutionary optimization in quantitative structure-activity relationship: an application of genetic neural networks, J. Med. Chem., Vol. 39, 1996, pp. 1521-1530.

[14] Yasri, A., and Hartsough, D., Toward an optimal procedure for variable selection and QSAR model building, J. Chem. Info. Comput. Sci., Vol. 41, 2001, pp. 1218-1227.

[15] Hasegawa, K., Miyashita, Y., and Funatsu, K., GA strategy for variable selection in QSAR studies: GA-based PLS analysis of calcium channel antagonists, J. Chem. Info. Comput. Sci., Vol. 37, 1997, pp. 306-310.

[16] Izrailev, S., and Agrafiotis, D. K., Variable selection for QSAR by artificial ant colony systems, SAR and QSAR in Environ. Res., Vol. 13, No.3-4, 2002, pp. 417-423.

[17] Agrafiotis, D. K., and Cedeño, W. Feature selection for structure-activity correlation using binary particle swarms, J. Med. Chem., Vol. 45, 2002, pp. 1098-1107

[18] Cedeño, W., and Agrafiotis, D. K. "Application of niching particle swarms to QSAR and QSPR", Proceedings of the 14-th European Symposium on QSAR, Bournemouth, UK, September 8-13, 2002.

[19] Wettschereck, D., Aha, D. W., and Mohri, T., "A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms", Artificial Intelligence Review, Vol. 11, Iss. 1-5, 1997, pp. 273-314.

[20] Aha, D. 1998. "Feature weighting for lazy learning algorithms". In Liu, H., and Motoda, H., eds., Feature Extraction, Construction and Selection: A Data Mining Perspective. Norwell MA: Kluwer.

[21] Kohavi, R., Langley, P., & Yun, Y., "The utility of feature weighting in nearest-neighbors algortihms", In Proceedings of the European Conference on Machine Learning (ECML97) , 1997.

[22] Raymer , M. L., Punch, W. F., Goodman, E. D., Kuhn, L. A., and Jain, A. K., "Dimensionality Reduction Using Genetic Algorithms", IEEE Transactions on Evolutionary Computation, Vol. 4, 2000, pp. 164-171.

[23] Komosinski, M., Krawiec, K., "Evolutionary weighting of image features for diagnosing of CNS tumors", Artificial Intelligence in Medicine, 2000, Vol. 19 (1), P: 25-38, ISSN: 0933-3657.

[24] D. K. Agrafiotis, and V. S. Lobanov, "An efficient implementation of distance-based diversity metrics based on k-d trees", J. Chem. Info. Comp. Sci. , Vol. 39, 1999, pp. 51-58.

[25] Kennedy, J., and Eberhart, R. C., Particle swarm optimization, Proc. IEEE International Conference on Neural Networks, Perth, Australia, IEEE Service Center, Piscataway, NJ, IV, 1995, pp. 1942-1948.

[26] Kennedy, J., The particle swarm: social adaptation of knowledge, IEEE International Conference on Evolutionary Computation, Indianapolis, IN, IEEE Service Center, Piscataway, NJ, 1997, pp. 303-308.

[27] Shi, Y. H., and Eberhart, R. C., A modified particle swarm optimizer, IEEE International Conference on Evolutionary Computation, Anchorage, AL, 1998a.

[28] Shi, Y. H., and Eberhart, R. C., Parameter selection in particle swarm optimization, 7-th Annual Conference on Evolutionary Programming, San Diego, CA, 1998b.

[29] Rao, C. R. Some problems involving linear hypotheses in multivariate analysis. Biometrika 1959, 46, 49-58.

[30] Brits, R., Engelbrecht, A. P., and van den Bergh, F. A niching particle swarm optimizer. Proceedings of the 4th

Asia-Pacific Conference on Simulated Evolution and Learning 2002 (SEAL 2002), Singapore. pp. 692-696, 2002.

[31] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., "Optimization by simulated annealing", Science, Vol. 220 (459), 1983, pp. 671-680.

[32] Agrafiotis, D. K., Stochastic algorithms for maximizing molecular diversity, J. Chem. Info. Comput. Sci., 1997, 37, 841-851.

[33] Rassokhin, D. N., and Agrafiotis, D. K., Kolmogorov-Smirnov statistic and its applications in library design, J. Mol. Graphics Modell., 2000, 18, 370-384.Bowman, B., Debray, S. K., and Peterson, L. L. Reasoning about naming systems. *ACM Trans. Program. Lang. Syst., 15,* 5 (Nov. 1993), 795-825.

[34] Dorigo, M.; Gambardella, L. M. IEEE Trans. EVol. Comput. 1997, 1, 53-66 and references therein.

[35] Maddalena, D. J., Johnson, G. A. R., Prediction of receptor properties and binding affinity of ligands to benzodiazepine/$GABA_A$ receptors using artificial neural networks, *J. Med. Chem.*, **1995**, *38*, 715-724.

[36] Hirst, J. D., King, R. D., and Sternberg, M. J. E., *J. Comp.-Aided Mol. Design*, **1994**, *8*, 405-420.