The Use of a Self Organizing Map to Generate QSAR Sets

Rajarshi Guha
Pennsylvania State University

June 10, 2003

Introduction

I have been working on two aspects of QSAR modelling - model generation and development of algorithms. In the former area I have been involved in a study of artemisinin analogues.[1] Artemisinin is a well known anti-malarial. However certain analogues are neurotoxic and much research has been carried out to synthesize non neurotoxic analogues. The compounds in my current study have been studied using the CoMFA[2] methodology. My goal was to investigate whether the analogues could be modelled using the simpler ADAPT methodology. This is ongoing work and current results seem to indicate that the computational neural networks are able to produce good predictive models.

In this report I will be focussing on my work with the Kohonen self organizing map[3] (SOM) and its use in the generation of representative training, cross-validation and prediction sets (collectively referred to as QSAR sets).

Selection of QSAR Sets

The development of quantitative structure activity and quantitative structure property relationships are dependent on the creation of training and prediction sets. In some methodologies a cross validation set is also required. There are several techniques available for the generation of QSAR sets. The simplest is random selection; that is, the members of the dataset are randomly placed into the the training, cross-validation and prediction sets . Another method is activity binning, whereby the dataset is binned according to the experimental values of property under study. However an important feature of these methods is that they do not take into account the global features of the dataset such as similarity between molecules. Thus the composition of the QSAR sets are not necessarily representative of the dataset in terms of similarity. The aim of this study was to investigate whether a Kohonen self organizing map could be used to generate QSAR sets whose composition would be representative of that of the whole dataset and whether such QSAR sets would lead to better models.

Theory of the SOM

A SOM is an unsupervised neural network, in that it uses only the independent variables of the dataset, in this case, molecular structure descriptors. The SOM can be viewed as an elastic net of points, which are molded to the specific features of the compounds used for training. Training occurs as the SOM's neurons compete with each other for selection. At each training iteration, the selected neuron and its neighbors are modified to resemble the applied example compound.

Although SOM's can appear in a variety of forms,[3] this study implemented a map in the form of a square grid. In order that each neuron has the same number of neighbors the grid is designed so that it wraps around the edges, effectively transforming the grid of neurons into a torus.

Each compound in the training set is represented by a vector

$$X_i = (X_{i1}, X_{i2}, \cdots, X_{in})$$

where n is the number of molecular structure descriptors employed. Each neuron on the SOM grid is also a vector and is denoted by

$$m_i = (m_{i1}, m_{i2}, \cdots, m_{in})$$

The neurons on the grid are initialized with random vectors. The size of the grid is chosen by trial and error, guided by a rule of thumb described by Chen[4] which states that the number of neurons should be approximately one to three times the number of examples in the training set. The training process for a SOM is iterative. Each training iteration involves comparing each member of the dataset to all the neurons in the grid and determining the grid neuron that is closest, in terms of Euclidean distance to the submitted neuron.The grid neuron that is most similar to the input vector is the winner. Then, the winning neuron and the surrounding neurons are modified, according to this equation

$$m_i(t+1) = m_i(t) + h_{ci}(t)[x(t) - m_i(t)]$$

where t represents training iterations, m_i represents a grid neuron and x represents the training set vector. Here h_{ci} is termed the neighborhood kernel,[3] and determines which neurons are neighbors and how such neighboring neurons will be modified. Neurons that are further away (in a topological sense) from the winning neuron are modified to a smaller degree than neurons that are closer. The simplest neighborhood kernel is the bubble function[5] which is non-zero for the neighborhood but zero elsewhere. The map in this study implemented a Gaussian kernel[3] defined as

$$h_{ci} = \alpha(t) \left(-\frac{||r_c - r_i||^2}{2\sigma^2(t)} \right)$$

where $\sigma(t)$ is the neighborhood radius at time t which monotonically decreases with time. Thus, the number of neurons considered to be neighbors decreases as training progresses. The term $||r_c - r_i||$ represents the Euclidean distance between the winning neuron and the neighboring neuron. $\alpha(t)$ is the learning factor, and it influences the extent to which a neuron should be modified. Initially neurons within a large radius surrounding the selected neuron are considered neighbor neurons. The radius of the neighborhood is decreased in successive training iterations, and in the last stages of training only the nearest neighbors of the selected neuron are modified. The effect of this variable neighborhood function is that in the early stages of training the neurons are modified on a global scale, which leads to a global ordering. Near the end of training, the smaller neighborhood results in fine tuning of the map features. The neighborhood function thus controls the sensitivity of the map.

The actual modification is controlled by the learning factor, $\alpha(t)$. The learning factor is a function that monotonically decreases from 1 to 0 as training progresses. Once it reaches zero, training stops. Kohonen[3] mentions several ways of modifying $\alpha(t)$, and the implementation used in this study employs a constant decrement of 0.01, which implies that after 100 training iterations $\alpha(t)$ will be zero. This represents an upper limit on the number of training iterations.

After the SOM was trained, the results were analyzed to detect clusters of neurons. In this context, a cluster refers to neurons that have similar Euclidean distances from each other. As mentioned in Satoh[6] "recognition of boundaries of clusters in a Kohonen network is a difficult task". This was implemented by considering two neurons having a distance less than a user specified value to be a part of the same cluster. An arbitrary neuron was selected and assigned an arbitrary class label. Next, the distances to all the nearest neighbor neurons was evaluated. Using the rule mentioned above, the neighboring neurons were assigned classes; either the same class as the initial neuron or the opposite class. This procedure was then repeated with all the neurons in the grid.

The result of the cluster detection procedure was to divide the training set into two classes. As mentioned before, the arbitrariness of cluster detection lies in the fact that the user must specify a *distance threshold* value. Too small a value or too large a value results in all the training set members being assigned to the same class. It is thus clear that the threshold value cannot be chosen at random.

From SOM Classes to QSAR Sets

Having completed the classification of the dataset with the SOM we used these results to generate the actual QSAR sets. Labelling the two SOM classes as Class I and Class II we assume for example that Class I had 75% and Class II had 25% of the dataset respectively. Our premise is that QSAR sets which contain Class I and Class II molecules distributed according to their percentages in the overall dataset will be more representative of the overall dataset and thus should lead to good predictive models. Continuing with the example, let us assume that we have a dataset of 100 molecules and the SOM classifier splits this dataset in to 75 molecules in class I and 25 molecules in class II. We also assume that for the QSAR sets, the training set should contain 80% of the dataset and the cross-validation and prediction sets should each contain 10%. To make the training set composition similar to that of the overall dataset it will have 80 molecules, of which 75% (60 molecules) will be from class I and 25% (20 molecules) will be from class II. Similarly the cross-validation and prediction sets will each have 10 molecules, of which 75% (8 molecules) will be from class I and 25% (2 molecules) will be from class II. The breakup of the QSAR sets among the SOM classes discussed above is represented diagrammatically in Figure 1.

Implementation

To test this method we generated QSAR models using the 333-compound pcDHFR dataset that was studied by Mattioni[7] et al. To generate the QSAR sets we fed combinations of Dragon descriptors to a 13x13 SOM, and its output was used to generate the sets. After the QSAR sets were generated, we calculated ADAPT descriptors for the entire dataset of 333 molecules. This generated 248 descriptors per compound. The number of descriptors was then reduced via objective feature selection (using correlation and identical testing) to generate a reduced pool of 74 descriptors. The reduced pool of ADAPT descriptors was then used with the QSAR sets created from each of the Dragon descriptor combination, to generate linear (Type I) and non-linear (Type II and Type III) computational neural network (CNN) models using the ADAPT methodology. In total we used six combinations of Dragon descriptors to generate six sets of Type I, II and III QSAR models. The original study only reported Type III models and hence only these models will be discussed here.

Results

To generate fully non-linear Type III models a neural network is linked to a genetic algorithm, which searches for the best descriptor subsets to optimize the weights of the CNN. The results for the Type III models are summarized in Table 1. Though the number of descriptors in the two best models (see Table 1) is significantly lower than the number in the published model, they are of similar types, the majority being simple structural counts and topological path descriptors.

The MoRSE - 2D Autocorrelation Dragon descriptor combination generated QSAR sets which produced a Type III model whose prediction set RMS error was slightly larger than the original value whereas the prediction set error for the models that were generated from QSAR sets produced by using the GETAWAY and the MoRSE - WHIM Dragon descriptor sets match the predicted value. However, in all cases the RMS errors for the training and cross-validation set were significantly larger than those for the reported model. The higher cross-validation set error could indicate a loss of generalizability in these models. On the other hand the RMS errors for the training and cross-validation sets generated from the MoRSE - GETAWAY combination are much closer to those reported, though the prediction set error is now significantly larger. However, the attractive feature of the models generated from QSAR sets produced by MoRSE - 2D Autocorrelation, GETAWAY and MoRSE - WHIM Dragon descriptor sets are that they are 5- or 6-descriptor models. Furthermore the number of neurons in the hidden layers in these three models are all less than those published, indicating a simpler neural network.

The R^2 values for the two best models (i.e. the models using QSAR sets generated using the MoRSE - 2D Autocorrelation and MoRSE - WHIM Dragon descriptor combinations) are close to those reported for the best model. These are summarized in Table 2. The R^2 values for the training and cross-validation sets produced by the MoRSE - WHIM combination compare favorably to those published.

The plot of the predicted versus experimental values from the model generated using QSAR sets produced using the MoRSE - WHIM Dragon descriptor combination is shown in Figure 2. Molecules in the prediction set were classified as outliers if their predicted value was two standard deviations away from the mean. This criterion led to one outlier, whose structure is shown in Figure 3. The best published model also classified a single outlier. Ideally we would like the same outliers to be detected by either method. Although this is not the case, it should be noted that there is a structural similarity in the outliers presented in Figure 3. Although the original work does not provide an explanation of why that outlier is not predicted well, the fact that the SOM based technique predicts a structurally similar outlier indicates that that this technique is able to take into account similarity features of the dataset in the creation of the QSAR sets.

An important feature of the two best Type III CNN models (using QSAR sets generated from the MoRSE - 2D Autocorrelation and the MoRSE - WHIM Dragon descriptor combinations) is the consistency between the RMS errors for the training, cross-validation and prediction sets. In many cases a low RMS error for the prediction set would be indicative of a good predictive model. However, at the same time if the RMS errors for the training and cross validation sets are much lower than that of the prediction set it could indicate that the model lacks generalizability. Thus one would strive for models that have similar or consistent RMS errors for all the three QSAR sets. As can be seen, this does not occur for the published results. However, for the best Type III models generated by the SOM based method, though the RMS errors for the training and cross-validation sets are higher than those reported in the original model the RMS errors are more consistent over all the three sets. The standard deviation of the RMS errors for the three QSAR sets in the original model is 0.11 whereas the standard deviations in the case of the two best models noted above are 0.02 in both cases. This suggests that the models generated by this method have both sufficient generalizability as well as predictive ability. However, apart from the conclusions regarding the nature of the models themselves these results are indicative of the fact that the QSAR sets that are generated by the SOM are indeed similar to each other and representative of the data set as a whole thus leading to similar predictions made during training and after training (using the external prediction set).

We also reran the original 10-6-1 Type III CNN model five times with different QSAR sets generated using activity binning. These results are summarized in Table 4. As can be seen there is a large variation in the RMS errors for the three QSAR sets in each run. Furthermore, when compared to the RMS errors for the best Type III models generated using QSAR sets created by the SOM, we see that the SOM results in general lie midway between the RMS errors from the 10-6-1 models using QSAR sets from activity binning. We believe that this is a good indication for the consistency of results obtained using the SOM to generated representative QSAR sets.

Conclusions & Further Work

It thus appears that the technique of using a group of external descriptors coupled with a SOM to generate sets for QSAR modelling do generate improved results. The ability of the SOM to detect similarities in the dataset allows us to generate sets that are more representative of the overall data set. As a result models with fewer parameters (i.e., descriptors) are able to produce results comparable to the original model that had nearly twice the number of parameters and in addition produce consistent RMS errors over the three QSAR sets.

However though the study did lead to a model with better properties than the original it involved several arbitrary decisions such as the choice of initial descriptors for the SOM and choosing a specific SOM split out of the several runs. The algorithm could be substantially improved by implementing a method to optimize the threshold value, rather than having to specify a threshold value, so that classification of molecules in the SOM could be done automatically. Another improvement would be in the choice of initial descriptors. Since this study was exploratory in nature, we restricted ourselves to certain subsets of Dragon descriptors. This need not be the case and by including more or even all, Dragon descriptors the initial classification might be better. This would lead to a large number of descriptors. To reduce this descriptor pool one could apply the standard methods of feature selection (identical and correlation tests). Another method would be to take the first N principal components of all Dragon descriptors considered together, where N is the number of components that comprise 95% of the variance. This would avoid biases arising out of selection of specific descriptors.

Apart from the application of a KSOM to generate QSAR sets as described in this report, this technique could also be used to play a role in choosing whether an unknown molecule can be analyzed using a given QSAR model. Thus one might generate a model using a standard methodology based on a set of training and validation molecules. The real utility of the model would be if it could make correct (or reliable) predictions on a totally unknown molecule. However, it must be realized that the *query* molecule should have some features in common with the molecules that were used to train and validate the model. This is where a KSOM would be used to decide whether an unknown query molecule is sufficiently similar to the training and validation molecules. If so one would expect that the model would be able to make a reliable prediction for the query molecule.

In conclusion, the KSOM appears to be a useful and wide ranging tool in the QSAR modellers toolbox. It can play an important role at each stage of QSAR development, from set and feature selection to query compound selection.

References

1. Avery, M.; Alvim-Gaston, M.; Rodrigues, C.; Barreiro, B.; Cohen, F.; Sabnis, Y.; Woolfrey, J. *J. Med. Chem.* **2002,** *45,* 292.

2. Cramer, R.; PAtterson, D.; Bunce, J. *J. Am. Chem. Soc.* **1988,** *110,* 5959-5967.

3. Kohonen, T. *Self Organizing Maps;* volume 30 of *Springer Series in Information Sciences* Springer: 1994.

4. Chen, L.; Gasteiger, J. *J. Am. Chem. Soc.* **1997,** *119,* 4033.

5. Espinosa, G.; Arenas, A.; Giralt, F. *J. Chem. Inf. Comput. Sci.* **2002,** *42,* 343.

6. Satoh, H.; Sacher, O.; Nakata, T.; Chen, L.; Gasteiger, J.; Funatsu, K. *J. Chem. Inf. Comput. Sci.* **1998,** *38,* 210.

7. Mattioni, B.; Jurs, P. *Journal of Molecular Graphics and Modelling* **2003,** *21,* 391-419.

Dragon Descriptor Set	CNN Architecture	RMS Error		
		Training	Cross Validation	Prediction
BCUT & 2D Autocorrelation	5-3-1	0.63	0.68	0.79
BCUT & Galvez Indices	5-3-1	0.62	0.62	0.71
GETAWAY	5-2-1	0.73	0.73	0.65
MoRSE & 2D Autocorrelation	5-3-1	0.63	0.63	0.68
MoRSE & GETAWAY	9-5-1	0.49	0.59	0.76
MoRSE & WHIM	6-5-1	0.60	0.61	0.65
Published[7]	10-6-1	0.45	0.49	0.66

Table 1: Summary of Type III CNN Models Using Training, Cross Validation and Prediction Sets Created by the SOM and Dragon Descriptor Combinations.

	Training	Cross Validation	Prediction
MoRSE & 2D Autocorrelation	0.68	0.60	0.64
MoRSE & WHIM	0.75	0.78	0.64
Published[7]	0.83	0.78	0.64

Table 2: Comparison of R^2 Values for the Training, Cross Validation and Prediction Sets Which Were Created by the SOM using Dragon Descriptors (See Table 1 for model architectures)

	Random Sets			MoRSE - WHIM Sets		
	Mean RMSE	Std. Dev.	R^2	Mean RMSE	Std. Dev.	R^2
TSET	0.57	0.02	0.75	0.58	0.005	0.74
CVSET	0.59	0.03	0.73	0.57	0.0010	0.76
PSET	0.80	0.13	0.56	0.63	0.020	0.63

Table 3: Comparison of Statistics for Training, Cross Validation and Prediction Sets Generated Randomly Versus Sets Created by the SOM Using the MoRSE - WHIM Dragon Descriptor Combination. (Models were Type III CNN with a 6-5-1 architecture)

Serial No.	Training Set	Cross Validation Set	Prediction Set
1	0.45	0.59	0.81
2	0.45	0.52	0.73
3	0.44	0.63	0.95
4	0.64	0.64	1.00
5	0.67	0.61	0.95

Table 4: A Summary of the RMS Errors for the 10-6-1 Type III CNN Models Using 5 QSAR Sets Generated by Activity Binning

	Scrambled		MoRSE - WHIM	
	RMSE	R^2	RMSE	R^2
TSET	0.74	0.62	0.58	0.74
CVSET	0.85	0.48	0.56	0.76
PSET	0.90	0.39	0.59	0.63

Table 5: RMS Errors for a Type III CNN Model (6-5-1 architecture) Using a Scrambled Dependent Variable using Training, Cross Validation and Predictions Sets Created by the SOM Using the MoRSE - WHIM Dragon Descriptor Combination

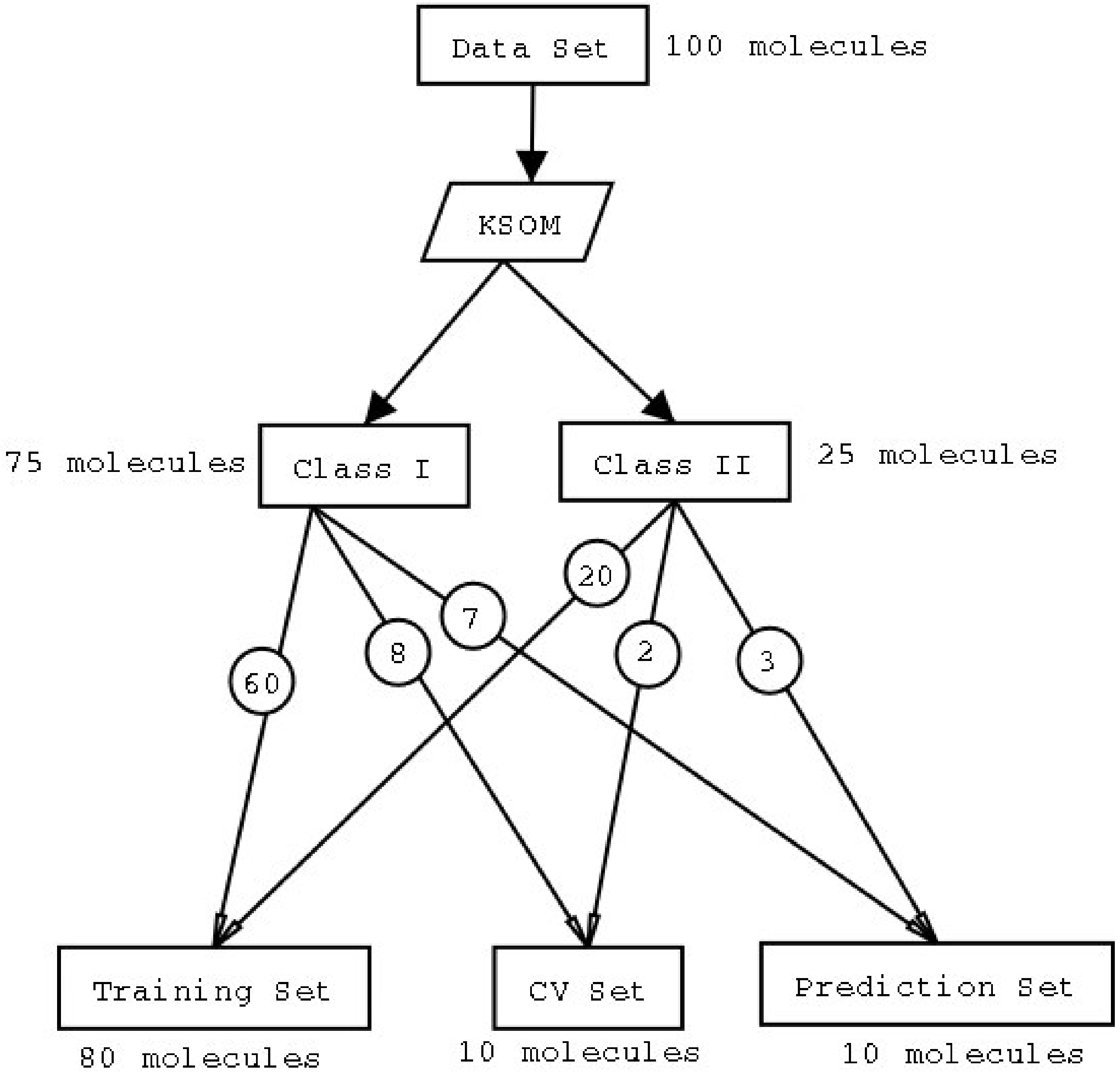

Figure 1: A diagrammatic representation of the method we use to generate QSAR sets from the SOM classification of the whole dataset. The numbers within circles are the number of molecules from that class that present in the specific QSAR set. This diagram assumes that the training, cross-validation and prediction sets contain 80%, 10% and 10% of the whole dataset respectively. In this example we assume that the SOM divides the set into two classes containing 75% and 25% of the whole dataset, respectively.

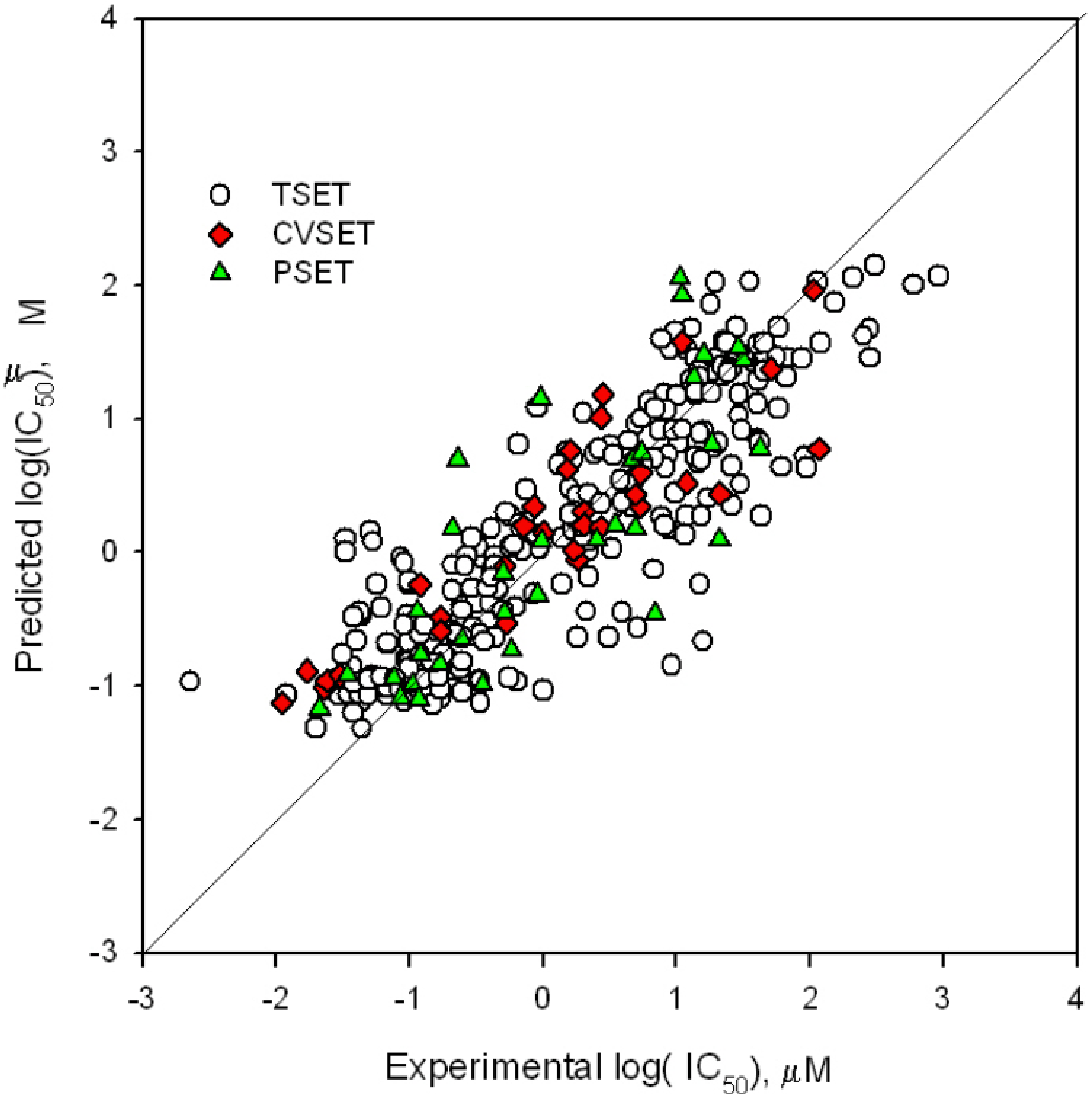

Figure 2: Plot of experimental vs. predicted $\log \mathrm{IC}_{50}$ for the Type III CNN model Generated Using Training, Cross Validation and Prediction Sets Created Using the SOM and MoRSE - WHIM Dragon Descriptor Combination.

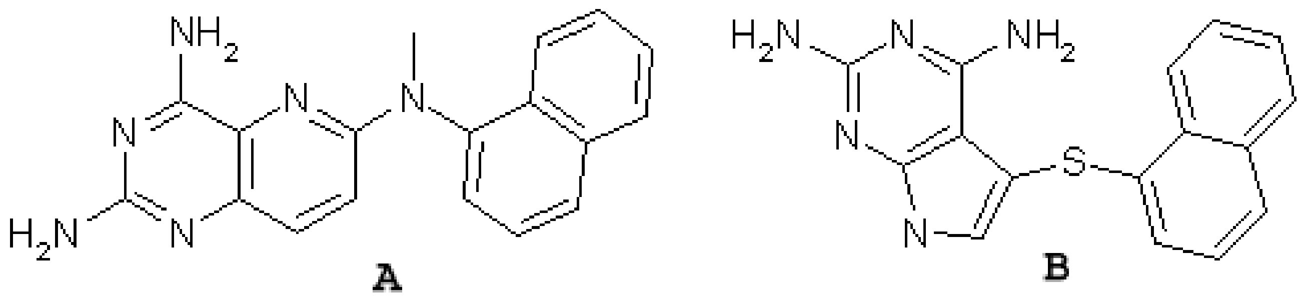

Figure 3: **A**. Outlier detected by best Type III CNN model in this study. **B**. Published outlier for the best pcDHFR model[7]