Bayesian Model Averaging for Ligand Discovery

Nicos Angelopoulos,*,† Andreas Hadjiprocopis,‡ and Malcolm D. Walkinshaw*,†

Department of Biological Sciences, Edinburgh University, Scotland, U.K., and Department of Computing, Higher Technical Institute, Nicosia, Cyprus

Received February 6, 2009

High-throughput screening (HTS) is now a standard approach used in the pharmaceutical industry to identify potential drug-like lead molecules. The analysis linking biological data with molecular properties is a major goal in both academic and pharmaceutical research. This paper presents a Bayesian analysis of high-dimensional descriptor data using Markov chain Monte Carlo (MCMC) simulations for learning classification trees as a novel method for pharmacophore and ligand discovery. We use experimentally determined binding affinity data with the protein pyruvate kinase to train and assess our model averaging algorithm and then apply it to a large database of over 3.7 million molecules. We compare the results of a number of variations on the central Bayesian theme to that of two Neural Network (NN) architectures and that of Support Vector Machines (SVM). The main Bayesian algorithm, in addition to achieving high specificity and sensitivity, also lends itself naturally to classifying test sets with missing data and providing a ranking for the classified compounds. The approach has been used to select and rank potential biologically active compounds and could provide a powerful tool in compound testing.

INTRODUCTION

High-throughput screening (HTS) is now a standard approach used in the pharmaceutical industry to identify potential drug-like lead molecules. Typically thousands, sometimes running up to one million, small molecule compounds are tested in a binding assay and affinity data is generated for each compound. More recently in a major initiative led by NCBI[1] (the National Centre for Biotechnology Information), information on the results of more than 800 bioassays has been collated and disseminated. The analysis linking biological data with molecular properties is a major goal in both academic and pharmaceutical research. We describe a novel approach for analyzing this activity data in terms of molecular "descriptor" properties.

Descriptor based approaches to ligand discovery have a long history in the field of QSAR.[2] Small drug-like molecules have been characterized by the well-known Lipinski Rule of Five[3] which essentially describes four properties (descriptors) common to most successful drug molecules. These are solubility, molecular weight, and the number of hydrogen bond donor and acceptor atoms in the molecule. There are however hundreds of characterizing descriptors for every molecule covering calculated physical properties like polarizability or shape properties describing for example the presence of ring structures or particular chemical groups in the molecule. Descriptors such as the weighted holistic invariant molecular (WHIM) descriptors[4] have been used in compound modeling for the prediction of molecular properties.[5−8] In this work, we use a very extensive set of 1600 descriptors calculated for each molecule with the program DRAGON 5.4.[9] The algorithm that has been developed selects sets of descriptors which correlate with biological activity. By identifying the key properties common to those molecules showing activity in the bioassay, we have a potentially useful method for predicting and identifying new active compounds.

The protein pyruvate kinase (PYK)[10] has been selected for the development of this work because of its importance as a potential anticancer and antiparasitic drug target. PYK acts as a tetramer and catalyzes the last step in the breakdown of sugar to form pyruvate. This pathway is required for survival of the trypanasomatid parasites that cause diseases, which include sleeping sickness and leishmaniasis.[11] Human PYK is also implicated in cancer pathogenesis and is crucial for the altered metabolism observed in tumor cells.[12] Inhibitors specific to the various pyruvate kinases are therefore of significant medical interest.[13]

In this paper, we compare three machine learning paradigms and their ability to increase information extracted from experimentally determined affinity to PYK: Bayesian model averaging techniques, Support Vector Machines, and Neural Network architectures. Affinity data are augmented with a large number of descriptors. In the Bayesian case, classification trees[14,15] are built using an MCMC approach. These trees are also known as decision trees and utilize the descriptors by creating split rules on their values providing predictive models of activity. We compare a number of algorithms from the three paradigms on a small part of the data holding out a 10% section for validating the algorithms. A comprehensive comparison is performed for three of the validated algorithms. In addition to performing at least as well as the SVM and NN algorithms, the Bayesian paradigm offers two advantages: intuitive interpretation of affinity probabilities as match ranking and the ability to average over missing descriptors in the database.

* To whom correspondence should be addressed. E-mail: n.angelopoulos@ed.ac.uk.
† Edinburgh University.
‡ Cyprus Higher Technical Institute.

10.1021/ci900046u CCC: $40.75
Published on Web 06/02/2009

Table 1. Descriptor Groups Produced by the Dragon Software

ID	descriptor name	population
1	constitutional descriptors	48
2	topological descriptors	119
3	walk and path counts	47
4	connectivity indices	33
5	information indices	47
6	2D autocorrelations	96
7	edge adjacency indices	107
8	BCUT descriptors	64
9	topological charge indices	21
10	eigenvalue based indices	44
11	Randic molecular profiles	41
12	geometrical descriptors	74
13	RDF descriptors	150
14	3D MoRSE descriptors	160
15	WHIM descriptors	99
16	GETAWAY descriptors	197
17	functional group counts	154
18	atom centered fragments	120
19	charge descriptors	14
20	molecular properties	30

Finally we approach our central objective of ligand discovery by retraining the main Bayesian algorithm on the full data set and screen the 3.7 million unique compounds contained in the database Eduliss 2 for potential active compounds.

DATA SETS

Biochemical Data. The data used in this analysis were taken from results published by the NIH Chemical Genomics Center.[16] In this assay, purified pyruvate kinase from the bacterium *Bacillus stearothermophilus* was used to generate ATP from ADP using its substrate phospoenolpyruvate. The generation of ATP was measured by the emission of light when ATP reacted with the enzyme luciferase. Over 50 000 small molecule compounds were tested for their inhibitory (and stimulatory) effects and an IC_{50} value for active compounds was measured. The IC_{50} value is the concentration of compound required to reduce the measured PYK activity by 50%. The curated database contains information classifying the compounds as active, inconclusive or inactive. There are 812 compounds labeled as inconclusive, which were removed from the set for the purposes of this study.

Prediction of novel ligands, binders to PYK, is from a database that stores catalogues of readily available chemical molecules. We used the Eduliss 2 database,[17] which is a web accessible collection of over 6 million, mainly commercially available, small molecule compounds. For each compound the 3D structure is stored as an SDF file along with the calculated descriptor values.

We used the Dragon software to produce chemical descriptors for the molecules both in the assay experiments and the database. There are 1665 descriptors that can be calculated for each molecule. Each descriptor belongs to one of the 20 groups shown in Table 1. However, the descriptor producing software is sometimes unable to calculate all the descriptors for an input molecule. For the assay molecules, 1571 of those descriptors were reliably produced. In addition, from the 3 776 760 unique compounds in Eduliss 2, only 1 537 089 have a complete set of the 1571 descriptors used here. It is thus important that classification can cope with missing values.

Figure 1 charts the missing values in the unique compounds data set. The area of very high missing percentages around the $x = 1500$ region correspond to the 13 descriptors in the "charge descriptors" group, The area of medium missing values ($y = 623\,791$) around $x = 900-1000$ corresponds to the whole set of descriptors in the 14th group (3D MoRSE descriptors).

ALGORITHMS

Bayesian Learning. Bayesian model averaging[18] (BMA) is a statistical machine learning technique where the full posterior distribution $P(M|D)$ over possible models M given data D is approximated. That is, the result of learning is a large number of models each with an associated probability. In this respect, BMA is unlike SVM and NN, which seek out a single model.

The Bayesian formulation employed here, expresses the posterior in terms of a likelihood function $P(D|M)$, which measures the goodness of fit of model M in explaining data D and a prior probability function $P(M)$, which can incorporate existing knowledge about the appropriateness of model M. Thus, we have

$$P(M|D) \propto P(D|M) \times P(M) \quad (1)$$

Because the likelihood is not a probability function, the RHS of (1) is normalized by dividing with $\Sigma_M P(D|M)P(M)$. Approximating the posterior is then necessary as the number of models is, in all but trivial cases, far too large to fully enumerate. BMA exploits Markov chain Monte Carlo algorithms to stochastically integrate the above summation.

Markov chain Monte Carlo methods for learning classification and regression trees were introduced by Chipman et al.[19] and applied to the analysis of breast cancer data. A more detailed exposition can be found in Denison et al.[20] These methods employ MCMC algorithms, which depend on the definition of both a prior and a set of functions for stochastically moving between models.

In this paper, we follow Angelopoulos and Cussens[21] because their approach only depends on the definition of a prior. Proposal steps are performed by the software automatically without the need for separate functions for moving between models, as the space is perceived as a partially constructed search structure. We develop their programs to cope with the high-dimensionality of the data and employ multiple chains to improve performance.

Classification and regression trees use a number of decisions on features/descriptors to classify each datum to one of a number of possible classes. An example is shown in Figure 2. In this paper, we limit discussion to classification trees because we are interested in classifying them into two discrete alternatives: active and inactive. Regression trees allow classification over continuous spaces.

In the trees displayed here, nodes are numbered sequentially. Internal nodes are labeled by a descriptor and an associated split value. Molecules whose value for the label descriptor is less or equal to the split value follow the left branch, where the rest follow the right branch. Leaf nodes classify a molecule as active with probability proportional to the number of training set actives in that leaf (see Figure 2 for an example). Tree T and its parameters Θ, when applied to molecules M, define a distribution over classes C,

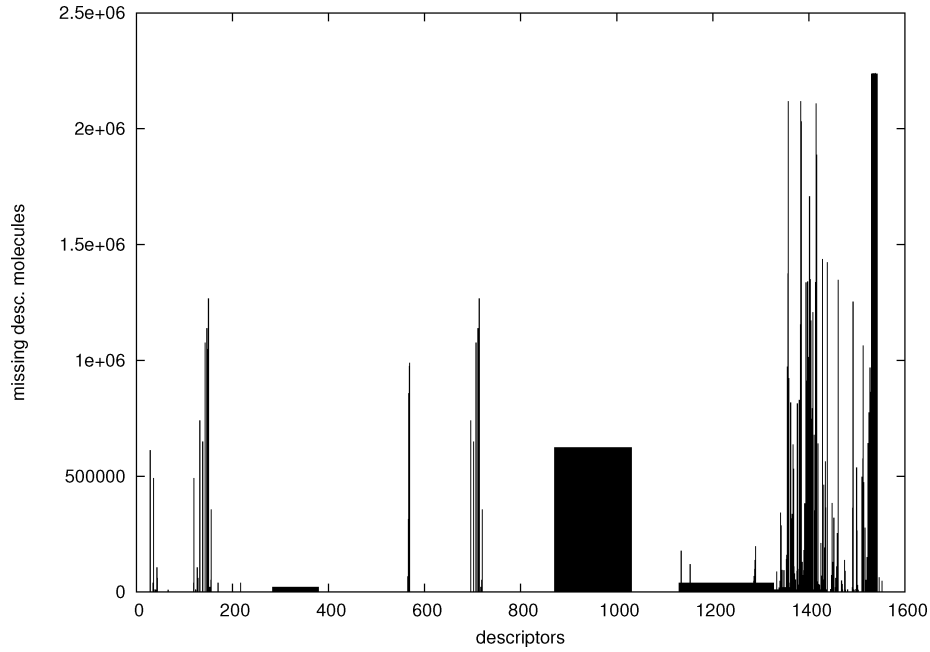

Figure 1. Number of unique molecules in Eduliss 2 missing a specific descriptor (y) against descriptors (x).

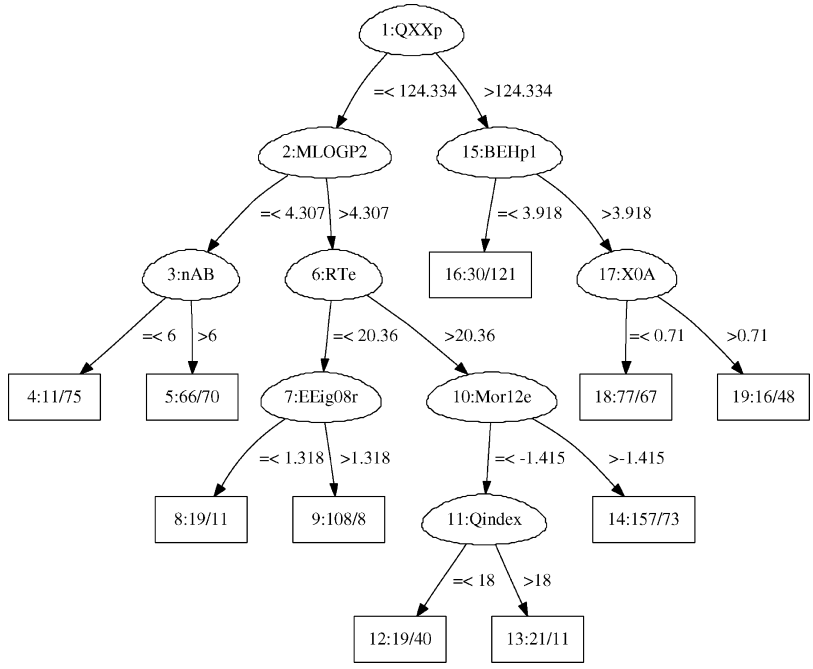

Figure 2. Tree at iteration 1000 of an MH run. Oval nodes are decisions on a single descriptor and the edges downward from the node carry a decision on an associated value for this descriptor. Leaf nodes present a distribution over classes.

$p(C|T,\Theta,M)$. Given that we are interested in two classes: active (A) and inactive (I), then for the tree T with b number of leaves, Θ is the matrix of parameters with b rows each of the form $[p_{i,\mathbf{A}},p_{i,\mathbf{I}}]$ where $1 \leq i \leq b$. So Θ defines a distribution over the possible classes at every leaf whereas, each tree in T assigns each molecule to a leaf according to its descriptor values.

Because the number of possible trees grows exponentially on the number of descriptors and their possible values, a variety of approximate algorithms have been proposed to learn such trees from data.[22,23] Metropolis−Hastings (MH) is a stochastic algorithm that has been used extensively in the literature to sample from the posterior probability distributions that can not be fully determined. MH is an MCMC algorithm that uses stochastic moves to construct a Markov chain. Under weak conditions the chain approximates the desired posterior. Here we are interested in the probability of a tree in explaining our data: $P(T|M)$, where M is a set of descriptors for a set of molecules. MH constructs a chain of visited trees by stochastically proposing at each iteration a new tree that is either accepted or rejected. The move at iteration i is with accordance to proposal distribution $q(T_i,T_i')$ and the proposed tree T_i' is accepted with probability $\alpha(T_i,T_i') = \min(1,R)$ where

$$R = \frac{p(T_i')P(C|T_i',M)q(T_i,T_i')}{p(T_i)P(C|T_i,M)q(T_i',T_i)} \tag{2}$$

The quotient in (2) has three main components. The first is based on the prior probability of the trees (i.e., $p(T)$). The

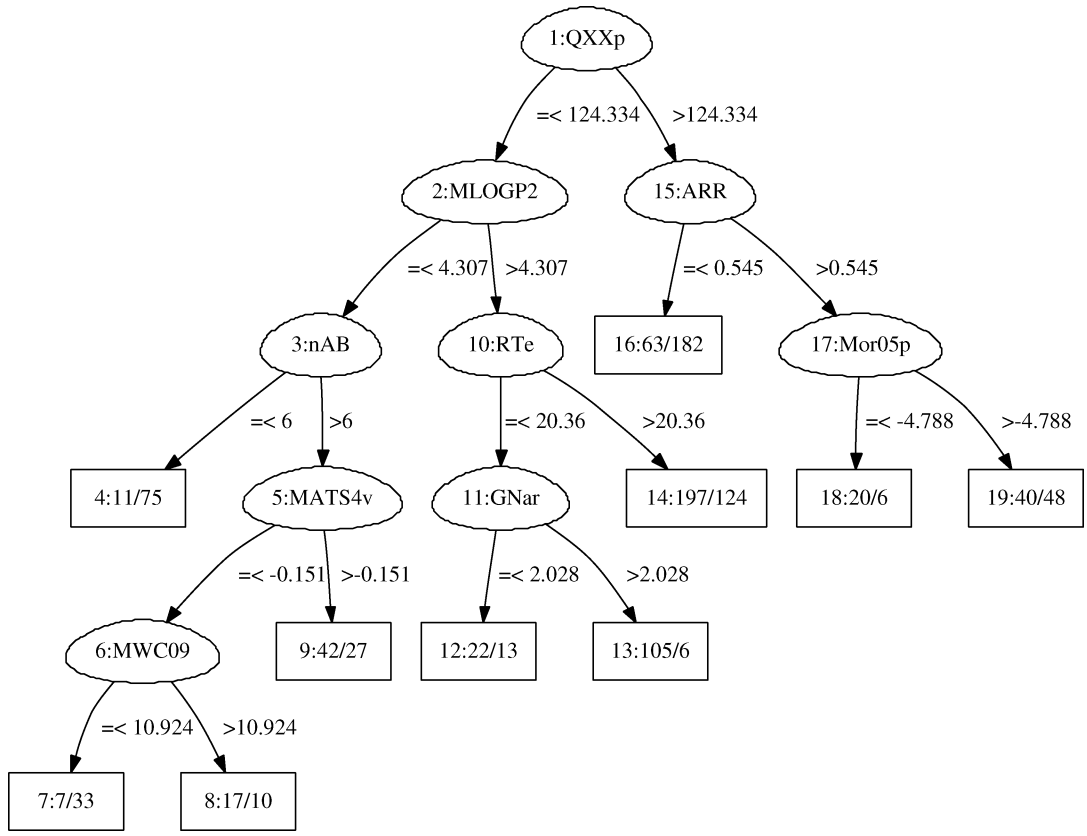

Figure 3. Tree at iteration 10 000 of same MH run as that of the tree in Figure 2.

Table 2. Classification Results for Experiments Ran on a Train-and-Test Partition of Data Set A

	true +ve		true −ve	
	mean	std error	mean	std error
MH	44.7	1.65	42.33	1.18
MH-Aw2	45.7	0.72	39.33	1.90
MH-T3	45.3	1.44	43.67	1.52
FFNN	46.8	0.98	42.6	1.09
FFNNE	48	1	42.66	1.53
SVM	49	n/a	43	n/a
Mixed	49.3	0.54	45	0.47
Mixed-Aw2	49	0.471	45.33	0.272
Mixed-T3	49.33	0.728	46.33	0.272

Table 3. Sensitivity, Specificity, and Runtime Figures for Fold 1 Data Set A

	sensitivity	specificity	time (s)
MH	0.7707	0.7298	18888
MH-Aw2	0.7879	0.6781	24636
MH-T3	0.7810	0.7529	61921
FFNN	0.8069	0.7345	96106
FFNNE	0.8276	0.7355	1121671
SVM	0.8448	0.7414	13
Mixed	0.85	0.7759	166175
Mixed-Aw2	0.8448	0.7816	214474
Mixed-T3	0.8505	0.7988	637423

second is based on the goodness of fit for a tree in explaining the data (i.e., $P(C|T,M)$) and is known as the marginal likelihood.[19] The third term is the quotient of transition probabilities.

We use a program which builds trees recursively. Starting with the trivial tree containing a single node, the program considers whether to split or not this root node. In the latter case, the process terminates and the single node tree is returned. In the former case, the program increases the depth by 1 and recursively calls itself to construct the left and the right subtrees. The probability with which the decision to split or not depends on the depth η of each node. The intuition being that the prior penalizes deep trees as they are liable to overfitting. Once a splitting variable has been decided upon, the associated value is selected uniformly from the variable's values present in the data set. Note that a minimum restriction of leaf node is set to 20 in all our experiments.

Splitting occurs with probability, $\zeta(1 + d_\eta)^{-\xi}$, where d_η is the depth of node η and ζ and ξ are user defined parameters controlling the size of the trees. Thereafter we assume $\zeta = 0.95$ and $\xi = 1$. Our approach follows the theory of Angelopoulos and Cussens.[21] The prior and the proposal are tightly coupled and the acceptance ratio becomes easy to compute. This is achieved by encoding the prior as a Distributional Logic Program[24] and letting the main component of the proposal be a backward jump to one of the probabilistic choices on the branch that led to T_i. From that choice onward T_i' is resampled according to the prior. Let π be the sequence of probabilistic choices that led to model M_i, π_b an element uniformly chosen from π, and π_f an alternative choice to π_b taken from the same distribution, then in terms of (2) the program achieves the following reduction

$$\frac{p(T_i')q(T_i, T_i')}{p(T_i)q(T_i', T_i)} = \frac{\pi_f}{\pi_b} \tag{3}$$

Our approach exploits the one-to-one correspondence of probabilistic choices to nodes in the tree to effect a uniform

Table 4. Ten-Fold Cross Validation for SVM, FFNN, and Mixed on Data Set A

	1	2	3	4	5	6	7	8	9	10	av
FFNN T^+ mean	46.8	42.5	43.8	38.8	40.4	44.5	46.2	42.7	41.4	44.8	43.2
FFNN T^+ error	0.98	1.19	1.19	1.66	0.94	0.98	0.72	2.16	1.44	0.47	1.17
FFNN T^- mean	42.6	48.9	44.9	39.8	41.8	46.3	45.6	44.8	48.2	46.0	44.9
FFNN T^- error	1.09	0.27	0.54	1.25	0.94	0.27	1.91	0.27	0.98	1.09	0.86
SVM T^+	49	44	44	39	41	47	49	47	42	51	45.3
SVM T^-	43	48	46	41	39	47	45	49	48	41	44.7
Mixed T^+ mean	49.3	41.7	46.3	40	37.3	42.7	48.3	45.3	39	45.7	43.6
Mixed T^+ error	0.54	0.27	0.98	0.94	0.54	0.54	0.27	0.54	0.47	0.27	0.53
Mixed T^- mean	45	50	49.7	43.7	42	51	47	49.3	52	48.3	47.8
Mixed T^- error	0.47	0	0.72	0.27	0	0	0.47	0.54	0	0.27	0.27
FFNN sensitivity	0.81	0.73	0.76	0.67	0.70	0.77	0.80	0.74	0.71	0.77	0.745
FFNN specificity	0.73	0.84	0.77	0.69	0.72	0.80	0.79	0.77	0.83	0.79	0.774
SVM sensitivity	0.84	0.76	0.76	0.67	0.71	0.81	0.84	0.81	0.72	0.88	0.781
SVM specificity	0.74	0.83	0.79	0.71	0.67	0.81	0.78	0.84	0.83	0.71	0.771
Mixed sensitivity	0.85	0.72	0.79	0.69	0.64	0.73	0.83	0.78	0.67	0.78	0.751
Mixed specificity	0.78	0.86	0.85	0.75	0.72	0.88	0.81	0.85	0.90	0.83	0.824

backtracking strategy. The probabilistic part of this program is similar to that presented in Angelopoulos and Cussens.[21] However, our prior differs in how features and associated values are selected. Because of the high-dimensionality of our data, the approach taken here is to construct the set of possible split values on-the-fly and after a split descriptor is selected rather than using cached values. As discussed in the following section, we also employ multiple chains for the same reason.

The Monte Carlo part of MH allows approximation of the posterior from the constructed chain. The posterior probability of tree T is equal to the number of times it appears in the chain ($\#T$) over the total number of iterations I, ($P(T|M,C) = (\#T)/(I)$). More importantly the framework assigns probability to features of the trees. By model averaging principles, the class of a datum can be computed over the whole chain and over a number of independent chains.

Support Vector Machines. Support Vector Machines (SVM) is a supervised learning method for classifying data into two classes.[25] During training, the SVM algorithm attempts to find a hyperplane to separate the training data into two distinct groups. Assuming linearly separable data

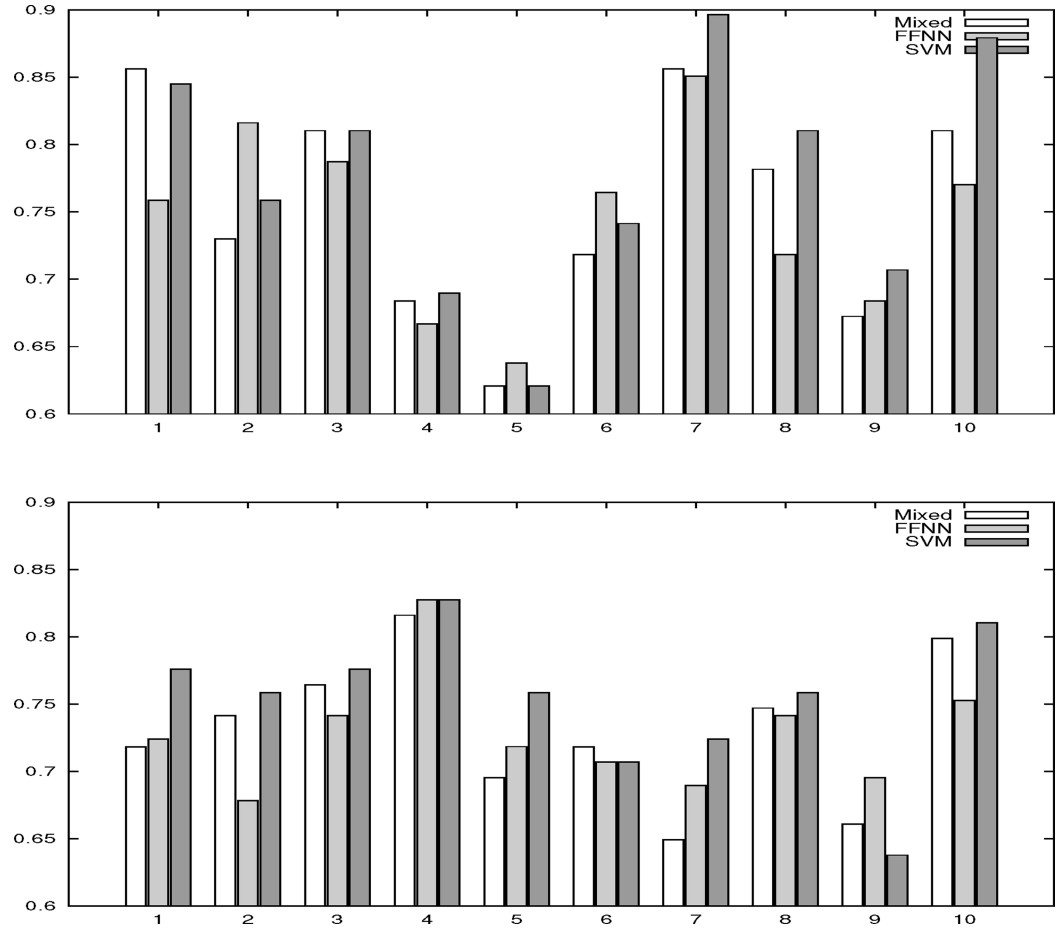

Figure 4. Sensitivity for data set B (top) and data set C. Averages: $FFNN_B = 0.742$, $FFNN_C = 0.729$, $SVM_B = 0.776$, $SVM_C = 0.753$, $Mixed_B = 0.754$, $Mixed_C = 0.731$.

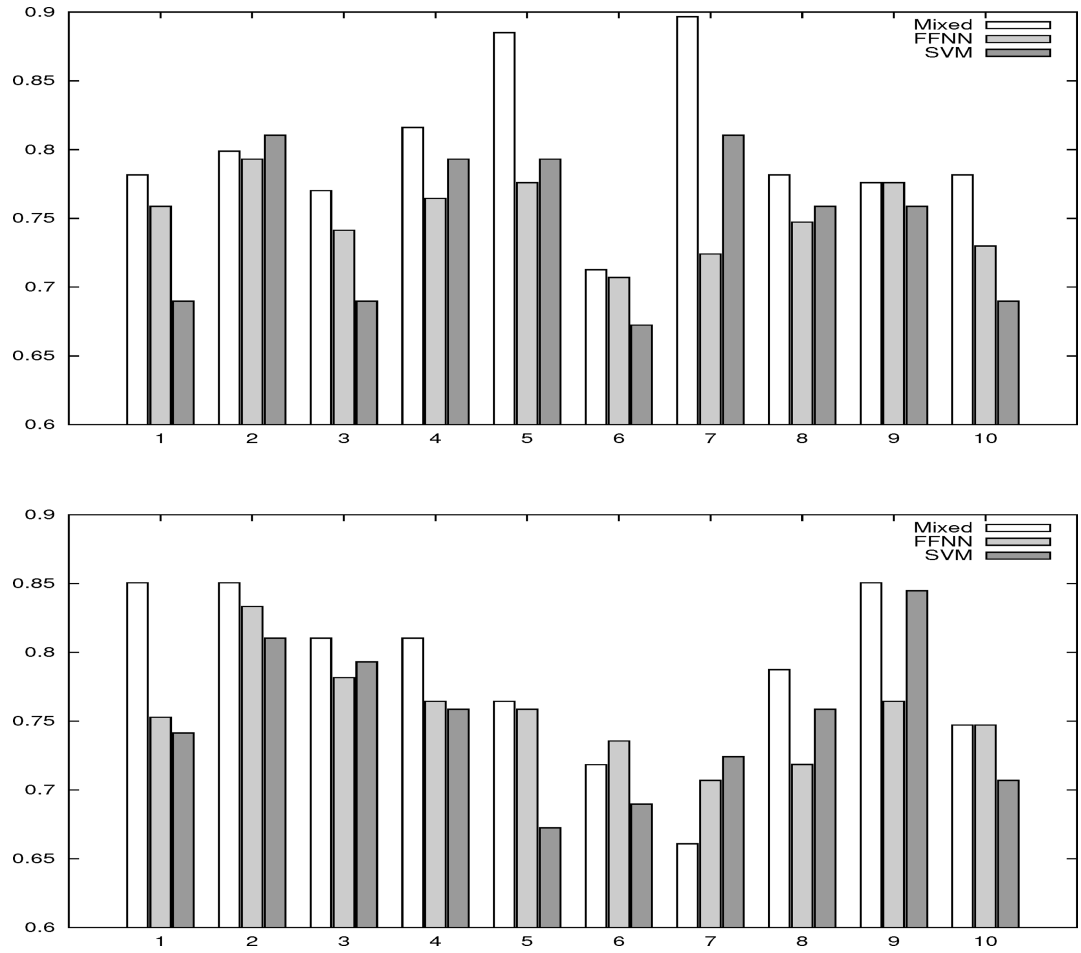

Figure 5. Specificity for data set B (top) and data set C. Averages: $FFNN_B = 0.758$, $FFNN_C = 0.750$, $SVM_B = 0.747$, $SVM_C = 0.750$, $Mixed_B = 0.800$, $Mixed_C = 0.785$.

(i.e., data whose two classes are separated by a line/plane/hyperplane), there are an infinite number of hyperplanes which can discriminate between the two. In this respect, SVM chooses the hyperplane which maximizes the margin between the two groups on the assumption that the larger the margin the lower the error of the classifier when dealing with unknown, unclassified data.

In the case of linearly inseparable data, the original SVM algorithm has been extended so as to provide nonlinear classification via the use of kernel functions that map the input data into a higher dimensional space where linear separation can be achieved.

The main advantage of SVM is that because it is formulated as a quadratic programming problem, there are no local minima (a serious disadvantage for Neural Networks); the algorithm is guaranteed to reach the globally optimum solution. On the other hand, SVM, again as a problem of quadratic programming, is associated with a training complexity (convergence time and memory) which is highly dependent on the number of training data vectors. This is not the case with feed-forward neural networks whose performance depends mainly on the dimensionality of the data. Additionally, SVM's performance depends on model selection, that is, choice of kernel function and its parameters and the margin parameter.

A tutorial about SVM is provided by Burges.[26] Meyer et al.[27] compare the performance of SVM with a large number of other classification methods. The results of our tests were obtained using an implementation of SVM from[28] with default settings.

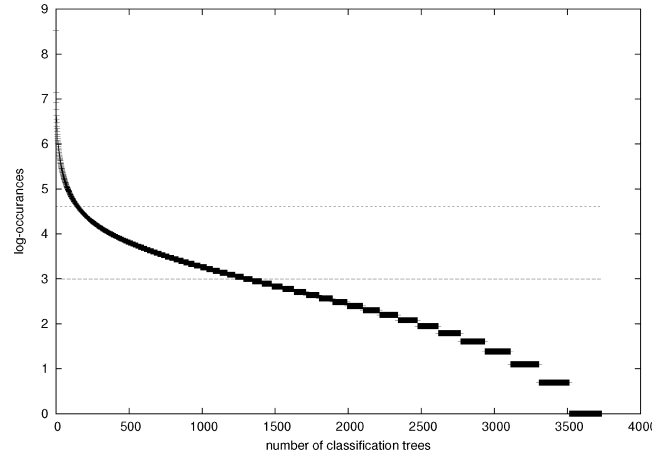

Figure 6. Log-occurrence values for each model in chain of Mixed algorithm, which was trained with all 1164 examples of data set A. The horizontal lines mark $\log(20) = 2.9957$ and $\log(100) = 4.6052$.

Feed-Forward Neural Networks. Feed-forward neural networks (FFNN) are computational techniques inspired by the physiology of the brain, epitomising the connectionist view of distributing knowledge and decision-making among a network of massively connected, simple processing units interacting at a local level. The mathematical foundation of FFNN lies on Kolmogorov's general representation theorem

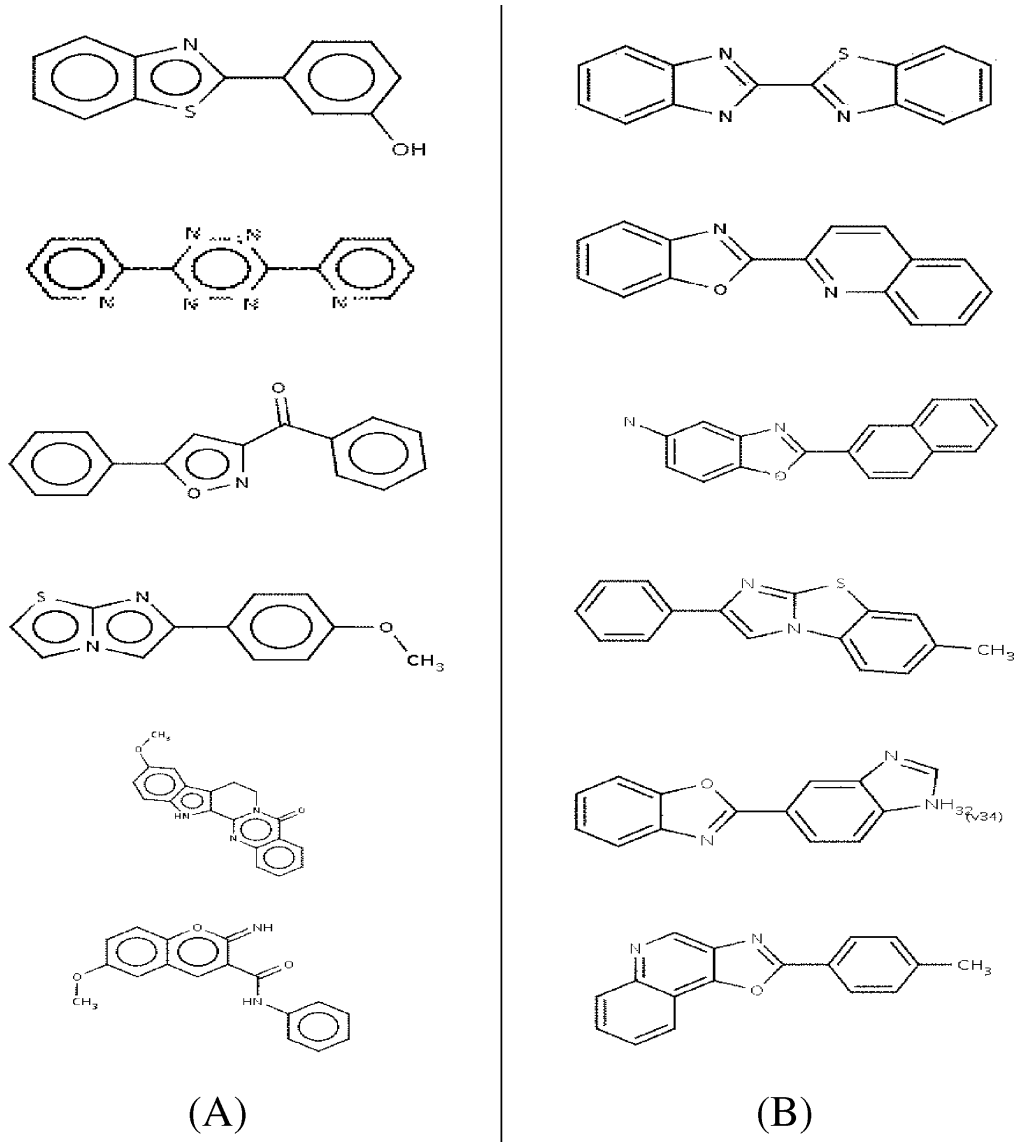

Figure 7. (A) Six active molecules from the training set. (B) The six molecules scoring highest probability of activity. From top to bottom their probabilities are 0.904738, 0.904384, 0.902034, 0.901443, 0.901044, and 0.900508.

which states that any continuous multivariate function can be represented by superpositions and compositions (a network) of univariate functions (of neurons). Furthermore, FFNN are universal function approximators; a FFNN with a single hidden layer employing as many nonlinear units as required, can approximate any continuous function arbitrarily well.[29]

FFNN training[30] is exercised through a supervised process by which the network is presented with a sequence of inputs and respective desired outputs. "Learning" occurs by adjusting the free parameters of the network (the weights) in a way that the discrepancy between the actual and desired responses (the error) is minimized. In general, the adjustment of the weights follows the steepest-descent direction of the error-weights surface. Other, non-gradient-descent training methods do exist, for example, based on genetic algorithms or borrowed from thermodynamics, for example, simulated annealing.

The ability of a trained FFNN to classify or map correctly a previously unseen input vector is called Generalization. Largely, generalization depends on how accurately the training data captures the underlying data-generation model and the architecture of the FFNN. The latter being a major problem since there is no systematic method for selecting the most appropriate network model for the data at hand other than rules of thumb. Additionally, a FFNN may demand excessive resources when dealing with high-dimensional data under the strain of adjusting a huge number of parameters (weights) because of combinatorial explosion.

FFNN entities (FFNNE)[31] have been proposed as a means of dealing with high-dimensional data within a neural network framework but without the side effects usually associated with monolithic FFNN. A FFNNE consists of a large number of FFNN forming a highly interconnected network arranged in layers. Each input-layer FFNN deals with a relatively small subset of the parameters of the input data, sending signals to FFNN of the layers ahead.

EXPERIMENTS

Our overall objective is to extract descriptor-based models from the data sets described in the previous section. We approach this in three stages. First, we select a subset of the inactives and add them to the active compounds. We run

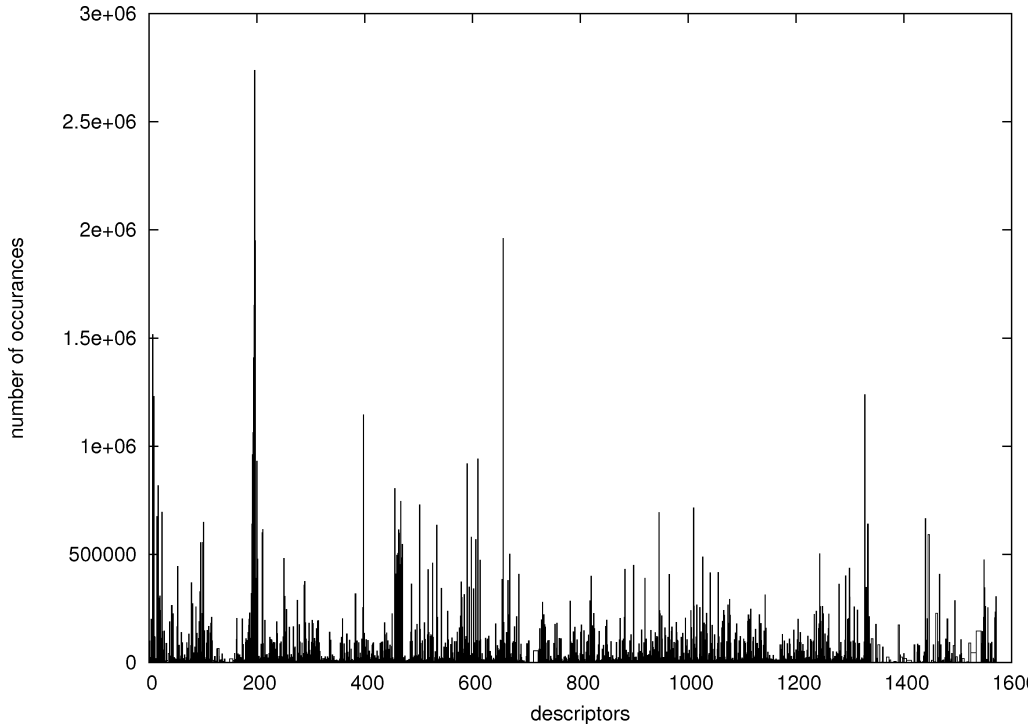

Figure 8. Classification of all unique molecules of Eduliss 2. Frequencies for all the descriptors in the subset of classification trees that have $U > 20$.

learning experiments on a 90% portion of the derived data set holding 10% as a test set. This stage allows a general overview of how well a variety of algorithms from the three paradigms perform on data of known class. Then, the second stage validates three of the algorithms on a 10-fold cross validation for the derived data set and on two additional data sets, which are constructed from alternative selections of the inactives. In the third stage, we use 100% of the first data set to retrain the main algorithm and use the output to classify 3.7 million molecules of unknown class.

The 50 000 molecules of the experimentally determined affinities have a vast bias toward nonbinding compounds. Precisely 582 molecules for which we can calculate a reasonably full set of descriptors have been classified to be active (positive). To avoid dominance of the inactive (negative) examples and to reduce the computational power needed, we randomly selected 582 inactives for the original data set and added them to the active molecules. We will refer to this collection of data as data set A. We then chose 524 molecules from each active and inactive sets to form the training set with the remaining 58 molecules from each class as the test set.

Single-Fold Validation. We first ran one million long iterations of the MH algorithm. We will refer to these as MH runs. The algorithm did well to explore branches from depth 2 or more. However, it rejected the vast majority of proposed trees that radically changed the current tree, that is, when backtracking to depth 0 or 1. The trees in Figure 2 and Figure 3 are from the same run and display the chain tree at iteration 1000 and 10 000, respectively. It can been seen that the 3 leftmost nodes near the root of the tree are the same in both of the trees. A major contributing factor to this behavior is the multidimensionality of the data. The longer the chain stays at the same root the harder it becomes for nodes within the tree to be improved upon by a single move.

Table 2 shows results for the experiments we have ran on the first fold selection of data set A. Each row provides the mean and standard deviation over 10 runs. Each run starts from different initial conditions by selecting a different seed to the random number generator. Two sets of values are shown: one for the correctly identified active molecules and one for the inactive molecules. Note that our test set contains 58 active and 58 inactive examples. The first row in Table 2, MH, gives the values for the 1 million iterations of the MH algorithm which we have already described. On average out of the 58 active test molecules it correctly classifies 44.7, while it does slightly worst in classifying inactives.

In a second approach (MH-Aw2), we increased the relative importance of class A to be twice that of class I. The results are similar to the standard runs and are shown in the MH-Aw2 row of Table 2. The lack of significant increase in true actives classification possibly points to the marginal effect the likelihood has in improving the search beyond that threshold. Furthermore, the slight increase noted, comes at the price of decreased accuracy in predicting inactives.

We also ran tempering chains (MH-T3) on data set A. In tempering, a number of hot chains are run in addition to the main chain. The main motivation is that the hot chains move in smoother likelihood spaces and thus might be less likely to get stuck in local maxima. Following standard practice, we use temperatures in the region of $0 < \tau < 1$ that are used as exponents to the likelihood function. Thus the cold chain corresponds to temperature $\tau = 1$. In the tempering experiments, we use $T = (0.833, 0.714, 0.625)$. The MH-T3 row in Table 2. shows that tempering helps to slightly improve classification of inactive test data. Inspection of the chains, although showing improvement of mixing at the root, indicates that chains still get stuck to a single root after a few thousand iterations.

To deal with the high-dimensionality of the feature (here descriptor) space in the Bayesian approach, we propose

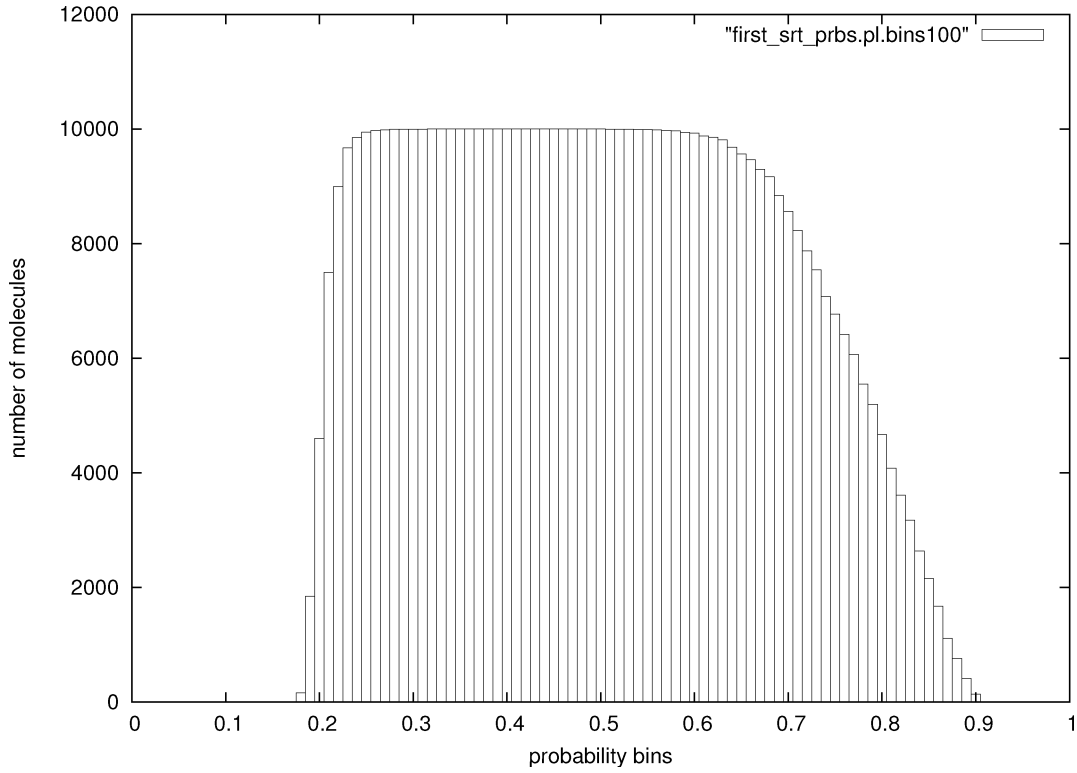

Figure 9. Classification of all unique molecules of Eduliss 2. Probability bins (width = 0.01) against number of molecules.

restarts of the MH algorithm. We ran 100 chains at 100 000 iterations each and combine their outputs to a single chain. We refer to these as runs of the Mixed algorithm. Bayesian model averaging provides a robust theoretical framework for this approach. In effect, the original MH algorithm is interleaved with an independent sampler.

The results from running the Mixed algorithm are shown in Table 2 on the row labeled Mixed. There is a clear improvement in the quality of the results achieved and the robustness of the system. Similarly to the standard MH we also ran experiments with varying: actives weight equal to 2 and three hot chains. We will refer to these experiments by Mixed-Aw2 and Mixed-T3, respectively. By using a weight of 2 for actives, we see marginal improvements in the results. In the case of tempering we also see some but no substantial improvement, which we attribute to the fact that the independent sampler complements the MH well, so there is no extra benefit in using tempering for the Mixed algorithm.

We ran two sets of NN-based experiments: FFNN and FFNNE (entities). For the former, a network with 1513 inputs was employed. The network consists of 6 hidden layers with 1261, 991, 606, 291, 97, and 29 units each, respectively, and it is fully connected. Training was stopped after 1000 iterations. The FFNNE consists of 499 FFNN, each configured to have a number of inputs between 350 and 850 and three hidden layers, yielding a total of about 100 million neurons.

Finally we ran SVM experiments with the *SVMlight* library using the linear kernel function. We have also experimented with alternative kernels but the linear version as reported here produced the best results consistently.

In Table 3, we present the sensitivity and specificity[32] results for all the experiments described so far. Letting T^+, T^-, F^+, and F^- be the numbers of true positive, true negative, false positive, and false negative, then

$$\text{sensitivity} = \frac{T^+}{T^+ + F^-} \qquad \text{specificity} = \frac{T^-}{T^- + F^+} \tag{4}$$

Table 3 also incorporates average runtimes for each algorithm. Note that there is little variation in all cases for restarting each algorithm with a different random seed. The SVM is by far the fastest algorithm because it only required 13 s of CPU time. The Bayesian algorithms exhibit linear increases between the standard and the T3 versions. For instance, Mixed-T3 is four times that of Mixed because it uses three extra hot chains. In cross-paradigm comparison, FFNN is comparable to Mixed and FFNNE to Mixed-T3. However, it is worth noting that Mixed chains can easily be run in a distributed fashion. In our experiments, a cluster of 8 modest performance units have been utilized to linearly reduce running times. The numbers reported here refer to total times.

Cross Validation on Three Data Sets. To validate the results on data set A, we further selected nine disjointed training and testing subsets and performed 10-fold cross validation (original selection being the first fold). Each training set contains 90% of the original data for training purposes and the remaining 10% is used for testing. In what follows, we present results for the SVM, FFNN, and Mixed algorithms. In Table 4, we present the results for all fold segments of data set A. We show means of true positives and true negatives (out of 58) along with their standard error. Finally, the last six rows show the sensitivity and specificity results derived from the mean true positive and negative values.

In terms of sensitivity, results for all predictors are comparable, with SVM doing marginally better. On average,

FFNN, 0.745; SVM, 0.781; Mixed, 0.751. In terms of specificity, Mixed performs consistently better (on average FFNN, 0.774; SVM, 0.771; Mixed, 0.824).

The positive likelihood ratio (LR-positive) incorporates sensitivity and specificity into one single metric providing an estimate of how well positives are predicted. In the application considered here the ability to predict active compounds (positives) is of paramount importance and LR-positive can be regarded as an overall indicator. It is given by (sensitivity)/(1 − specificity) and for the three predictors, FFNN, SVM and Mixed, is has the values of 3.30, 3.41, and 4.27 respectively.

Another important measure is stability of prediction. FFNN in most cases have much larger spread in the number of prediction within one segment. That is, they have larger standard error entries in Table 4. This reinforces the current view in the literature that FFNN performance is prone to starting conditions and local minima. Variation of prediction is not an issue for SVM because it is purely deterministic in the sense that it does not depend on any random choices.

In terms of computational effort, SVM is by far the most efficient algorithm; its training over all data sets/folds takes just a few seconds (see Table 3). As far as the other algorithms are concerned, FFNN takes 27 h (on a 2.13 GHz CPU with 4 GB RAM), Mixed takes 77% more, and FFNNE takes 117% more.

Finally, because the inactive molecules in data set A were randomly selected from a much larger pool, we repeated the 10-fold cross validation on two further selections of negatives (data sets B and C). Sensitivity and specificity for these are shown in Figure 4 and Figure 5. As it can be seen, in absolute terms both algorithms show results in the same region as that of data set A. When comparing the algorithms against each other, our conclusions from A are also valid for B and C. SVM, FFNN, and Mixed do roughly equally well in terms of sensitivity, but Mixed performs better in terms of specificity. In predicting active compounds as measured by LR-positive, Mixed also does better. For data set B, the LR-positive values for FFNN, SVM, and Mixed are 3.07, 3.07, and 3.77, while for data set C the corresponding values are: 2.96, 3.01, and 3.4.

Classifying Eduliss 2 Molecules. To identify possible leads for pyruvate kinase, we trained the Mixed algorithm on all 1164 examples of data set *A* and used the produced chains to classify molecules in a large database. As each classified molecule is also given a probability of activity, the whole database can be ranked for potential ligands.

The produced chains had 477,930 models of which 373,469 were distinct across chains. Of those, only trees visited over a threshold number of times are retained and merged into a single chain. In Figure 6 the distribution of models (x-axis) against the logarithm of times of appearance (y-axis) in the chain is shown. The trees are sorted in descending order of the logarithm. As it can be seen, the chains appear to move well in the space in that there are no models with large number of visits.

We ran classifications of the database for two values of a cutoff threshold U, 100 and 20. The former reduces the population of the chain to 15 032, whereas the latter reduces the chain to 134 431 trees. In what follows, we present results for $U = 20$, which in Figure 6 corresponds to removing all trees bellow the $y = \log(20) = 2.9957$ line. Even if included, these trees would have small individual contributions. It is worth noting that the most visited tree appeared in the chain 5040 times and its log-likelihood is $\log(P(C|T,M)) = -616.1$. This is the single model that fits the data best according to the scoring function.

The 3.7 million unique compounds of Eduliss 2 are classified according to the averaged prediction of all the models in the chain. Let E be the set of molecules in the database, T the set of classification trees with associated parameters Θ, C the set of classes, with τ and θ are members of the respective sets, then

$$P(C|E,T,\Theta) = \sum_{\tau \in T, \theta \in \Theta} P(\tau|M,C)P(C|E,\tau,\theta)$$

The probability of activity for a molecule in the database is thus dependent on the summation of contributions from all trees. Each consists of two parts: the importance of a tree (measured by its posterior, which is simply the times it appears in the chain over the length of the chain) and the classification of the molecule by that tree ($P(C|E,\tau,\theta)$).

When there are missing descriptors, we use an extra layer of averaging. In general, each tree that contains decisions based on missing descriptors can be used to produce K decisions. This is by means of choosing both branches of the tree when such a decision is encountered. The overall probability of activity is derived by summing all K decisions on activity and dividing by K.

The probability distribution over classes for each molecule can be regarded as an activity ranking. In Figure 7, six active molecules from our data set are presented side-by-side with the top six hits from the Eduliss 2 database. There is clear molecular similarity between the top hits and known active molecules, which without being conclusive evidence, is a promising sign. Both families consist of between two and four planar ring systems. The prediction of such related and similar looking molecules out of the database of 3.7 molecules suggests that the algorithm provides a self-consistent selection procedure with hit-molecules occupying the same region of chemical space as the known active compounds. It is worth noting that in virtual screening the objective is to enrich the chances of finding bioactive molecules by considering number of successes in large sets (enrichment factors).

Figure 8 shows the frequencies for each descriptor in our set. Each occurrence of the descriptor into a tree that appeared in the chain for more than $U > 20$ has been counted. thus multiple appearances on a single tree are accounted for. For instance the counts of some descriptors exceed the number of classifying trees used. In Figure 8, we can see a good mix for a large number of descriptors which is further evidence to the stochastic activity of the algorithm.

Figure 9 depicts a plot of number of molecules (y) against probability of activity (x). The latter has been sorted into discrete bins of width 0.01. The probability curve has a very regular well spread contour. The final high probability bin contains seven compounds with probabilities between 0.90 and 0.91. The top six compounds are shown in Figure 7.

DISCUSSION

This paper provides a new methodology that can be applied quite generally to analyze the results of large scale biological

and pharmacological screens and is therefore of broad interest and utility. By using machine learning techniques, we have shown how existing software, data sets, and descriptors can be combined and utilized for lead discovery.

This is the first time that Bayesian model averaging for classification trees is applied to biochemical descriptor classification. The Bayesian methodology allows further analysis of the results. Such analysis is based on the fact that a whole probability distribution over models is constructed and tree features can be averaged similar to the classification averaging we have already described. Such model features can include identification of contributors that are major contributors to best scoring molecules.

The results of the Bayesian approach have shown promise and were at least as good if not better to SVM and Neural Networks. LR-positive averages over data sets A, B, and C were FFNN, 3.09; SVM, 3.16; Mixed, 3.81. It has to be noted that SVM require far less computational time than any of the other methods. However, computational power is not the most common bottleneck in ligand discovery.

Bayesian model averaging over classification trees is an appealing framework for ligand discovery as in vitro screening results can be used to provide a simple model of bioactivity which in turn provides a convenient method for predicting and selecting sets of new compounds with a higher chance of having the required properties. There are four notable features of this approach. First, mixing chain generation methodologies can address the multidimensionality of the data. Second, molecules with missing descriptor values can still be classified and included in the analysis. Third, probabilities can readily be interpreted as a rank of ligand-binding activity. Finally, the MCMC experiments described here are inherently easy to run in a distributed environment and thus can be efficiently run on clusters of computers.

ACKNOWLEDGMENT

Thanks to Paul Taylor for helpful discussions, Kun-Yi Hsin for help with the data and access to EDULISS 2, and to James Cussens for discussions on priors. This work was in part supported by U.K.'s BBSRC project D00604 X/1. We would also like to thank the two anonymous referees for productive and helpful comments.

Supporting Information Available: Average figures for sensitivity, specificity, and likelihood ratios for all three data sets achieved by the three main algorithms and specificity and sensitivity histograms for data set A. This information is available free of charge via the Internet at http://pubs.acs.org/.

REFERENCES AND NOTES

(1) National Center for Biotechnology Information. http://www.ncbi.nlm.nih.gov/ (accessed April, 2009).
(2) Hansch, C.; Hoekman, D.; Gao, H. Comparative QSAR: Toward a Deeper Understanding of Chemicobiological Interactions. *Chem. Rev.* **1996**, *96*, 1045–1076.
(3) Lipinski, C. A.; Lombardo, F.; Dominy, B. W.; Feeney, P. J. Experimental and Computational Approaches to Estimate Solubility and Permeability in Drug Discovery and Development Settings. *Adv. Drug Delivery Rev.* **2001**, *46*, 3–26.
(4) Todeschini, R.; Lasagni, M.; Marengo, E. New Molecular Descriptors for 2D and 3D Structures. Theory. *J. Chemom.* **1994**, *8*, 263–272.
(5) Tonga, J.; Liu, S.; Zhou, P.; Wu, B.; Li, Z. A Novel Descriptor of Amino Acids and Its Application in Peptide QSAR. *J. Theor. Biol.* **2008**, *253*, 90–97.
(6) Tabarakia, R.; Khayamiana, T.; Ensafia, A. Wavelet Neural Network Modeling in QSPR for Prediction of Solubility of 25 Anthraquinone Dyes at Different Temperatures and Pressures in Supercritical Carbon Dioxide. *J. Mol. Graph. Model.* **2006**, *25*, 46–54.
(7) Todeschini, R.; Gramatica, P.; Navas, N. 3D-Modeling and Predication by WHIM Descriptors. Part 9. Chromatographic Relative Retentation Time and Physicochemical Properties of Polychlorinated Biphenyls (PCBs). *Chemom. Intell. Lab. Syst.* **1998**, *40*, 53–63.
(8) Byvatov, E.; Fechner, U.; Sadowski, J.; Schneider, G. Comparison of Support Vector Machine and Artificial Neural Network Systems for Drug/Nondrug Classification. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 1882–1889.
(9) DRAGON 5.4. http://www.talete.mi.it/ (accessed 17 November 2008).
(10) Inglese, J.; Auld, D. S.; Jadhav, A.; Johnson, R. L.; Simeonov, A.; Yasgar, A.; Zheng, W.; Austin, C. P. Quantitative High-Throughput Screening: A Titration-Based Approach That Efficiently Identifies Biological Activities in Large Chemical Libraries. *Proc. Natl. Acad. Sci. U. S. A.* **2006**, *103*, 11473–11478.
(11) Nowicki, M. W.; Tulloch, L. B.; Worralll, L.; McNae, I. W.; Hannaert, V.; Michels, P. A.; Fothergill-Gilmore, L. A.; Walkinshaw, M. D.; Turner, N. J. Design, Synthesis and Trypanocidal Activity of Lead Compounds Based on Inhibitors of Parasite Glycolysis. *Bioorg. Med. Chem.* **2008**, *16*, 5050–5061.
(12) Christofk, H. R.; Vander Heiden, M. G.; Harris, M. H.; Ramanathan, A.; Gerszten, R. E.; Wei, R.; Fleming, M. D.; Schreiber, S. L.; Cantley, L. C. The M2 Splice Isoform of Pyruvate Kinase Is Important for Cancer Metabolism and Tumour Growth. *Nature* **2008**, *452*, 230–233.
(13) Chan, M.; Tan, D. S.; Sim, T. S. *Plasmodium falciparum* Pyruvate Kinase As a Novel Target for Antimalarial Drug-Screening. *Travel Med Infect Dis.* **2007**, *5*, 125–131.
(14) Breiman, L.; H., F. J.; Olshen, R. A.; Stone, C. J. *Classification and Regression Trees*; Wadsworth International: Belmond, CA, 1984.
(15) Ripley, B. D. *Pattern Recognition and Neural Networks*; Cambridge University Press: Cambridge, U.K., 1996.
(16) NIH Chemical Genomics Center. http://ncgc.nih.gov/db/ (accessed October 2008).
(17) Hsin, K.-Y. Eduliss 2, Edinburgh University Ligand Selection System. http://eduliss.bch.ed.ac.uk/ (accessed April 2009).
(18) Hoeting, J. A.; Madigan, D.; Raftery, A. E.; Volinsky, C. T. Bayesian Model Averaging: A Tutorial. *Stat. Sci*. **1999**, *14*, 382–401.
(19) Chipman, H.; George, E.; McCulloch, R. Bayesian CART Model Search (With Discussion). *J. Am. Stat. Assoc.* **1998**, *93*, 935–960.
(20) Denison, D. G. T.; Holmes, C. C.; Mallick, B. K.; Smith, A. F. M. *Bayesian Methods for Nonlinear Classification and Regression*; Wiley, Chichester, U.K., 2002.
(21) Angelopoulos, N.; Cussens, J. Tempering for Bayesian C&RT. In *Machine Learning*; Proceeding of the 22nd International Conference, Bonn, Germany, ACM: New York, NY, U.S.A., 2005; pp 17−24.
(22) Quinlan, J. R. Induction of Decision Trees. *MachineLearn.* **1986**, *1*, 81–106.
(23) Buntine, W. L. Learning Classification Trees. *Stat. Comp.* **1992**, *2*, 63–73.
(24) Angelopoulos, N.; Cussens, J. Exploiting Informative Priors for Bayesian Classification and Regression Trees. In *Artificial Intelligence*; Proceedings of the 19th International Joint Conference, Edinburgh, U.K., IJCAI Press: Rochester, U.S.A., 2005; pp 641−646.
(25) Cortes, C.; Vapnik, V. Support-Vector Networks. *Machine Learn.* **1995**, *20*, 273–297.
(26) Burges, C. J. C. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discovery* **1998**, *2*, 121–167.
(27) Meyer, D.; Leisch, F.; Hornik, K. The Support Vector Machine under Test. *Neurocomputing* **2003**, *55*, 169–186.
(28) Joachims, T. In *Advances in Kernel Methods: Support Vector Machines*; Schölkopf, B., Burges, C. J., Smola, A. J., Eds.; MIT Press: Cambridge, MA, 1998; Chapter 11, pp 169−184.
(29) Cybenko, G. Approximation by Superpositions of a Sigmoidal Function. *Math. Control Signals Syst.* **1989**, *2*, 303–314.
(30) Bishop, C. *Neural Networks for Pattern Recognition*; Clarendon Press: Oxford, U.K., 1995.
(31) Hadjiprocopis, A.; Smith, P. Feed Forward Neural Network Entities. In *Biological and Artificial Computation: From Neuroscience to Technology*; Mira, J., Moreno-Diaz, R., Cabestany, J., Eds.; Lecture Notes in Computer Science; Springer-Verlag: Spain, Berlin/Heidelberg, Germany, 1997; Vol. 1240, pp 349−359.
(32) Altman, D.; Bland, J. Diagnostic Tests 1: Sensitivity and Specificity. *Brit. Med. J.* **1994**, *308*, 1552.

CI900046U