Bridging the Gap Between Neural Network and Kernel Methods: Applications to Drug Discovery

Pierre BALDI [a,b,1], Chloe AZENCOTT [a,b] and S. Joshua SWAMIDASS [a,b,2]
[a] *Department of Computer Science, University of California, Irvine*
[b] *Institute for Genomics and Bioinformatics, University of California, Irvine*

Abstract. We develop a hybrid machine learning architecture, the Influence Relevance Voter (IRV), where an initial geometry- or kernel- based step is followed by a feature-based step to derive the final prediction. While other implementations of the general idea are possible, we use a k-Nearest-Neighbor approach to implement the first step, and a Neural Network approach to implement the second step for a classification problem. In this version of the IRV, the rank and similarities of the k nearest neighbors of an input are used to compute their individual relevances. Relevances are combined multiplicatively with the class membership values to produce influences. Finally the influences of all the neighbors are aggregated to produce the final probabilistic prediction. IRVs have several advantages: they can be trained fast, they are easily interpretable and modifiable, and they are not prone to overfitting since they rely on extensive weight sharing across neighbors. The IRV approach is applied to the problem of predicting whether a given compound is active or not with respect to a particular biochemical assay in drug discovery and shown to perform well in comparison to other predictors. In addition, we also introduce and demonstrate a new approach, the Concentrated ROC (CROC), for assessing prediction performance in situations ranging from drug discovery to information retrieval, where ROC curves are not adequate, because only a very small subset of the top ranked positives is practically useful. The CROC approach uses a change of coordinates to smoothly magnify the relevant portion of the ROC curve.

Keywords. Neural Netowrks, Kernels, Chemoinformatics, Drugs

Introduction

A reasonable distinction often made in machine learning, for instance in supervised classification and regression problems, is the distinction between "feature-based" and "geometry-based" methods. Feature-based methods, as exemplified by neural networks (NN), tend to derive a prediction from numerical features extracted from the raw input data. On the other hand, geometry-based methods, as exemplified by kernel or k-nearest neighbor (kNN) methods, derive a prediction by looking at the similarities between the

[1]Corresponding Author: Department of Computer Science, University of California Irvine, Irvine, CA 92697-3435, USA; E-mail: pfbaldi@ics.uci.edu

[2]Current address: Department of Pathology and Immunology, Washington University, St. Louis, MO.

current input instance and all the other instances available in the training set, hence looking at the geometry of the neighborhood of the input instance.

While somewhat useful, this distinction is far from absolute since, for instance, the pairwise similarities of an input to the other training instances could also be considered as "features". Here, we introduce a hybrid approach, the Influence Relevance Voter (IRV) that further develops a bridge between feature- and geometry-based approaches, and can be viewed as a two-steps process where an initial geometry-based step is followed by a feature-based step to derive the final prediction. While other implementations of the general idea are possible, here we use a kNN approach to implement the first step, and a NN approach to implement the second step. Thus, we use a neural network to derive a prediction from a description of the neighborhood associated with the k nearest neighbors and by aggregating the influences exerted by each neighbor. The proposed hybrid approach is demonstrated on one of the most fundamental problems in chemoinformatics and drug discovery aimed at predicting whether a given compound is active or not with respect to a particular biochemical assay.

In addition to the hybrid prediction methods, we also propose a novel metric for analyzing classification performance results in situations where only a small subset of the top ranked positives is practically useful. This situation occurs frequently in problems ranging from drug discovery to information retrieval, where typically only a few dozens of compounds or documents can be tested in the laboratory or inspected manually. In these cases, common metrics such as the ROC curve or the AUC are somewhat meaningless. We propose a new approach, the Concentrated ROC (CROC), which concentrates the analysis on the relevant portion of the ROC curve by using a smooth coordinate transformation.

1. Chemical Data, Representations, and Similarity Measures

Chemical Data. The IRV is illustrated here on the standard problem of virtual High-Throughput Screening (vHTS) in drug discovery. Specifically, we use the benchmark dataset associated wtih the Drug Therapeutics Program (DTP) AIDS Antiviral Screen made available by the National Cancer Institute (NCI) [1], which we refer to as the HIV data set. The set contains assay results for 42,678 chemicals experimentally tested for their ability to inhibit the replication of the Human Immunodeficiency Virus (HIV) *in vitro*. Initially, the data is divided into three classes, "inactive", "active", and "moderately active". In what follows, we combine the "active" and "moderately active" classes into a single "active" class containing about 3.5% of the data. Thus the final dataset contains 1,503 active compounds and 41,175 inactive compounds.

Chemical Representations. Chemical compounds are described by their labeled molecular graph, where the vertices correspond to the atoms of the molecule, labeled by the atom type (e.g. Carbon (C), Nitrogen (N), Oxygen (O)), and the edges correspond to the bonds connecting these atoms together, labeled by the bond type (e.g. single, double). In chemoinformatics, these variable-size molecular graphs are routinely converted into long, fixed-size, binary fingerprint vectors. Each bit in a fingerprint representation corresponds to the presence or absence of a particular feature in the molecular graph. Typical sets of features used in the literature [2,3,4] are labeled paths up to a certain length, or labeled trees up to a certain depth. While the methods to be presented can be applied with any set of features, here we use circular substructures [3], corresponding to labeled trees of depth up to 2. Each vertex is labeled by the atomic symbol of the corresponding atom together with the number of non-hydrogen atoms to which it is bonded (e.g., C3 for a carbon with three non-hydrogen atoms attached), and each bond is labeled according to its bond type (single, double, triple, or aromatic). We choose this particular labeling scheme and set of features because the resulting representations seem to yield the best performance based on experiments performed using other datasets (not reported).

Chemical Similarity Measures. Although there are several possible ways of defining similarity between fingerprint vectors [5], by far the most widely used approach is the well known Jaccard-Tanimoto similarity measure [2]. If $\vec{A}$ and $\vec{B}$ are the fingerprint vectors representing molecules $\mathcal{A}$ and $\mathcal{B}$, the Jaccard-Tanimoto similarity between $\mathcal{A}$ and $\mathcal{B}$ is defined by:

$$S(\mathcal{A},\mathcal{B}) = S(\vec{A},\vec{B}) = \frac{A \cap B}{A \cup B} \tag{1}$$

where $A \cap B$ is the number of 1-bits in the intersection $\vec{A}$ AND $\vec{B}$, and $A \cup B$ is the number of 1-bits that appear in the union $\vec{A}$ OR $\vec{B}$. It is well known that the Jaccard-Tanimoto similarity measure satisfies the Mercer's conditions and, hence, yields a Mercer kernel [6].

2. The Influence Relevance Voter

2.1. The IRV Architecture

Given a parameter k, the IRV can be viewed as a compact NN architecture which produces a final probabilistic classification using information extracted from the k nearest neighbors in the training set. The prediction is computed from the "influence" of each neighbor, which in turn is computed as the product of each neighbor's "relevance," encoding how much each neighbor should affect the prediction, and "vote," encoding the direction towards which the prediction should be pushed. Figure 1 shows the architecture of the IRV, using a notation similar to the plate notation used for graphical models, which displays the local dependencies that are replicated throughout the network, illustrating its structure and exposing the extensive weight-sharing across the network. Formally, the IRV's output is given by

$$z(\mathcal{X}) = \sigma\left(w_z + \sum_{i=1}^{k} I_i\right) \tag{2}$$

where $\mathcal{X}$ is the test molecule, i ranges from 1 to k over all the k nearest neighbors, I_i is the "influence" of the ith neighbor on the output, w_z is the bias of the output node, and $\sigma(\cdot)$ is the logistic function $1/(1+e^{-x})$. These influences indicate exactly how much, and in which direction, each training example contributes to the prediction and can be used to interpret each final prediction. The influence of the ith node is defined multiplicatively by

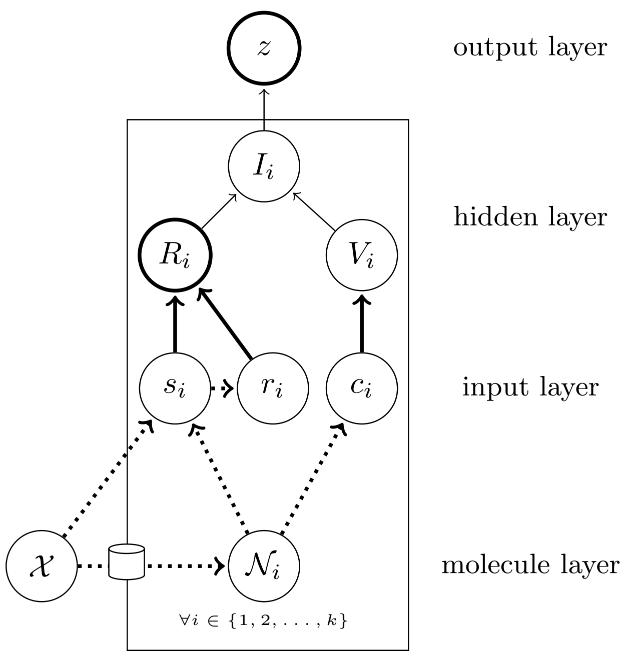

Figure 1. IRV architecture.

$$I_i = R_i V_i, \tag{3}$$

where R_i is the relevance and V_i is the vote of the ith neighbor. For this study, the relevance is defined as

$$R_i = \sigma\left(w_y + w_s s_i + w_r r_i\right) \tag{4}$$

where s_i is the similarity $S(\mathcal{X}, \mathcal{N}_i)$ of the ith closest neighbor $\mathcal{N}_i$ to the test compound $\mathcal{X}$, r_i is the rank of the ith neighbor in the similarity-sorted list of neighbors, w_s and w_r are parameters tuning the importance of different inputs, and w_y is the bias of the logistic unit. In order to facilitate learning, all inputs are normalized to standard units during the preprocessing step, by subtracting the corresponding mean and dividing by the corresponding standard deviation. For this study, we define the vote by

$$V_i = \begin{cases} w_0 \text{ if } c_i = 0 \\ w_1 \text{ if } c_i = 1 \end{cases} \tag{5}$$

where w_0 is the weight associated with inactive neighbors, w_1 is the weight associated with active neighbors, and $c_i \in \{0, 1\}$ is the class of the ith neighbor. As usual, the logistic output of the IRV can be interpreted as the probability

$$z(\mathcal{X}) \approx P(\mathcal{X} \text{ is active} | \mathcal{X}\text{'s structure, training data}) \tag{6}$$

and the IRV can be trained to minimize, for instance by gradient descent, the relative entropy (or negative log-likelihood) between the true target distribution $t(\mathcal{T})$ and the predicted distribution $z(\mathcal{T})$ across all molecules $\mathcal{T}$ in the training set

$$-\sum_{\mathcal{T}} t(\mathcal{T}) \log z(\mathcal{T}) + (1 - t(\mathcal{T})) \log(1 - z(\mathcal{T})) \tag{7}$$

It is possible to add weight decay terms to the error function, equivalent to defining a gaussian prior on the weights, to prevent over-fitting during training. In practice, overfitting is unlikely because so few parameters are optimized during training. Three parameters, w_s, w_r, and w_y, are shared across all k neighbors. The model requires only three additional parameters, w_1, w_0, and w_z. Therefore, a total of only six parameters are learnt from the data. Nevertheless, weight decay is included in this implementation because it seems to further decrease the training time.

Each influence, I_i, encodes how much, and in what direction, its associated neighbor pushes the prediction. Influences with the greatest absolute values affect the output the most, their associated neighbors are the data upon which the prediction is made. Those with negative values push the output towards an inactive prediction, while those with positive values push the output towards an active prediction. Thus the influences and the prediction are readily understandable by direct inspection and visualization.

2.2. Variations on the IRV

The IRV's performance can be fine-tuned by several variations. Here, we consider: (1) the number of neighbors selected; (2) the class-balance of the neighbors; and (3) the weighting of different classes during training. We have experimented with different values of k. The network's performance is not sensitive to the exact value of k above a certain threshold, but sometimes small performance improvements can be realized. For conciseness, here we report the results obtained with $k = 20$ which allows us to obtain good results with low computational cost. It is also possible to apply hyperparameter optimization methods to try to optimize k. Rather than presenting the IRV network with nearest neighbors irrespective of their class, it is possible instead to present an equal number of neighbors from each class. For example, instead of selecting the 20 nearest neighbors, one can select both the 10 nearest active and the 10 nearest inactive compounds. For the HIV data, results are reported using the 20 nearest neighbors, irrespective of their class, since this appeared to work slightly better. During training, it is also possible to over-weight the log-likelihood of one of the classes, so as to increase its contribution to the training process. This can be used to address the class imbalance in most vHTS datasets. For example, if there are 20 times more inactive chemicals than active ones, then the error associated with each of the active compounds can be multiplied by 20 during training to restore the balance. Preliminary experiments indicate that this variation does not appreciably improve performance on the HIV dataset, so it is excluded from the results. Finally, there are many possible architectural variations on the IRV, such as adding hidden layers in the NN, or using more complex representations of the neighborhood. These are left for future work.

2.3. *Other vHTS Algorithms*

In the simulations, we compare the IRV to several other classification methods. One of the simplest methods for vHTS is the maximum similarity algorithm (MAX-SIM) [7, 8], in which each test molecule is scored by its highest similarity to a known active compound. Formally, if all known active compounds are denoted by $\{\mathcal{X}_1, \ldots, \mathcal{X}_p\}$, the score of a new compound $\mathcal{X}$ is given by

$$z(\mathcal{X}) = \max_{i=1}^{p} S(\mathcal{X}, \mathcal{X}_i) \tag{8}$$

MAX-SIM has been well studied [7,8] and is straightforward to implement. However, it is a rather unrefined method, which takes very little information into account and does not make use at all of known inactive compounds. Therefore, we consider it as a baseline approach, and expect more elaborate methods to perform better. A second obvious method to be used in the comparison is k-Nearest-Neighbor, where we score a compound using the fraction of its k-nearest-neighbors that are active. Finally, as the Tanimoto metric satisfies Mercer's condition [6], we can use it as a kernel in a standard SVM algorithm (see also [9] for other examples of kernels applied to VHTs problems). Given the large size of typical vHTS datasets, we use the SVMTorch online SVM implementation [10] to reduce both memory requirements and computational time. SVM predictors have been reported to have good accuracy in vHTS experiments, although their output does not always come with a probabilistic interpretation [11] and the underlying inferences can remain opaque, as the most representative examples, which are typically far from the decision boundary, are not part of the support vectors. This can be significant in vHTS problems, where ranking active compounds ahead of inactive ones is more important than correctly learning a separation boundary between the two classes, and understanding why a chemical is predicted to be active can be as important as knowing it is active.

3. Quantifying Early Recognition: a CROC on ROC

Although the vHTS problem is often cast as a classification problem, and prediction performance is analyzed using standard ROC curves and AUC metrics, one can argue that it is not really a classification problem and ROC/AUC performance is not really meaningful. Often, for practical reasons, only a small subset of compounds at the top of the predicted list of active compounds can be tested in the laboratory. Therefore most of the ROC curve is of little interest. Casting the problem as a ranking problem does not resolve the issue since again the precise ranking of the compounds, especially towards the $x = 1$ side of the ROC curve, is completely irrelevant. What seems to really matters is the notion of "early enrichment", "early recognition", or "early retrieval", i.e having as many true positive as possible at the beginning of the ROC curve. This issue is not particular to vHTS but is found in many other applications, from fraud detection to information retrieval. Thus a second contribution of this work is the development of a new metric to capture and measure this notion of early enrichment or early recognition.

Several strategies can be envisioned to derive performance measures that can address the early enrichment problem. A possible approach is to look only at the early portion of

	Top 100	Top 1,503
RANDOM	3	53
MAX-SIM	58	590
kNN	*94*	731
SVM	90	*750*
IRV	**97**	**764**

Table 1. Number of active hits retrieved among the top 100 and 1,503 compounds of the prediction-sorted list for various vHTS algorithms on the HIV data (which contains 1,503 actives). Best performances are in **bold** and second-best performances are in *italics*.

the ROC curve, and the corresponding underlying area, between 0 and some threshold t (e.g. $t = 0.05$) or at the derivative of the ROC curve very near the origin. While reasonable, such approaches are not entirely satisfying since they lack smoothness or require using an arbitrary cutoff that may not scale well with the problem at hand. Thus here we propose a general strategy, which we call Concentrated ROC, or CROC, which basically consists in magnifying the portion of the ROC curve one is interested in, using a smooth continuous function to transform the x-axis. To magnify the early portion of the ROC, one needs to use a continuous function $f : [0, 1] \rightarrow [0, 1]$ which is monotonically increasing (hence one to one), convex down (negative second derivative), and satisfies $f(0) = 0$ and $f(1) = 1$. Examples of simple functions with these properties include: $f(x) = x^{1/(1+\alpha)}$, $f(x) = [\log_2(x + 1)]^{1/(1+\alpha)}$, and $f(x) = (1 - e^{-\alpha x})/(1 - e^{-\alpha})$ with $\alpha > 0$ being the magnification factor. The choice of function can be determined empirically with some experimentation, or more theoretically if one knows how "interest" in the top hits decays with their rank (as measured, for instance, in terms of number of clicks for web page retrieval). For these theoretical reasons and some experimentation, here we choose the latter exponential reparameterization of the x axis to derive CROC curves and the corresponding area under the CROC, the AUC-CROC. The CROC curve is derived by plotting the ROC curve once the x axis has been reparameterized by the function f.

4. Results

Here we present the cross-validated performance of a random classifier, denoted by RANDOM, the MAX-SIM algorithm, the k-Nearest-Neighbor, the SVM, and the IRV on the HIV dataset. Since the SVM cannot be leave-one-out cross-validated in a reasonable amount of time on such a large dataset, we restrict ourselves to 10-fold cross-validation experiments. However, a full leave-one-out cross-validation of the IRV can easily be carried out even on datasets made of several hundred thousands of compounds.

Typically, experimentalists in laboratories are willing to further test on the order of 10-1,000 compounds. Table 1 presents the number of active hits retrieved among the top 100 compounds of the prediction-sorted list. Since the data contains 1,503 active chemicals, the table also reports the number of active hits retrieved among the top 1,503 compounds of the prediction-sorted list. The IRV performs slightly better than the SVM, retrieving 7 more hits among the top 100. Moreover, it is trained faster (in the order of a few minutes versus a few hours per fold for the SVM).

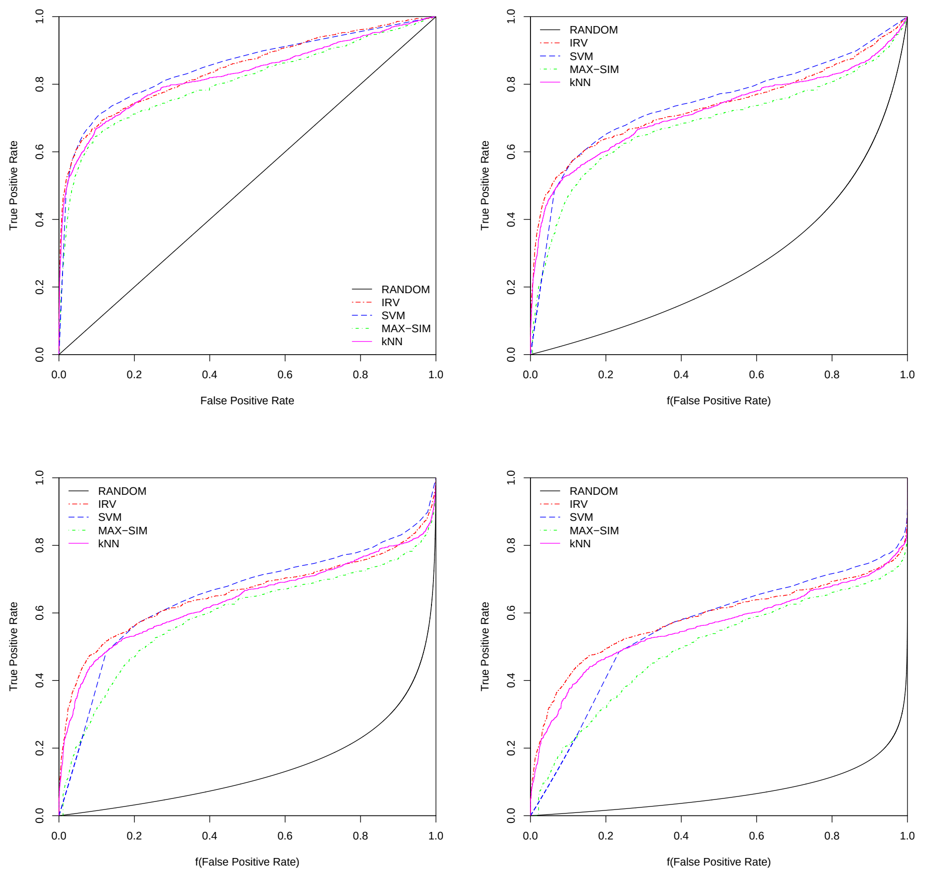

Figure 2. 10-fold cross-validated performance of the MAX-SIM, kNN, SVM, and IRV algorithms on the HIV data. In clockwise order: (a) standard ROC curves; (b) CROC curves derived by reparameterizing the x axis using $\alpha = 3.3$ ($f(0.2) = 0.5$); (c) CROC curves derived by reparameterizing the x axis using $\alpha = 7$ ($f(0.1) = 0.5$); and (d) CROC curves derived by reparameterizing the x axis using $\alpha = 14$ ($f(0.05) = 0.5$).

Table 2 presents the cross-validated area under the ROC and CROC curves computed for $\alpha = 3.3$, $\alpha = 7$, and $\alpha = 14$ for the same experiments. These three values of α correspond to three exponential transformations f that respectively send the points $x = 0.2$, $x = 0.1$, and $x = 0.05$ onto the midpoint $x = 0.5$. The corresponding curves are depicted in Figure 2. Whereas it is difficult to evaluate on the ROC curve (Figure 2a) which one of the methods performs better at low cutoffs, a noticeable difference becomes obvious on the CROC curves (Figures 2b,2c, and 2d). As the magnification factor α is increased, ambiguities in the early part of the ROC curve are clearly resolved.

The area under the ROC curve for a random classifier is 0.5. The area under the corresponding CROC curve depends on the value of the magnification factor α (Figure 2). [A simple calculation shows that the AUC-CROC(RANDOM) is given by $\frac{1}{\alpha} - \frac{e^{-\alpha}}{1-e^{-\alpha}}$.] It is therefore not as straightforward to compare the performance of a classifier to random in the CROC framework. To address this issue, we suggest to use the normalized

	AUC-ROC	AUC-CROC ($\alpha = 3.3$)	AUC-CROC ($\alpha = 7$)	AUC-CROC ($\alpha = 14$)
Random	0.500	0.265	0.142	0.071
MAX-SIM	0.806	0.683	0.592	0.497
kNN	0.828	0.718	0.638	*0.559*
SVM	**0.852**	*0.723*	*0.644*	0.557
IRV	*0.845*	**0.734**	**0.656**	**0.584**

Table 2. Area under the ROC and CROC curves, for various vHTS algorithms 10-fold cross-validated on the HIV data. Best performances are in **bold** and second-best performances are in *italics*.

	nAUC-ROC	nAUC-CROC ($\alpha = 3.3$)	nAUC-CROC ($\alpha = 7$)	nAUC-CROC ($\alpha = 14$)
MAX-SIM	0.614	0.569	0.524	0.459
kNN	0.655	0.617	0.578	*0.525*
SVM	**0.704**	*0.623*	*0.585*	0.523
IRV	*0.690*	**0.638**	**0.599**	**0.552**

Table 3. Difference between the area under the ROC (resp. CROC) curve and the area under the corresponding random ROC (resp. CROC) curve, normalized to $[0, 1]$, for various vHTS algorithms 10-fold cross-validated on the HIV data. Best performances are in **bold** and second-best performances are in *italics*.

AUC-CROC (nAUC-CROC) metric, defined by nAUC-CROC=[(AUC-CROC)-(AUC-CROC(random))]/[1-(AUC-CROC(random))]. Intuitively, this metric is defined as the area under the CROC curve scaled so that the performance of a random classifier corresponds to 0 and the performance of a perfect classifier corresponds to 1.

Table 3 presents nAUC-CROC values. The area under the ROC curve of the SVM classifier is slightly larger than the area under the ROC curve of the IRV. However, the IRV performs slightly better than the SVM in terms of area under the CROC curve, in agreement with the enrichments presented in Table 1, suggesting that the IRV is indeed more suited to early retrieval than the SVM, whereas the SVM is better at globally separating the active class from the inactive one, for this problem. Finally, Figure 3 demonstrates how IRV outputs can be interpreted by visual inspection of the neighborhood influences.

5. Conclusion

We have presented the IRV approach for classification or regression problems which combines feature-based and geometry-based methods and uses a NN to refine a kNN predictor by postprocessing features extracted from the neighborhood. We have also introduced the CROC metric to measure early enrichment. We have demonstrated that the IRV and the CROC give good results on a challenging drug discovery data set. Additional experiments are in progress to further validate these results on other problems and data sets. While the CROC approach has been used here to magnify the x axis, it can also be used to magnify both axes simultaneously. Other directions for possible future work include: (1) the exploration of IRV architectural variations (e.g additional hidden layers); (2) automatic setting of k or α; and (3) use the AUC-CROC or nAUC-CROC as an objective function to be maximized during learning, in order to maximize early enrichment.

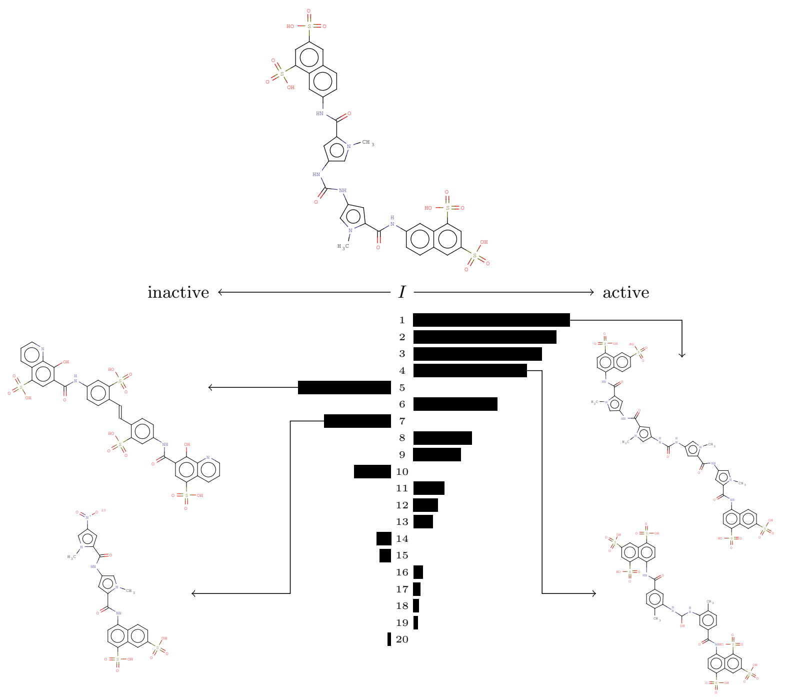

Figure 3. The 20 influences on an accurately predicted hit from the HIV dataset, obtained in the cross-validation experiment, are displayed as a bar graph. The experimental data both supporting and countering the prediction is readily apparent. The structures of four neighbors, two actives and two inactives, are shown. Compounds on the right are structurally similar and active. Compounds on the left are structurally similar and inactive.

Acknowledgments: Work supported by NIH Biomedical Informatics Training grant (LM-07443-01), NSF MRI grant (EIA- 0321390), NSF grant 0513376, and a Microsoft Research Award to PB. We would like also to acknowledge the OpenBabel project and OpenEye Scientific Software for their free software academic licenses.

References

[1] http://dtp.nci.nih.gov/docs/aids/aids_data.html. Last accessed May 2009.

[2] A. R. Leach and V. J. Gillet. *An Introduction to Chemoinformatics*. Springer, Dordrecht, The Netherlands, 2005.

[3] M. Hassan, R. D. Brown, S. Varma-O'Brien, and D. Rogers. Cheminformatics Analysis and Learning in a Data Pipelining Environment. *Molecular Diversity*, 10:283–299, 2006.

[4] S. J. Swamidass and P. Baldi. Bounds and algorithms for exact searches of chemical fingerprints in linear and sub-linear time. *Journal of Chemical Information and Modeling*, 47(2):302–317, 2007.

[5] J. D. Holliday, C. Y. Hu, and P. Willett. Grouping of Coefficients for the Calculation of Inter-Molecular Similarity and Dissimilarity Using 2D Fragment Bit-Strings. *Comb. Chem. High Throughput Screen.*, 5(2):155–66, 2002.

[6] S. J. Swamidass, J. H. Chen, J. Bruand, P. Phung, L. Ralaivola, and P. Baldi. Kernels for Small Molecules and the Predicition of Mutagenicity, Toxicity, and Anti-Cancer Activity. *Bioinformatics*, 21(Supplement 1):59, 2005. Proceedings of the 2005 ISMB Conference.

[7] Jérôme Hert, Peter Willett, David J. Wilton, Pierre Acklin, Kamal Azzaoui, Edgar Jacoby, and Ansgar Schuffenhauer. Comparison of Fingerprint-Based Methods for Virtual Screening Using Multiple Bioactive Reference Structures. *Journal of Chemical Information and Modeling*, 44(3):1177–1185, 2004.

[8] Jérôme Hert, Peter Willett, David J. Wilton, Pierre Acklin, Kamal Azzaoui, Edgar Jacoby, and Ansgar Schuffenhauer. Enhancing the Effectiveness of Similarity-Based Virtual Screening Using Nearest-Neighbor Information. *Journal of Medicinal Chemistry*, 48(3):7049–54, 2005.

[9] P. Mahé, L. Ralaivola, V. Stoven, and J. P. Vert. The pharmacophore kernel for virtual screening with support vector machines. *Journal of Chemical Information and Modeling*, 46:2003–2014, 2006.

[10] R. Collobert and S. Bengio. SVMTorch: Support Vector Machines for Large-Scale Regression Problems. *Journal of Machine Learning Research*, 1, Sep. 2001 2001.

[11] Michael E. Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.