BIOINFORMATICS

Vol. 20 no. 11 2004, pages 1702–1709
doi:10.1093/bioinformatics/bth144

Why neural networks should not be used for HIV-1 protease cleavage site prediction

Thorsteinn Rögnvaldsson and Liwen You*

Intelligent Systems Laboratory, School of Information Science, Computer and Electrical Engineering, Halmstad University, Box 823, 301 18 Sweden

Received on August 25, 2003; revised on December 26, 2003, accepted on January 19, 2004
Advance Access publication February 26, 2004

ABSTRACT

Summary: Several papers have been published where nonlinear machine learning algorithms, e.g. artificial neural networks, support vector machines and decision trees, have been used to model the specificity of the HIV-1 protease and extract specificity rules. We show that the dataset used in these studies is linearly separable and that it is a misuse of nonlinear classifiers to apply them to this problem. The best solution on this dataset is achieved using a linear classifier like the simple perceptron or the linear support vector machine, and it is straightforward to extract rules from these linear models. We identify key residues in peptides that are efficiently cleaved by the HIV-1 protease and list the most prominent rules, relating them to experimental results for the HIV-1 protease.

Motivation: Understanding HIV-1 protease specificity is important when designing HIV inhibitors and several different machine learning algorithms have been applied to the problem. However, little progress has been made in understanding the specificity because nonlinear and overly complex models have been used.

Results: We show that the problem is much easier than what has previously been reported and that linear classifiers like the simple perceptron or linear support vector machines are at least as good predictors as nonlinear algorithms. We also show how sets of specificity rules can be generated from the resulting linear classifiers.

Availability: The datasets used are available at http://www.hh.se/staff/bioinf/

Contact: denni@ide.hh.se

1 INTRODUCTION

Machine learning algorithms like artificial neural networks (ANNs) and support vector machines (SVMs) have, over the last decade, become popular tools for data mining protein databases and predicting protein properties. The success of ANN algorithms on, e.g. predicting secondary structure, signal peptides and peptidase cleavage (Baldi and Brunak, 2001) from a primary sequence have inspired many to apply ANN algorithms to other problems within this field. This has, unfortunately, not always been done in a good principled way and datasets that were clearly unsuitable for this type of exercise have been used. This has resulted in overly complicated models that have been unnecessarily hard to interpret. We demonstrate this on the problem of modeling HIV-1 protease clevage site specificity, which is an example of a problem poorly suited for a nonlinear approach, but nevertheless several ANN and SVM models have been presented.

Understanding HIV-1 protease cleavage site specificity is very desirable, because efficiently cleaved substrates are also excellent templates for the synthesis of tightly binding chemically modified inhibitors (Beck *et al*., 2000). The HIV-1 protease cleavage specificity is, however, a challenging problem because of the high context sensitivity and broadness (i.e. the fact that inhibitors are not able to prevent the cleavage completely). No perfect rule is yet known that determines if and where a peptide will be cleaved by the HIV-1 protease, although many studies have been done on this subject.

Thompson *et al*. (1995) were the first to apply an ANN to the HIV-1 protease cleavage specificity problem. They used a standard feed-forward multilayer perceptron (MLP), achieving a classification accuracy of ∼88% on a test set with 39 out-of-sample peptides. Cai and Chou (1998) later repeated the work of Thompson *et al*. (1995) using an expanded dataset and a standard MLP with eight hidden units. They reported a classification accuracy of 92% correct on a test set with 63 peptides, concluding that the MLP was 'superior…in dealing with nonlinear problems such as predicting HIV protease cleavage sites' (Cai and Chou, 1998). Narayanan *et al*. (2002) repeated the experiment using the same data and the same MLP architecture as Cai and Chou (1998) but trying different splits of training and test sets; they achieved the same classification accuracy as Cai and Chou. Naraynan *et al.* (2002) also tried using a decision tree (with poor results) in order to extract rules for the HIV-1 protease cleavage. The decision tree was never able to predict the cleavage as well as the neural network. Cai *et al*. (2002) recently applied SVMs to the problem, trying several different kernel types (linear, polynomial and Gaussian), concluding that a Gaussian kernel was best but that the resulting predictor was inferior to the MLP they presented

*To whom correspondence should be addressed.

in 1998. The results with Gaussian kernels reported by Cai *et al*. (2002) are not significantly better than the results with linear kernels. Yet, Cai *et al.* stated that 'the SVM has grasped the complicated relationship between the oligopeptides and cleavability' and that the SVM has a 'strong ability in dealing with nonlinear problems such as predicting HIV protease cleavage sites' (Cai *et al*., 2002).

In this paper, we show that (contrary to what has been stated earlier) the HIV-1 protease cleavage site specificity problem is a linear problem that can be solved with a simple linear model and that specificity rules can be generated from this. We also discuss why this may not be so surprising for a problem/dataset of this type. The lack of evidence that the problem is nonlinear may mean that protease specificity problems in general are not likely to be nonlinear and we discuss why this may be so, or it may be a result of the fact that the dataset is not very large (a general problem in this domain). From an Occam's razor point of view, however, it is wiser to use new data to try to disprove the linear rules rather than to prove nonlinear rules.

2 SYSTEM AND METHODS

2.1 Lock and key

The standard paradigm for protease–peptide interaction is the 'lock and key' model where a sequence of amino acids fits as a key to the active site in the protease, which in the HIV-1 protease case is estimated to be eight residues long. The protease active site pockets are denoted by S (Schechter and Berger, 1967)

$$\mathbf{S} = S_4 S_3 S_2 S_1 S_{1'} S_{2'} S_{3'} S_{4'} \tag{1}$$

corresponding to residues P in the peptide

$$\mathbf{P} = P_4 P_3 P_2 P_1 P_{1'} P_{2'} P_{3'} P_{4'}. \tag{2}$$

The scissile bond is located between positions P_1 and $P_{1'}$, and P_i can take on any one of the following 20 amino acid values {A, R, N, D, C, Q, E, G, H, I, L, K, M, F, P, S, T, W, Y, V}. There are 20^8 possible values for string **P**. If the amino acids in **P** (the 'key') fit the positions in **S** (the 'lock'), then the protease will cleave the octamer between positions P_1 and $P_{1'}$. It is the goal of the machine learning algorithm to learn this 'lock and key' rule from a set $\mathcal{D}$ of N experimental observations

$$\mathcal{D} \equiv \{(\mathbf{P}(n), d(n))\}_{n=1,\ldots,N}, \tag{3}$$

where $\mathbf{P}(n)$ are observed octamers and $d(n)$ are corresponding class labels ($d = 1$ for cleaved octamers and $d = 0$ for uncleaved octamers).

2.2 Preprocessing

It is standard procedure, at least when ANN models are used to learn protein properties, to map sequence **P** to a sparse orthonormal representation (Qian and Sejnowskij, 1988). Each amino acid is then represented by a 20-bit vector with 19 bits set to zero and one bit set to one, and each amino acid vector orthonogal to all other amino acid vectors. This places an octamer **P** in one of the corners **X** of a $20 \times 8 = 160$ dimensional hypercube

$$\mathbf{P}(n) \longrightarrow \mathbf{X}(n). \tag{4}$$

Vector **X** fulfills the following eight constraints:

$$\mathbf{X} \cdot \mathbf{1}_{[(i-1)\cdot(20+1)-(i\cdot 20)]} = 1, \quad i = 1, \ldots, 8, \tag{5}$$

where $\mathbf{1}_{i-j}$ is a 160-dimensional binary vector with positions i through j set to one and the remaining set to zero. These constraints reduce the effective dimensionality of the problem from 160 down to 152.

Some consequences of using the orthonormal representation are listed below:

- The space is very high dimensional.
- All the octamers are mapped to corners in a hypercube. There are no observations inside the hypercube and linear interpolation is not useful.
- The number of possible octamers is $20^8 \ll 2^{160}$ so the hypercube is extremely sparsely populated even if one has access to all possible octamers.
- A nonlinear problem in a hypercube must be of 'XOR' type, i.e. distant corners in the hypercube belong to the same category whereas closer corners belong to opposite categories.

The properties given above are general for any problem that is mapped to the orthonormal sparse representation. In addition, there may be other properties that are specific for the problem domain. For this particular problem domain, predicting protein properties from an amino acid sequence, shift invariance has a profound effect on the distribution of data in the hypercube. Shift invariance means that the category remains unchanged if we shift peptide **P** one position to the left or right. For instance, the peptide KVFGRCELAAAMKRHGLDN is not cleaved by HIV-1 protease, which means that all the octamers KVFGRCEL, VFGRCELA,..., MKRHGLDN belong to the uncleaved category. These octamers differ only by a one position shift and they all belong to the same category, so the uncleaved category is shift invariant. The cleaved category, however, is not shift invariant because we believe the cleaving to occur at one specific site (between residues P_1 and $P_{1'}$) and not in nearby sites. Shift invariance places octamers that belong to the same category in very distant parts in the hypercube. Take the octamers KVFGRCEL and VFGRCELA for example. Their corresponding **X** representations differ by eight bits, which (due to the constraints) is the maximum possible Hamming distance between two peptides in the hypercube, yet they belong to the same category.

Intuitively, if both categories display shift invariance then the observations could be so mixed up that it is hard to separate them in a simple way, e.g. with a linear classifier. However, if any one or both of the categories do not display shift invariance, then the odds should be more in favor of a linear classifier. Examples of protein property prediction problems that display shift invariance for both categories are secondary structure prediction, signal peptide prediction and protein family classification. These are problem areas where it is likely that nonlinear, e.g. ANN, approaches are suitable. Protease specificity problems, on the other hand, should be more amenable for linear classifiers.

2.3 Classification algorithms

A linear classifier tries to find a relationship of the form

$$d(n) = \Theta[\mathbf{W} \cdot \mathbf{X}(n) + w_0] \tag{6}$$

between $\mathbf{X}(n)$ and $d(n)$. Here, $\mathbf{W}$ is a vector of free parameters, the dot represents the vector scalar product, w_0 is a threshold term and Θ is the Heaviside function (step function). Popular examples of linear classifiers are the simple perceptron, the linear SVM (LSVM) and linear logistic regression (Duda *et al*., 2001; Ripley, 1996). We have used the simple perceptron and the linear SVM in our experiments, because they are very fast and well suited for problems with binary inputs. Furthermore, the simple perceptron is guaranteed to find a linear solution if one exists.

A nonlinear classifier tries to find a relationship of the form

$$d(n) = \Phi[\mathbf{W}, \mathbf{X}(n)] \tag{7}$$

between $\mathbf{X}(n)$ and $d(n)$. Here, $\mathbf{W}$ is a vector of free parameters and Φ is a nonlinear function producing zero or one as output. Popular examples of nonlinear classifiers are the multilayer perceptron, the SVM and decision trees (Duda *et al*., 2001; Quinlan, 1986). We have used multilayer perceptrons in our experiments for comparison with previous results.

Common for both linear and nonlinear classifiers is that the parameter vector $\mathbf{W}$ is tuned by minimizing an error measure. However, the tuning process is typically orders of magnitudes faster for linear classifiers than for nonlinear classifiers.

It is good scientific practice to begin by excluding the simple things first, i.e. the possibility that a simple algorithm can solve the problem should be ruled out before a more complicated algorithm is tried. A linear classifier is simpler than a nonlinear classifier, because a linear classifier can solve fewer classification problems than the nonlinear classifier. This is usually expressed in terms of the Vapnik–Chervonenkis (VC) dimension and classifier capacity (Hertz *et al*., 1991; Lee *et al*., 1997).

2.4 Interpreting the linear classifier

A benefit with a linear model is that it is more easily interpreted than a nonlinear one and can be used to generate rules that can guide further experiments. Multilayer perceptrons are considerably more difficult to interpret (Tickle *et al*., 1998).

If the number of amino acid positions in $\mathbf{P}$ is 'large' (at least four) then it is possible to approximate the distribution of the linear argument $\mathbf{W} \cdot \mathbf{X} + w_0$ with a normal distribution. This normal distribution has mean $\mu = w_0 + \sum_{k=1}^{K} \mu_k$ and variance $\sigma^2 = \sum_{k=1}^{K} \sigma_k^2$, where

$$\mu_k = \frac{1}{20} \sum_{i=1}^{20} w_{i+(k-1)\cdot 20}, \tag{8}$$

$$\sigma_k^2 = \frac{1}{20} \sum_{i=1}^{20} w_{i+(k-1)\cdot 20}^2 - \mu_k^2, \tag{9}$$

and w_j are the components of the parameter vector $\mathbf{W}$. The μ_k will be identical if the training has been properly done, as a consequence of the constraints in Equation (5) since vector $\mathbf{W}$ should lie in the same subspace as the patterns $\mathbf{X}(n)$.

The normal distribution approximation yields

$$p(\mathbf{W} \cdot \mathbf{X} + w_0 > 0) \approx \frac{1 - \mathrm{erf}[-\mu/(\sigma\sqrt{2})]}{2} \tag{10}$$

for the probability for cleaving, where erf is the error function. Furthermore, if an amino acid is fixed in position m, corresponding to parameter w_j in $\mathbf{W}$, then the cleaving probability becomes

$$p(\mathbf{W} \cdot \mathbf{X} + w_0 > 0 \mid \text{pos. } m \text{ fixed}) \approx \frac{1 - \mathrm{erf}[-(\mu_{-m} + w_j)/(\sigma_{-m}\sqrt{2})]}{2}, \tag{11}$$

where μ_{-m} is the mean value when μ_m is excluded from the sum, and σ_{-m}^2 is the variance when σ_m^2 is excluded from the sum. This allows us to rank the importance of amino acids in different positions, based on the ratio of expressions (11) and (10). If the ratio is very large or very small, then the fixed amino acid in position m is important for the linear cleaving decision. A ratio $\gg 1$ means that the fixed amino acid increases the probability for cleaving. A ratio $\ll 1$ means that the fixed amino acid decreases the probability for cleaving.

The process can be carried further, fixing two amino acids, etc., as long as there are sufficiently many unfixed positions for the normal approximation to hold.

2.5 Data

We have used the dataset published by Cai and Chou (1998). The same data were used by Cai and Chou (1998); Narayanan *et al*. (2002), and Cai *et al*. (2002). It is, with minor modifications, a superset of the datasets used in studies previous to 1998 (Thompson *et al*., 1995; Chou, 1996). It contains 362 observations with 114 cleaved and 248 non-cleaved. The probability for linear separability, without any knowledge of

how the samples are distributed among the two categories, is 0.12% and linear separability is thus not trivially obvious (Hertz *et al*., 1991).

3 RESULTS

3.1 Learning the HIV-1 protease data

The first experiment was to use the simple perceptron (Duda *et al*., 2001), as implemented in the Matlab neural network toolbox, on the entire dataset of 362 observations. This was done to check whether the dataset was linearly separable or not. The simple perceptron algorithm is guaranteed to converge in a finite number of steps to a linear solution if a linear solution exists, and it converged very quickly (within seconds) to a perfect solution, proving that the full HIV-1 protease dataset is linearly separable. This experiment showed that there exists at least one (probably more) hyperplane that separates the cleaved octamers from the non-cleaved ones, and that the available data do not support the point that the HIV-1 protease cleavage site specificity is a nonlinear problem, contrary to what has been claimed before.

The second experiment was to successively remove positions P_i from the octamers **P** and see if the problem remained linear. We found that positions P_3 and $P_{3'}$ or P_3 and $P_{4'}$ could be removed without changing the linear separability of the data. We therefore used the two cases $P_4P_2P_1P_{1'}P_{2'}P_{4'}$ and $P_4P_2P_1P_{1'}P_{2'}P_{3'}$ in our subsequent experiments.

The third experiment was to try different classifiers, two linear and three nonlinear, on the data, using different size training sets and estimating the generalization errors using cross-validation. The linear classifiers were the simple perceptron, as above, and the LSVM, implemented in the OSU SVM Matlab Toolbox version 3.00. The nonlinear classifiers were multilayer perceptrons with two, four and eight hidden units, again using the Matlab neural network toolbox. The cross-validation was done in the following way: N_{train} samples were drawn, without replacement, from the dataset $\mathcal{D}$. These samples were then used to train the classifier until the training error was zero (perfect classification of the training data). The prediction was then tested on the remaining samples. This was repeated 100 times for every training set size, and the test error was averaged over the 100 runs. The test error is binomially distributed and the 95% confidence bars for the average test error were computed. The results for $P_4P_2P_1P_{1'}P_{2'}P_{4'}$ are shown in Figure 1. The results for $P_4P_2P_1P_{1'}P_{2'}P_{3'}$ were very similar (data not shown). The significance limits were too wide to yield any significant differences between the classifiers, but the linear classifiers had on average a lower test error and a higher Matthews correlation coefficient than the nonlinear classifiers. Furthermore, the more hidden units there were in the MLP, the worse the average test error tended to be. The conclusion was that nothing was gained with an MLP compared with using a linear classifier. Cai and Chou (1998) and Narayanan *et al*. (2002) report an error rate of ∼7–8% with an MLP with eight hidden units and with a training dataset size of about 289–300 patterns. This is better than the MLP8 results in Figure 1, but not significantly better than the linear models.

3.2 Specificity rules

The fourth experiment was to extract rules from the two linear classifiers using the technique described in the Methods section. All the 362 available observations were used to construct the linear classifiers for the analysis, expecting this to yield the best possible linear separation of the cleavable and noncleavable peptides (overtraining is less of a problem with simple classifiers). The simple perceptron solution depended on the order in which the observations were presented and we therefore averaged the solution over 100 models, each trained with a different random pattern order. The LSVM, however, had no such dependency, prompting us to construct only a single LSVM model using the 362 samples.

The base level probabilities for cleaving when no amino acid was fixed, expression (10), were 5.3 and 11.4% from the simple perceptron and the LSVM, respectively, for the $P_4P_2P_1P_{1'}P_{2'}P_{4'}$ case. For the $P_4P_2P_1P_{1'}P_{2'}P_{3'}$ case, they were 4.4 and 11.3%. We list the 5% most cleavable and noncleavable single amino acid patterns in Table 1. Because the two linear algorithms use different training methods, hyperplanes from them are different. Consequently, there are two slightly different sets of rules. The most important positions for cleaving are P_2, P_1, $P_{1'}$ and $P_{2'}$. A glutamate (E) or glutamine (Q) in position $P_{2'}$ increases the probability for cleaving. A phenylalanine (F), tyrosine (Y) or leucine (L) in position P_1 increases the probability for cleaving. A proline (P) in position $P_{1'}$ increases the probability for cleaving. A valine (V) or alanine (A) in position P_2 also increases the probability for cleaving.

There are 6000 possible pairs. Here, we only picked the 20 pairs of amino acids that had the strongest impact on the cleaving/non-cleaving decision, which are listed in Table 2. Possible motifs that emerge are: for positions (P_1, $P_{2'}$) `xxx[FYML]x[EQ]xx`; for positions (P_2,P_1) `xx[VA][FYM]xxxx` and for positions (P_2, $P_{2'}$) `xx[VA]xxExx`.

3.3 Validation

The fifth and final experiment was to test the classifiers and the rules in Tables 1 and 2 on experimental data collected after the HIV-1 protease data were published.

Some very interesting new data for HIV-1 protease have been collected using bacteriophage peptide display libraries (Smith *et al*., 1995), which is a procedure that comes close to random sampling of the peptide space. Beck *et al*. (2000) used a phage display library to screen for sequences efficiently cleaved by the HIV-1 protease, and list 45 efficiently cleaved sequences. Of these, 42 (93%) contain one or more

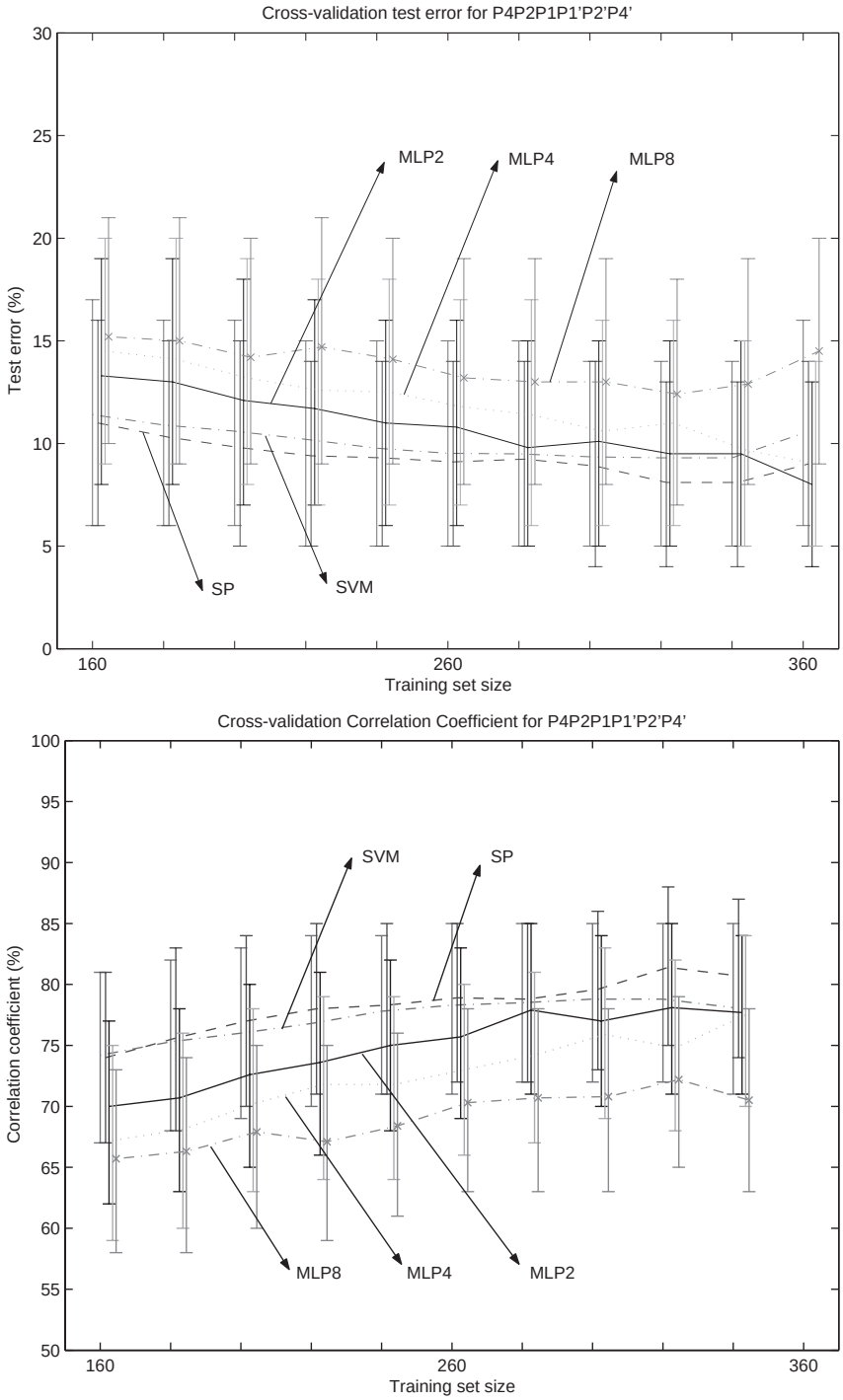

Fig. 1. Cross-validation test errors for models that use the input $P_4P_2P_1P_{1'}P_{2'}P_{4'}$. The top panel shows the test errors (% misclassified) and the bottom panel shows the Matthews correlation coefficient on the test sets.

of the single amino acids listed in the top left half of Table 1 and 39 (87%) contain one or more of the amino acids in the top right half. All $P_4P_2P_1P_{1'}P_{2'}P_{4'}$ classifiers predict 41 of the 45 sequences to be cleavable, and the four sequences where errors are made lie close to the linear decision boundary. For the $P_4P_2P_1P_{1'}P_{2'}P_{3'}$ case, the simple perceptron and linear SVM predict 38 and 39 sequences to be cleaved, respectively, and 41 sequences for MLPs.

Table 1. The 12 single amino acids and positions that influence the cleaving/non-cleaving decision the most, arranged in order of decreasing importance

$P_4P_2P_1P_{1'}P_{2'}P_{4'}$ Simple perceptron Sequence	p_c (%)	LSVM Sequence	p_c (%)	$P_4P_2P_1P_{1'}P_{2'}P_{3'}$ Simple perceptron Sequence	p_c (%)	LSVM Sequence	p_c (%)
xxxxxExx	37	xxxxxExx	40	xxxxxExx	25	xxVxxxxx	36
xxxFxxxx	33	xxxFxxxx	38	xxxFxxxx	21	xxAxxxxx	35
xxxYxxxx	20	xxVxxxxx	34	xxxLxxxx	18	xxxFxxxx	35
xxxxxQxx	19	xxAxxxxx	32	xxxYxxxx	17	xxxxxExx	34
xxxxPxxx	19	xxxxxQxx	30	xxxxPxxx	16	xxxYxxxx	34
xxVxxxxx	18	xxxxxxxK	29	xxxxxQxx	16	xxxxPxxx	30
xxxKxxxx	0.2	xxxxxKxx	0.5	xxxKxxxx	0.0	xxxKxxxx	0.2
xxxxxKxx	0.2	xxxKxxxx	1.1	xxxxxKxx	0.1	xxKxxxxx	0.7
xxKxxxxx	0.4	xxFxxxxx	1.1	xxKxxxxx	0.2	xxxDxxxx	0.9
xxQxxxxx	0.5	xxxxxSxx	1.5	xxQxxxxx	0.2	xxQxxxxx	1.2
xxxxCxxx	0.7	xxQxxxxx	1.5	xxxQxxxx	0.4	xxFxxxxx	1.2
xxFxxxxx	0.7	xxKxxxxx	1.5	xxxxCxxx	0.5	xxxxxKxx	1.7

The columns denoted p_c show the estimated probability for cleaving. The lower half shows those amino acids that inhibit cleavage, and the top half shows the amino acids that promote cleavage.

Table 2. The 20 pairs of amino acids and positions that influence the cleaving/non-cleaving decision the most, arranged in order of decreasing importance

$P_4P_2P_1P_{1'}P_{2'}P_{4'}$ Simple perceptron Sequence	p_c (%)	LSVM Sequence	p_c (%)	$P_4P_2P_1P_{1'}P_{2'}P_{3'}$ Simple perceptron Sequence	p_c (%)	LSVM Sequence	p_c (%)
xxxFxExx	91	xxxFxExx	82	xxxFxExx	73	xxVFxxxx	80
xxxYxExx	81	xxVxxExx	79	xxxLxExx	69	xxVYxxxx	79
xxxLxExx	76	xxAxxExx	78	xxxYxExx	66	xxAFxxxx	78
xxxFxQxx	75	xxVFxxxx	77	xxxxPExx	61	xxAYxxxx	78
xxVxxExx	74	xxAFxxxx	75	xxVxxExx	58	xxVxxExx	74
xxxxPExx	74	xxxYxExx	73	xxxFxQxx	58	xxxFxExx	74
xxVFxxxx	70	xxxMxExx	73	xxxFPxxx	58	xxxYxExx	74
xxxFPxxx	70	xxxFxQxx	72	xxxMxExx	55	xxAxxExx	74
xxIxxExx	69	xxxxxExK	72	xxVFxxxx	55	xxVMxxxx	73
xxxMxExx	67	FxxxxExx	70	xxxLxQxx	53	xxAMxxxx	72
xxxKxKxx	≈0	xxFxxKxx	≈0	xxxKxKxx	≈ 0	xxKKxxxx	≈0
xxKKxxxx	≈0	xxxKxKxx	≈0	xxKKxxxx	≈ 0	xxQKxxxx	≈0
xxQKxxxx	≈0	xxQxxKxx	≈0	xxQKxxxx	≈ 0	xxFKxxxx	≈0
xxKxxKxx	≈0	xxKxxKxx	≈0	xxxKCxxx	≈ 0	xxYKxxxx	≈0
xxxKxSxx	≈0	xxxxCKxx	≈0	xxKxxKxx	≈ 0	xxxKCxxx	0.01
xxQxxKxx	≈0	CxxxxKxx	≈0	xxxKxHxx	≈ 0	xxKDxxxx	0.01
xxxTxKxx	≈0	xxFKxxxx	≈0	xxxKNxxx	≈ 0	xxxKxKxx	0.01
xxFKxxxx	≈0	xxYxxKxx	≈0	xxxKxGxx	≈ 0	xxxKNxxx	0.01
xxxKCxxx	≈0	xxxxxKxV	≈0	xxQxxKxx	≈ 0	xxGKxxxx	0.01
xxxPxKxx	≈0	xxFxxSxx	≈0	xxxQxKxx	≈ 0	xxxKKxxx	0.01

The columns denoted p_c show the estimated probability for cleaving and '≈ 0%' means that the probability for cleaving is <0.01%.

Beck *et al*. (2001) list 38 sequences that they tried cleaving with both HIV-1 protease and feline immunodeficiency virus (FIV) protease. Of these, 13 were not cleaved and 25 were cleaved by HIV-1 protease. All 25 (100%) cleaved sequences contain at least one of the profiles listed in Table 1, and 16 (64%) of the 25 cleaved sequences contain one of the pair profiles listed in Table 2. All classifiers correctly predict all cleavable sequences to be cleavable, except the simple perceptron with one misclassification in the $P_4P_2P_1P_{1'}P_{2'}P_{3'}$ case. All classifiers predict less than half of the non-cleaved

sequences correctly in the $P_4P_2P_1P_{1'}P_{2'}P_{4'}$ case and more than half of them correctly in the $P_4P_2P_1P_{1'}P_{2'}P_{3'}$ case.

Tözsér *et al*. (2000) tried variations on the MA/CA cleavage site in HIV and the CA/NC cleavage site in human T-cell leukemia virus (HTLV). They list 49 sequences which they tried for cleavage by the HIV-1 protease. Three of the sequences were definitely not cleaved by the HIV-1 protease, eight were not determined and the remaining 38 were cleaved to varying degrees. We grouped the 38 cleaved ones into three groups: strong ($40 < k_{\rm cat}/K_m$, eight cases), medium ($10 < k_{\rm cat}/K_m \leq 40$, 10 cases) and weak ($0 < k_{\rm cat}/K_m \leq 10$, 20 cases). All 38 (100%) cleaved octamers contain one or more of the single amino acids listed in Table 1. All classifiers predict two of the non-cleaved sequences correctly but miss the third. For cleaved sequences, all misclassified sequences belong to weakly cleaved sequences.

The overall prediction accuracies on these three out-of-sample datasets (124 new patterns in total) were, on average, 87% for the two linear classifiers (92.1% sensitivity, 92.9% specificity and 43.1% correlation coefficient) and 89% for the three MLP models (93.9% sensitivity, 93.5% specificity and 49.8% correlation coefficient). The simple perceptron and the MLP models were all combined into committees with 100 members for the predictions. The difference was not significant when tested with McNemar's test (Ripley, 1996).

4 DISCUSSION

Linear algorithms are, in general, less complex than nonlinear algorithms. From an Occam's razor perspective, linear classifiers should therefore be ruled out before nonlinear classifiers are used. Our experiments show that there is no support in the commonly used HIV-1 protease cleavage dataset for a nonlinear algorithm, and we see no point in advocating the use of neural networks, non-LSVMs or decision trees for this problem before a dataset is available that supports nonlinear models. The experiments also show that even a linear classifier is under-specified by the full dataset, since two residue positions (P_3 and $P_{3'}$ or P_3 and $P_{4'}$) could be removed and the problem was still linearly separable.

The rules listed in Tables 1 and 2 make sense from a chemical point of view. glutamate (E) and glutamine (Q) are similar amino acids, so exchanging one for the other in the $P_{2'}$ position should not make a big difference. Phenylalanine (F), tyrosine (Y), methionine (M) and leucine (L) are also somewhat similar amino acids and it makes sense that exchanging one for the other would not cause dramatic changes. Similarly, valine (V) and alanine (A) are also quite similar and exchanging one for the other in position P_2 should not matter much. In addition, those amino acids in our rules fit the cleavage pocket very well (Beck *et al*., 2000, 2001; Tözsér *et al*., 2000).

The out-of-sample tests on the data published by Beck *et al*. (2000, 2001) are interesting since their data represent an almost random sampling of the peptide space, searching for very cleavable sequences. The data from these two references support the single amino acid rules in Table 1 since 67 out of the 70 (96%) listed cleavable peptides contain one of the rules in the table. The probability for this to happen by chance is $<10^{-18}$ (considering the fact that some octamers occur more than once in the lists).

Table 3. Cleavage specificity rules mentioned in the references (the table shows those amino acids and positions that are characteristic for cleaved octamers)

Pos.	Beck *et al*. (2001)	Beck *et al*. (2000)	Tözsér *et al*. (2000)
P_4	S	S	S T
P_3			F L
P_2		V	
P_1		F Y	F Y L M
$P_{1'}$			P
$P_{2'}$	E Q	E	
$P_{3'}$		T S	

Table 3 lists rules that are described in Beck *et al*. (2000, 2001) and Tözsér *et al*. (2000). Our predicted rules match very well with them, which supports the validity of Table 1 and also of Table 2. It is also worth noting that a decision tree does not come up with rules that are anywhere nearly as clear as the ones in Table 1 (Narayanan *et al*., 2002).

The out-of-sample performance of the linear classifiers (average 87% correct) is most likely not representative for random data and should not be compared with previously reported accuracies of 92%. All the negative (non-cleavable) test cases were, by experimental design, similar to cleavable sequences. Beck *et al*. (2001) selected sequences that were cleaved either by HIV-1 protease of FIV protease, which is similar to HIV-1 protease. Thus, the negative cases were sequences cleaved by FIV protease but not by HIV-1 protease, which probably are sequences that are difficult to predict the cleaving for. Similarly, Tözsér *et al*. (2000) varied single amino acids in two template sequences taken from the HIV and the HTLV polyproteins.

To conclude, the linear classifiers proved to be as good as nonlinear classifiers on this dataset. This simplified the interpretation considerably, making it possible to generate specificity rules that can be tested in the laboratory and used to falsify/verify the hypothesis of linearity. Furthermore, the linear classifiers are so fast to work with that analyzing a dataset and producing rules from it is done in a few hours, or less (of course, depending on the size of the data).

We emphasize that linearity is not equivalent to context independence. The linear decision rule leaves room for a lot of context dependence, as illustrated in Table 2, but it is straightforward to extract rules for this context dependence.

ACKNOWLEDGEMENTS

The authors would especially like to thank Dr Per-Åke Jovall for valuable discussions throughout the course of the work. We are very grateful to Dr Zachary Q. Beck and Prof. John H. Elder for pointers to important references. We also thank Dr Daniel Garwicz for discussions and information on proteases.

REFERENCES

Baldi,P. and Brunak,S. (2001) *Bioinformatics—The Machine Learning Approach*, 2nd edn. The MIT Press, Cambridge, MA.

Beck,Z.Q., Hervio,L., Dawson,P.E., Elder,J.E. and Madison,E.L. (2000) Identification of efficiently cleaved substrates for HIV-1 protease using a phage display library and use in inhibitor development. *Virology*, **274**, 391–401.

Beck,Z.Q., Lin,Y.-C. and Elder,J.E. (2001) Molecular basis for the relative substrate specificity of human immunodeficiency virus type 1 and feline immunodeficiency virus proteases. *J. Virol.*, **75**, 9458–9469.

Cai,Y.-D. and Chou,K.-C. (1998) Artificial neural network model for predicting HIV protease cleavage sites in protein. *Adv. Eng. Software*, **29**, 119–128.

Cai,Y.-D., Liu,X.-J., Xu,X.-B. and Chou,K.-C. (2002) Support vector machines for predicting HIV protease cleavage sites in protein. *J. Comput. Chem.*, **23**, 267–274.

Chou,K.-C. (1996) Prediction of human immunodeficiency virus protease cleavage sites in proteins. *Anal. Biochem.*, **233**, 1–14.

Duda,R.O., Hart,P.E. and Stork,D.G., (eds) (2001) *Pattern Classification*, 2nd edn. John Wiley & Sons, Inc., New York.

Hertz,J., Krogh,A. and Palmer,R.G. (1991) *Introduction to the Theory of Neural Computation*. Lecture Notes, Santa Fe Institute, Studies in the Science of Complexity, **1**. Addison-Wesley.

Lee,W.S., Bartlett,P.L. and Williamson,R.C. (1997) Correction to 'lower bounds on the VC-dimension of smoothly parametrized function classes'. *Neural Comput.*, **9**, 765–769.

Narayanan,A., Wu,X. and Yang,Z. (2002) Mining viral protease data to extract cleavage knowledge. *Bioinformatics*, **18**, S5–S13.

Qian,N. and Sejnowskij,T.J. (1988) Predicting the secondary structure of globular proteins using neural network models. *J. Mol. Biol.*, **202**, 865–884.

Quinlan,J.R. (1986) Induction of decision trees. *Machine Learning*, **1**, 81–106.

Ripley,B.D. (1996) *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge.

Schechter,I. and Berger,A. (1967) On the size of the active site in proteases. *Biochem. Biophys. Res. Commun.*, **27**, 157–162.

Smith,M.M., Shi,L. and Navre,M. (1995) Rapid identification of highly active and selective substrates for stromelysin and matrilysin using bacteriophage peptide display libraries. *J. Biol. Chem.*, **270**, 6440–6449.

Thompson,T.B., Chou,K.-C. and Zheng,C. (1995) Neural network prediction of the HIV-1 protease cleavage sites. *J. Theoret. Biol.*, **177**, 369–379.

Tickle,A.B., Andrews,R., Golea,M. and Diederich,J. (1998) The truth will come to light: directions and challenges in extracting the knowledge embedded within trained artificial neural networks. *IEEE Trans. Artif. Neural Networks*, **9**, 1057–1068.

Tözsér,J., Zahuczky,G., Bagossi,P., Louis,J.M., Copeland,T.D., Oroszlan,S., Harrison,R.W. and Weber,I.T. (2000) Comparison of the substrate specificity of the human T-cell leukemia virus and human immunodeficiency virus proteinases. *Eur. J. Biochem.*, **267**, 6287–6295.