Journal of Data Science **8**(2010), 269-288

# Information Fusion for Biological Prediction

Stefan Jaeger and Su-Shing Chen
*Shanghai Institutes for Biological Sciences*

*Abstract*: Information fusion has become a powerful tool for challenging applications such as biological prediction problems. In this paper, we apply a new information-theoretical fusion technique to HIV-1 protease cleavage site prediction, which is a problem that has been in the focus of much interest and investigation of the machine learning community recently. It poses a difficult classification task due to its high dimensional feature space and a relatively small set of available training patterns. We also apply a new set of biophysical features to this problem and present experiments with neural networks, support vector machines, and decision trees. Application of our feature set results in high recognition rates and concise decision trees, producing manageable rule sets that can guide future experiments. In particular, we found a combination of neural networks and support vector machines to be beneficial for this problem.

*Key words:* Classifier combination, cleavage site detection, decision trees, HIV, information fusion, neural networks, support vector machines.

## 1. Introduction

The fight against AIDS is one of the most prominent endeavors in current health programs. One of the important strategies followed is to stop viral replication in people with HIV (human immunodeficiency virus) infection. In order to do so, a possible starting point is the inhibition of enzymes essential to the replication of the AIDS virus. HIV-1 protease is such an enzyme. The proteins comprising the human immunodeficiency virus are produced in the form of long polyproteins, which must be cleaved in order to yield the active protein components of the mature virus. HIV-1 functions to cleave the nascent polyproteins during viral replication, with the chemical action taking place at a localized active site on its surface. Researchers have the idea of preventing the chemical action of the protease by binding molecules, so-called HIV-1 protease inhibitor drugs, to its active site. These inhibitors permanently occupy the active site and thus prevent permanently the normal functioning of the HIV-1 protease enzyme (Lumini and Nanni, 2006). The development of efficient inhibitors is a difficult task,

though. HIV-1 protease cleaves at different sites with little or no sequence similarity (You, Garwicz and Rögnvaldsson, 2005). Where a peptide will be cleaved by HIV-1 protease is largely unknown. However, a comprehensive understanding of the cleavage site structure is necessary to synthesize efficient inhibitors. The standard model for protease-peptide interaction is the "lock-and-key" principle in which a sequence of amino acids fits as a key to the active site in the protease (Lumini and Nanni, 2006). In the case of HIV-1 protease, the length of the key is eight; i.e., it comprises eight amino acids. Given that there are altogether 20 different amino acids, the size of the total search space for potential cleavage sites is $20^8$. An exhaustive search of this space is currently prohibitive; a fact that is not likely to change in the foreseeable future. Therefore, this situation calls for a computer-aided approach.

We can formulate cleavage site detection as a typical classification problem, with the key of amino acids defining the input features and a corresponding binary class label indicating the presence or absence of a cleavage site. The problem therefore allows us to apply the whole plethora of techniques developed in the pattern recognition field, such as neural networks, etc. Of course, this does not mean that there is an out of the box solution for this kind of problem. On the contrary, classifier design and feature selection for HIV-1 protease cleavage site detection are hard problems still waiting for an efficient solution. In particular, the high-dimensional feature space and very small training sets are an especially challenging obstacle here. The manageable number of existing publications dealing with this problem have created decent recognition rates. Nevertheless, the recognition rates still leave much to be desired in view of the ultimate goal: generating powerful rules for cleavage site characteristics that could lead future in-vitro experiments. HIV-1 protease cleavage site detection is a hard problem in this respect.

Here, we present a new set of features for HIV-1 protease cleavage site prediction, and also experiment with classifier combination. The idea of applying multiple classifiers to cleavage site detection is relatively new. The few existing approaches are basically ensemble methods that have their focus on the creation of classifiers (Lumini and Nanni, 2006). However, we concentrate on the actual combination, trying to show the usefulness of a new information-theoretical combination method that the authors have already successfully applied to other application domains. We structured our paper as follows: Section 2 discusses the advantages of classifier combination, listing the most important research directions in this field. Section 3 then presents existing approaches for HIV-1 protease cleavage site prediction, and also introduces our own feature set and approach. In Section 4, we show our practical results, followed by a conclusion at the end of the paper.

## 2. Classifier Combination

The combination of classifiers has turned out to be one of the most promising approaches in pattern recognition. Many researchers currently investigate this interesting field in areas as different as handwriting recognition for human computer interaction or cleavage site detection for applications in computational biology (Jaeger, Ma and Doermann, 2008). Multiple classifier systems (MCS) offer an advantage over single classifier systems in that they distribute information among more than one classifier. This reduces the complexity of the problem, given that one individual classifier is often overburdened with the task of accommodating different information in a single architecture. For instance, hyperplanes provided by statistical classifiers, such as neural networks, represent only one type of information that is usually quite different from the class boundaries computed by decision trees or nearest neighbor classifiers. Generally speaking, a set of simple classifiers is easier to train and optimize than a single system with high complexity. The main motivation for multiple classifier systems is, of course, the prospect of outperforming the best single classifier with the combined performance of the whole classifier set. In fact, a multiple classifier system can provide excellent recognition rates, even if all its constituent classifiers show only mediocre performance when applied individually. In such a system, the weakness of one classifier is usually compensated by the strength of at least one other classifier.

An important issue that needs to be addressed when implementing multiple classifier systems is the combination scheme; i.e., the question of how to integrate the diverse outputs of the various classifiers of a multiple classifier system into a single classification result. In practice, classifier outputs are not necessarily probabilities in the mathematical sense. Very often they describe distances to decision boundaries and can thus have arbitrary values. This poses no problem for a single classifier system where only values of the same kind need to be considered. For multiple classifier systems, however, the incompatibility between outputs is a serious problem.

Outputs of classifiers are denoting the confidence of the classifiers in their classification result. Incompatibility between these confidence values typically shows in the different ranges and scales they have. In order to combine classifiers in a meaningful manner, we have to overcome these incompatibilities. Many combination schemes tackling this problem have been proposed in the literature. Unfortunately, despite the large number of techniques proposed, there is still no widely accepted framework that most experts would agree to be optimal (Jaeger, Ma and Doermann, 2008). In fact, the large number of quite different techniques proposed shows the uncertainty present. Given this unsatisfactory situation, we

confine ourselves to mention just a few approaches that have gained some significance over the past years: the approach by Dempster/Shafer or the Behavior-Knowledge Space (BKS) method. The former is the familiar theory of evidence based on belief functions as it is known in the field of artificial intelligence, while the latter computes extensive statistics for the n-best lists of each classifier. Both methods are relatively complex techniques that need a lot of training data in order to be able to compute meaningful statistics on the classifier's confidence values. The same holds for techniques trying to solve the compatibility problem by just applying another classification step on the output values during post-processing. For this reason, these techniques are not suited to HIV-1 protease cleavage site detection. It is also by no means obvious that more complex combination schemes are indeed more powerful than elementary methods, such as selecting the class with the maximum classifier score (max-rule) or selecting the class with the maximum sum of classifier scores (sum-rule). Much theoretical work still needs to be done here to fully understand the theoretical basis of classifier combination.

Here, we apply two simple methods, namely voting and a new information-theoretical method. The simple idea behind voting is to count for each specific class the number of classifiers that output this particular class as the most likely class for a given input pattern. The final classification result is then simply the class that receives the most votes, after applying a tie-breaker in case of ties, if necessary. Since voting considers only the position of candidate classes in the n-best list of classifiers, incompatible output does not negatively affect voting. Despite its simplicity, majority voting can be very powerful. The second combination method, a new information-theoretical method developed recently by the authors, will be explained in more detail below. Both combination schemes do not guarantee performance improvements; i.e., recognition rates higher than the best individual classifier of the multiple classifier system. In case of the information-theoretical method, however, the authors have achieved consistently good results in various application domains.

## 3. HIV Cleavage Site Prediction as a Classification Problem

In recent years, the predominant understanding of HIV cleavage site prediction has been that of a classification problem, and this is also the point of view that we adopt in this paper. For a review of earlier attempts, readers are referred to Chou (1996). Two things are key to the design of a classifier: the input coding and the classification architecture itself, which we are going to discuss in the following two subsections.

### 3.1 Coding

The input data to a classifier is of utmost importance since a careless choice of input variables can easily compound the classifier's problem of finding the decision boundaries for each class. There is no obvious input coding that would seem to be the natural choice for HIV cleavage site prediction. Most authors of recent works use octapeptide sequences (octamers) to encode the classifier input. An octapeptide sequence is composed of eight amino acids: $a_1a_2a_3a_4a_5a_6a_7a_8$, with each position $a_i$ denoting one of twenty possible amino acids. The point of cleavage, which is also known as scissile bond, is located in the middle of the peptide sequence; i.e., between $a_4$ and $a_5$. For instance, a typical input to the classifier could be

$$TQIMFETF \quad 1$$

where each character denotes an amino acid (e.g. T stands for Threonine) and the last digit indicates whether or not a scissile bond is present (either 1 or 0).

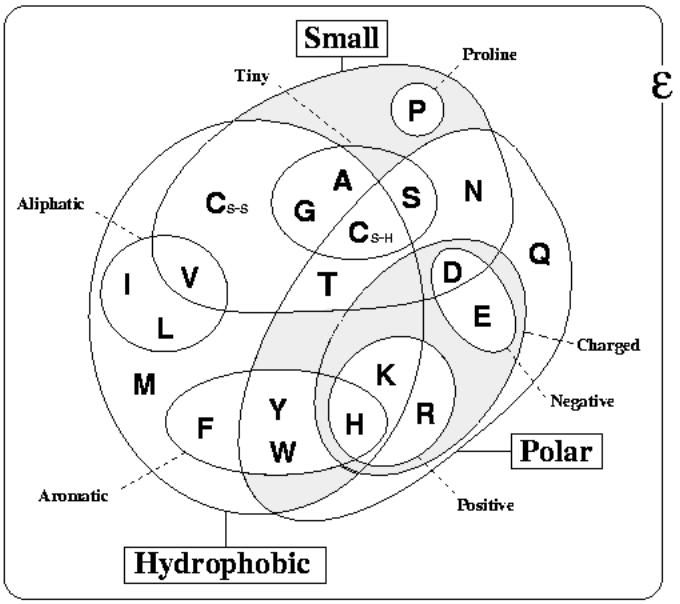


Figure 1: Grouping of amino acids according to their properties (Taylor, 1986).

To input octapeptide sequences into a classifier, such as a neural network, we need a suitable representation. Again, the representation best suited for this problem is not known, and so several different representations have been used in practice. The straightforward orthonormal encoding scheme, which represents each amino acid by a 20 bit vector with exactly one bit set to one and all of

the remaining bits set to zero, has been widely used ( Cai and Chou, 1998; Narayanan, Wu and Yang, 2002; Rögnvaldsson and You, 2004).

However, given that we have a sequence of eight amino acids, this type of encoding defines a very high-dimensional feature space, with each octamer being a corner in a $20 * 8 = 160$ dimensional hypercube. Due to the small number of training patterns typically available for HIV cleavage site prediction, we do not consider the orthonormal representation very promising as it leads to an extremely sparse space. A better alternative is the property-based encoding (You, Garwicz and Rögnvaldsson, 2005). Here, each amino acid is described by its physical or chemical properties. For instance, Figure 1 groups amino acids according to their properties (Taylor, 1986). This is just one of many possible groupings, but arguably covers the most protein context.

Without going into the details of the categorization in Figure 1, we see that there are three major attributes: small, polar, and hydrophobic. The latter refers to a molecule's property of being repelled from water. You *et al.* used two of these features to code amino acids, namely size and hydrophobicity (You, Garwicz and Rögnvaldsson, 2005). For example, the amino acid with the character code "F", which stands for Phenylalanine, transforms into the binary code "01", representing the fact that it is not small and that it belongs to the group of hydrophobic acids. This type of coding transforms an octamer to a binary string with 16 bits, replacing each amino acid at each position with the two bits for the two properties. As a result, the feature space becomes more dense compared to the orthonormal encoding scheme. It now covers $2^{16}$ instances.

In case of using as little as two features for property-based encoding, as in You, Garwicz and Rögnvaldsson (2005), different octamers can have identical codings. Apart from this undesired characteristic, we do not think that a coarse binary representation is the ideal solution. Instead of the binary encoding scheme, we propose to use the actual values of each physical or chemical property. This should enable the classifier to learn finer decision boundaries. In the following, we are going to encode each amino acid based on the values of four properties:

- Hydropathy index
- Molecular mass
- Polarity
- Occurrence percentage

The hydropathy index is a scale combining hydrophobicity and hydrophilicity of R groups. It can be used to measure the tendency of an amino acid to seek an aqueous environment (negative values) or a hydrophobic environment (positive values) (Nelson and Cox, 2005). The occurrence percentage describes the average

occurrence of a particular amino acid (R group) in a set of more than 1,150 proteins (Nelson and Cox, 2005). All four properties together describe an amino acid by a group of four real values. The four properties chosen by us are by no means the only possible choice. Other sets with a larger, or even smaller, number of features may work as well. The properties listed above seemed appealing to us and indeed lead to good recognition rates, so we did not see the necessity to investigate further into finding the optimal feature set. This may be a promising area for future research, though. The important point for us was to operate on real-valued features instead of binary input vectors, and show their effectiveness for this kind of problem.

Figure 2 shows a typical 32-dimensional, real-valued feature vector.

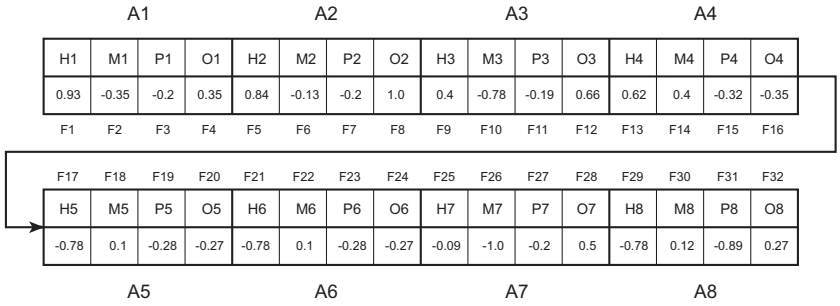


Figure 2: A typical feature vector.

The first sixteen features ($F1$ to $F16$) describe the four amino acids ($A1$ to $A4$) in front of the potential cleavage site, while the last sixteen features ($F17$ to $F32$) describe the succeeding amino acids ($A5$ to $A8$). Accordingly, each amino acid $A_i$ is encoded by four features at four different positions, $F_{4(i-1)+k}, k = 1, \ldots, 4$. The features of $A_i$ are $H_i, M_i, P_i, O_i$, describing its hydropathy index, molecular mass, polarity, and occurrence percentage, respectively. The actual feature values shown in Figure 2 are the result of a linear normalization of mean values and standard deviations. In particular, we normalize each feature $F_i$ independently according to

$$F_i := (F_i - mean(F_i))/std(F_i), \tag{3.1}$$

where *mean* and *std* compute the mean and standard deviation of $F_i$, respectively. This operation assigns each feature a mean of zero and a standard deviation of 1. It is the only pre-processing operation we perform on the features.

### 3.2 Classifiers

We employ three types of classifiers: neural networks, support vector machines, and decision trees. Artificial neural networks are a tried and trusted

technique that has been successfully applied to many different pattern recognition problems. They are statistical classifiers by nature. And so are support vector machines, which minimize the empirical classification error and maximize the geometric margin. Generally speaking, they provide recognition rates comparable to those of neural networks. However, they are based on an arguably nicer theoretical framework and have less parameters to tune.

On the other hand, decision trees are symbolic classifiers, which, in most practical cases, provide slightly inferior recognition rates compared to neural networks and support vector machines. Nevertheless, they are an interesting alternative because a trained decision tree allows deduction of rules providing insight into the problem, while most statistical classifiers treat the problem merely as a black box. With HIV-1 protease cleavage site detection in mind, let us have a closer look at each classification architecture.

**Neural networks**

Thompson *et al.* were the first to apply artificial neural networks to the problem of HIV-1 protease cleavage site detection (Thompson *et al.*, 1995; Kim *et al.*, 2008). Other noteworthy approaches applying neural networks are the works of Cai and Chou (1998), Kesmir *et al.* (2002) and Thomson *et al.* (2003). The latter exploited prior knowledge in combination with neural networks, achieving slightly higher recognition rates. Generally speaking, neural networks are at the forefront of powerful recognition techniques for HIV-1 protease cleavage site detection. They can reach recognition rates around 90% for this kind of problem.

Biologists are very much interested in getting a better understanding of HIV-1 protease cleavage site characteristics. Unfortunately, the distributed knowledge of neural networks is difficult to extract. There is no straightforward way to generate symbolic rules from a trained network that could guide future in vitro experiments. Nevertheless, extraction of rules from neural networks is possible, see e.g., Andrews *et al.* (1995), Diederich *et al.* (1999). In fact, several rule extraction techniques have already been applied to HIV-1 protease cleavage site detection, e.g., Kim *et al.* (2008), Yang, Dalby and Qiu (2004). Rögnvaldsson and You, and You *et al.*, use simpler linear classification models to extract rules to facilitate the rule extraction task ( Rögnvaldsson and You, 2004; You, Garwicz and Rögnvaldsson, 2005). The method they use is computationally expensive, however, and there is no guarantee that cleavage site detection is linear by nature.

**Support vector machines**

Gaussian support vector machines are similar in performance to neural networks (Cai and Chou, 1998; Cai *et al.*, 2002; Narayanan, Wu and Yang, 2002).

Like neural networks, they show better performance than linear support vector machines and decision trees (Rögnvaldsson and You, 2004; You, Garwicz and Rögnvaldsson, 2005).

### Decision trees

Decision trees are a very appropriate architecture for obtaining explicit knowledge about cleavage sites. From a decision tree, we can derive symbolic rules in a straightforward way. In terms of recognition performance, decision trees are largely overshadowed by neural networks and other classifiers. Their appealing way of representing knowledge makes them an interesting alternative nonetheless (Narayanan, Wu and Yang, 2002). The higher performance of neural networks can result in rules that are more powerful in the sense that they describe cleavage sites more accurately. On the other hand, the rules of a decision tree are straightforward to extract. There is a good chance that rules taken from a decision tree a more concise and also more intuitive. In addition, rules from a decision tree will most certainly be very different from the rules derived from a neural network for instance. They will thus provide new insights, despite their slightly lower performance. It is also possible that a comparison of rules derived from multiple classifiers may help to get more confidence in a particular rule: If two different classifiers produce a similar rule, then it is very likely that this rule indeed reflects the biological reality. In such a case, experimentalist can try to verify this rule by carrying out further experiments in the biological search space in order to confirm this rule empirically, generating as a by-product more training material with which the classifiers can be retrained. The advantages of such an iterative method over a blind search are obvious.

Our intention in this paper is to first create a classification system and build a better foundation for rule extraction before actually deriving rules. In this paper, we therefore confine ourselves to the classification problem and present only rules derived from decision trees. We use a standard design for our decision trees (Breimann *et al.*, 1993).

## 4. Experimental Results

In this section, we present our data sets, set-ups, and recognition rates.

### 4.1 Datasets

There is a notorious shortage of training and test data in many pattern recognition applications, including HIV-1 protease cleavage site detection. There are only a few publicly available datasets for this problem. Cai and Chou compiled

a popular dataset with 362 peptides that has been used in several publications (Cai and Chou, 1998; Narayanan, Wu and Yang, 2004; You, Garwicz and T. Rögnvaldsson, 2005).

Kim *et al.* collected another dataset with 392 new example patterns. They created this new dataset by collecting experimental data from published literature for oligopeptide sequences that have been exposed to the HIV-1 protease (Kim *et al.*, 2008).

Taken together, the datasets of Cai/Chou and Kim *et al.* provide a set of 754 distinct peptide sequences.

Another dataset produced by de Oliveira *et al.* is the result of experiments on variations of protease cleavage sites related to HIV-1 subtype C (Kim *et al.*, 2008; deOliveira *et al.* 2003). This dataset contains 133 cleaved octamers; 131 of which do neither occur in the Cai/Chou database nor in the database of Kim *et al.*.

At the time of this writing, all three databases are publicly available under http://www.cise.ufl.edu/∼suchen/sbl, where the Cai/Chou and Kim *et al.*. databases have been merged into a single file, with the filenames indicating the number of patterns included in the database, respectively, and the duplicates being removed from de Oliveira's database. As in Kim *et al.* (2008), we will refer to these two datasets as 754-dataset and 133-dataset, respectively.

Table 1: Neural network parameter tests

| #Neurons | 300 Epochs | 500 Epochs | 750 Epochs | 1000 Epochs |
|---|---|---|---|---|
| 10 | 88.28 | 87.46 | 86.41 | 86.70 |
| 20 | 89.02 | 88.41 | 88.55 | 88.14 |
| 30 | 88.64 | 88.90 | 88.40 | 87.86 |
| 40 | 89.15 | 88.70 | 87.97 | 88.03 |
| 50 | 88.36 | 89.21 | 88.64 | 88.78 |
| 60 | 89.21 | 89.35 | 89.38 | 87.61 |
| 70 | 88.98 | 89.72 | 88.79 | 89.30 |
| 80 | 88.73 | 89.35 | 88.98 | 88.84 |
| 90 | 89.08 | 88.82 | 89.27 | 89.16 |
| 100 | 88.81 | **89.86** | 89.38 | 89.18 |

### 4.2 Individual classifiers

Before we present our combined recognition rates, let us show the single recognition rates for each classifier and also some of the rules derived from the decision trees.

### Neural networks

In our experiments, we use a two-layer feed-forward network with hidden neurons and two output neurons, one for each possible answer (cleavage site or non-cleavage site). Each of the 32 input features connects to each of the hidden neurons. All neurons use the standard log-sigmoid transfer function to compute their output. As training function, we use the typical gradient descent method implemented through backpropagation in combination with an adaptive learning rate. Table 1 lists the recognition rates for different numbers of hidden neurons and training epochs.

All recognition rates are based on a 10-fold cross-validation performed on the entire set of samples described above. We observe only moderate difference in recognition performance, with the highest recognition rate achieved with 100 hidden neurons and the net trained with 500 epochs. These are the parameters we will use in our combination experiments later on.

### Support vector machines

For our experiments with support vector machines, we used the implementation in Chang and Lin (2001). Table 2 shows the recognition rates for a 10-fold cross-validation on our data set.

Table 2: Support vector machines: recognition rates for different kernels.

| Kernel Type | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| linear | 84.79 | 86.16 | 85.31 | 86.44 | 87.54 | 87.57 | 88.67 | 86.16 | 85.35 | 84.75 | 86.27 |
| polynomial | 86.12 | 88.67 | 91.22 | 88.98 | 88.14 | 88.45 | 87.32 | 85.59 | 85.92 | 86.16 | 87.66 |
| radial basis | 88.42 | 89.55 | 88.42 | 88.70 | 88.14 | 88.10 | 89.83 | 88.17 | 88.98 | 89.83 | 88.81 |
| sigmoid | 80.00 | 81.97 | 81.07 | 80.51 | 83.29 | 83.29 | 80.45 | 79.94 | 81.41 | 78.53 | 81.05 |

The table shows that the radial basis kernel is superior to other kernels. We will therefore use this kernel in our combination experiments.

### Decision trees

Figure 3 shows an example of a decision tree based on the features introduced in Section 3.1.

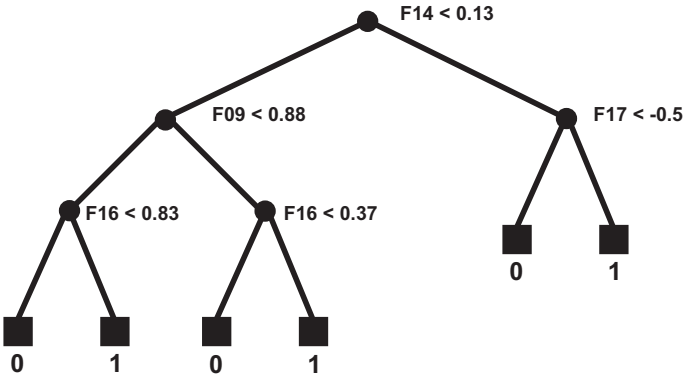


Figure 3: Tree 1.

We built the tree by using three-quarters of the 754-dataset for training. It is a fairly-well structured tree of moderate size. It takes only maximal three decisions to come to a conclusion for an unknown input pattern. The tree shown in Figure 3 reaches a performance of about 86.3% on the 133-dataset, which is a recognition rate that already performs favorably with other decision trees based on the orthonormal encoding schemes or other features discussed in Section 3.1 (Kim *et al.*, 2008). We see this result as a first confirmation that our proposed features are indeed powerful. The small size of the tree results in a compact set of short rules. For instance, following the rightmost branch of the tree, we can derive the following rule:

$$\text{IF} \quad (F14 >= 0.13) \quad \text{AND} \quad (F17 >= -0.5)$$
$$\longrightarrow \text{clevage site dected}$$

According to this rule, we have detected a cleavage site if Feature $F14$ is greater than, or equal to 0.13, and Feature $F17$ is greater than, or equal to $-0.5$. In other words, the rule predicts a cleavage site if the normalized molecular mass value of the fourth amino acid is at least 0.13 and the normalized hydropathy index of the fifth amino acid is at least $-0.5$. The appealing characteristic of all rules derivable from the tree in 3, including the rule given above, is their locality: Almost all features occurring on the left-hand side of each rule fall into a small section around the potential cleavage site. For example, the rule given above, contains the features $F14$ and $F17$ as building blocks, describing properties of amino acids at Position 4 and 5 of the given octamer. These are the positions right next to the potential cleavage site. This is another indication of the descriptive power of the chosen features, as one would expect a close proximity of causal connections to the cleavage site.

Figure 4 shows a second tree that we trained with the whole 754-dataset, instead of only using three-quarters of it.

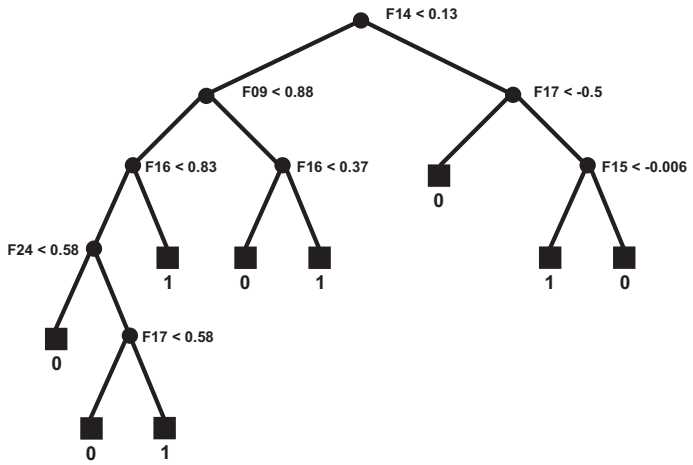


Figure 4: Tree 2.

Given the larger training set, the second tree is more complex than the first one in Figure 3. On the other hand, it provides a better performance: We tested the second tree with the same 133-dataset on which it achieves a recognition rate of over 90.0%; i.e., almost 4% more than the first tree. The second tree accomplishes this improvement by augmenting the first tree with additional branches defining a finer breakdown of the original coarse structure. For instance, following again the rightmost branch of the tree, we obtain the following, more specific rule:

$$\begin{array}{rl} \text{IF} & (F14 >= 0.13) \\ \text{AND} & (F17 >= -0.5) \\ \text{AND} & (F15 >= -0.006) \\ \longrightarrow & \text{no cleavage site found} \end{array}$$

This rule contains an additional exception, stating that in order for the site to be a cleavage site, the $F15$ value must be smaller than $-0.006$, in addition to the preconditions required by the previous rule.

Figure 5 shows a third tree, which is the result of a 10-fold cross-validation run on the whole dataset.

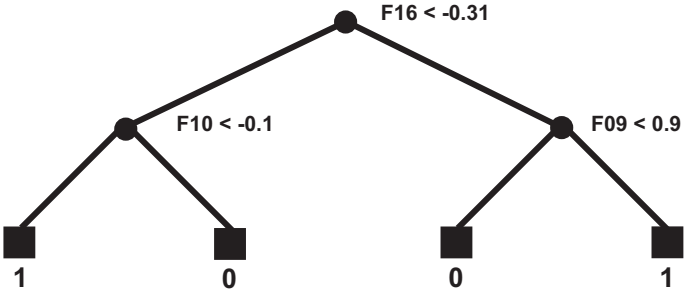


Figure 5: Tree 3.

This tree is even more concise than the two previous trees, while at the same time providing a relative good recognition rate of 85.31%. Expressed as a rule, the entire tree reads as follows:

$$\begin{array}{ll} \text{IF} & ((F16 < -0.31) \text{ AND } (F10 < -0.1)) \\ \text{OR} & ((F16 >= -0.31) \text{ AND } (F09 >= 0.9)) \\ \longrightarrow & \text{cleavage site detected} \end{array}$$

It is surprising that such a simple rule can already account for about 85% of all cleavage sites.

We also experimented with tree pruning, hoping that a pruned tree may perhaps be able to better generalize on the test set (Breimann *et al.*, 1993). However, we found that the unpruned trees perform better than their pruned correspondents. Table 3 shows a recognition run for pruned and unpruned trees.

Table 3: Decision tree: pruning vs no pruning

| Type | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Pruning | 83.10 | 83.90 | 84.14 | 82.15 | 87.22 | 85.55 | 85.59 | 83.38 | 83.15 | 83.94 | 84.21 |
| No pruning | 85.03 | 84.75 | 83.85 | 86.72 | 86.16 | 82.77 | 87.57 | 84.75 | 85.35 | 85.59 | 85.25 |

### 4.3 Combination

When it comes to combining classifiers another drawback of decision trees reveals. While neural networks and support vector machines readily provide confidence values, decision trees offer crisp decisions by nature. For this reason, we use voting to combine the output of neural networks, support vector machines, and decision trees. In order to take advantage of the confidence values of neural networks and support vector machines, we employ so-called informational confidence values, which replace the original confidence values (Jaeger, Ma and

Doermann, 2008). Informational confidence values have provided good results for character recognition and document processing (Jaeger, Ma and Doermann, 2008), and we are going to show their usefulness for HIV-1 protease cleavage site prediction as well. The next section will shortly explain the main idea of informational confidence.

**Informational confidence**

Generally speaking, a confidence value describes the confidence of a classifier in its recognition result. Confidence values are thus very important for post-processing techniques, such as language models or classifier combination, where they allow integration of information from different sources. Unfortunately, there is currently no commonly agreed standard method for computing confidence values. Confidence values come in flavors as different as distances to hyperplanes and probability estimates. Resting on information theory, one of the goals of informational confidence is to provide a standard representation for confidence. Assuming that every confidence value conveys information, informational confidence values have the following form:

$$K^{new} = -E * \ln\left(1 - p(K^{old})\right) \tag{4.1}$$

where $K^{old}$ denotes the raw confidence, provided by a neural network or support vector machine in our case, and $K^{new}$ is the newly computed informational confidence value. The scalar $E$ denotes an expectation value, which we simply set to the classifier's recognition rate in our experiments. The so-called performance function $p(K^{old})$ describes the performance of the raw confidence values, measured on an evaluation set. A small performance will result in low confidence and, vice versa, a high performance will lead to high informational confidence. Resolving for $p$ in Equation (4.1) reveals the definition of the performance function:

$$p(K^{old}) = 1 - e^{-\frac{K^{new}}{E}} \tag{4.2}$$

Based on the observation that the performance function describes an exponential distribution, we can compute performance estimates as follows, given a discretization and an evaluation set:

$$\hat{p}(K_i^{old}) = \frac{\sum_{k=0}^{i} n_{correct}(K_k^{old})}{N} \tag{4.3}$$

where $K_i^{old}$ is the ith confidence value, assuming discrete values, and $N$ is the number of patterns in the evaluation set. The help function $n_{correct}(K_k^{old})$ returns the number of patterns correctly classified with confidence $K_k^{old}$. The term

$\hat{p}(K_i^{old})$ thus estimates the relative number of patterns correctly classified with a confidence smaller than or equal to $K_i^{old}$. It defines a partial sum according to which the newly computed confidence values progress. Furthermore, the estimate defines a monotonously increasing function, ensuring that the new confidence values $K_i^{new}$ will also increase monotonously and will thus not change the relative order of the old values $K_i^{old}$. Insertion of the performance estimates into Equation (4.1) provides us with the new estimated informational confidence values $\hat{K}_i^{new}$:

$$\hat{K}_i^{new} = -R * \ln\left(1 - \hat{p}(K_i^{old})\right) \tag{4.4}$$

where $R$ is the recognition rate of the classifier estimated on the evaluation set. We perform this estimation process independently for each classifier; i.e., each neural network and support vector machine, and then simply combine all classifiers by summing up confidences for every class and choosing the class with the maximum overall confidence.

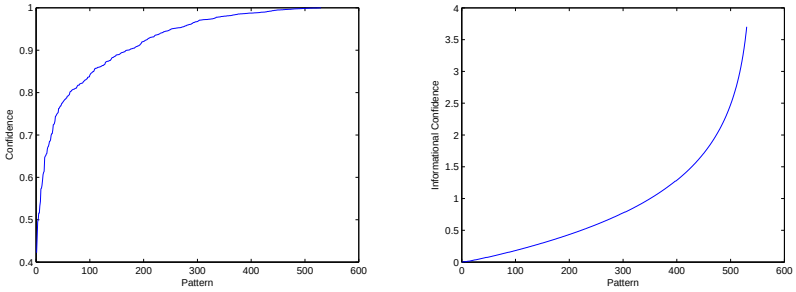


Figure 6: Confidence (left) and Informational Confidence (right) for a neural network.

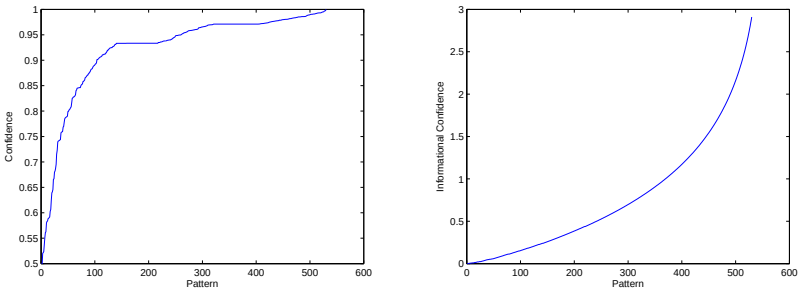


Figure 7: Confidence (left) and Informational Confidence (right) for a support vector machine.

Figure 6 shows an example for a neural network. The x-axis of both graphs in Figure 6 represents the training patterns, sorted according to their confidence

values. The y-axis shows the accumulated recognition rates (left-hand side) and the corresponding, progressing informational confidence values (right-hand side), both computed according to the equations given above. Figure 7 shows the analog diagrams for a support vector machine.

We used the training set to compute these statistics for both the neural network and the support vector machine. Table 4 shows the training rates for all three architectures.

Table 4: Training recognition rates of NN, SVM, and decision tree

| Classifier | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| NN | 98.12 | 98.87 | 98.49 | 98.49 | 98.49 | 98.49 | 99.25 | 97.93 | 96.99 | 96.99 | 98.21 |
| SVM | 96.80 | 96.99 | 96.23 | 96.42 | 96.80 | 96.60 | 96.42 | 95.68 | 95.86 | 96.61 | 96.44 |
| Tree | 95.86 | 95.86 | 96.61 | 95.48 | 96.80 | 95.09 | 95.29 | 95.86 | 95.48 | 95.67 | 95.80 |

All recognition rates are high, with the decision trees falling a bit behind, as expected. For classifiers performing less reliably or uniformly on the training set, the differences in Figures 6 and 7 would be more pronounced.

**Combined recognition rates**

Table 5 shows the individual and the combined recognition rates for all three architectures and a 10-fold cross-validation run.

Table 5: Combination of NN, SVM, and decision tree by voting and informational confidence.

| Classifier | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| NN | 88.67 | 85.88 | 89.01 | 86.24 | 87.04 | 87.57 | 87.57 | 89.24 | 89.80 | 88.95 | 88.00 |
| SVM | 89.80 | 88.42 | 87.89 | 88.20 | 89.01 | 87.29 | 88.70 | 89.24 | 88.10 | 88.39 | 88.50 |
| Tree | 82.72 | 82.49 | 83.38 | 84.55 | 82.82 | 81.36 | 79.94 | 84.99 | 84.42 | 84.70 | 83.14 |
| Voting | 87.82 | 87.57 | 89.58 | 87.92 | 88.73 | 87.57 | 88.70 | 90.08 | 90.65 | 90.08 | 88.87 |
| Inf. Conf. | 90.08 | 88.98 | 89.30 | 86.80 | 89.30 | 88.42 | 88.70 | 89.24 | 90.93 | 90.65 | 89.24 |

We can see that the support vector machines provide slightly higher recognition rates than the neural networks. Compared to these two classifiers, decision trees fall behind. They perform about 5% worse. The last two lines in Table 5 contain the combined rates achieved by voting and informational confidence, respectively. Voting provides very often, but not always, recognition rates that are at least as good as the best individual recognizer. It is, on average, about 0.37% better than the best individual recognizer; i.e., the support vector machine. This proves that, despite their inferior performance, decision trees can contribute

information that is useful for discrimination and that is not covered by other classifiers; albeit only little information in this case. We can make similar statements for the combination of neural networks and support vector machines. Again, the combined recognition rates are often better than the individual rates. On average, the combined recognition rates based on informational confidence are about 0.74% better than the best individual classifier. In particular, the combination of just two classifiers, support vector machines and neural networks, outperforms the voting combination of all three architectures. This shows that confidence values convey important information and our information-theoretical combination method succeeds in exploiting it.

## 5. Conclusion

In our paper, we have modeled HIV-1 protease cleavage site detection as a classification problem. In contrast to previous works, we have employed the physical and chemical properties of amino acids by using them as features for the classification process. In particular, we use the hydropathy index, molecular mass, polarity, and occurrence percentage. Our results show that these properties provide useful information for cleavage site prediction. We have applied three different classification architectures, namely neural networks, support vector machines, and decision trees. The first two provide slightly higher recognition rates than the latter; a fact that is in accordance with most publications comparing these architectures. Neverthless, using physical and chemical properties, we have shown that it is possible to generate very concise decision trees whose performance is only slightly worse. We think that decision trees still have a place in cleavage site prediction, arguing that the prospect of straightforward rule derivation should offset their slightly inferior performance.

We have shown that classifier combination leads to improved recognition rates for the task of cleavage site detection. Both of the combination schemes we investigated; i.e., voting as well as informational confidence, lead to better classification performances. While voting is an established technique, informational confidence is a rather new technique developed by the authors that, given the experiments reported here, has proven its usefulness for yet another application.

In conclusion, the recognition rates reported in this paper are comparable with the state-of-the-art in this area. We have shown that our chosen set of features provides appealing rules that can guide future practical experiments. We are convinced that there are still many useful chemical and physical properties of amino acids waiting to be discovered for HIV-1 protease cleavage site prediction. Finding these features is a goal of future work.

## References


Andrews, R., Diederich, J. and Tickle, A. B. (1995). Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge Based Systems* **8**, 373-389, 1995.

Breimann, L., Friedmann, J. H., Olshen, R. A. and Stone, C. J. (1993) *Classification and Regression Trees*. Chapman and Hall.

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

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

Chang, C.-C. and Lin, C,-J. *LIB SVM a library for support vector machines*, 2001. Software available at http //www.csie.ntu.edu.tw/∼cjlin/libsvm.

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. *Journal of Computational Chemistry* **23**, 267-274.

deOliveira, T, Engelbrecht, S., van Rensburg, E. J., Gordon, M., Bishop, K., zur Megede, J., Barnett, S. W. and Cassol, S. (2003) Variability at human immunodeficiency virus type 1 subtype C protease cleavage sites an indication of viral fitness. *Virology* **77** 9422-9430.

Jaeger, S., Ma, H., and Doermann, D. (2008). *Machine Learning in Document Analysis and Recognition*, chapter Combining Classifiers with Informational Confidence, pages 163-192. In *Studies in Computational Intelligence* (Edited by S. Marinai and H. Fujisawa). Springer.

Kesmir, C., Nussbaum, A., Schild, H., Detours, V. and Brunak, S. (2002). Prediction of proteasome cleavage motifs by neural networks. *Protein Engineering* **15**, 287-296.

Kim, H., Zhang, Y. Heo, Y.-S., Oh, H.-B. and Chen, S.-S. (2008). Specificity rule discovery in HIV-1 protease cleavage site analysis. *Computational Biology and Chemistry* **32**, 72-79.

Lumini, A. and Nanni, L. (2006). Machine learning for HIV-1 protease cleavage site prediction. *Pattern Recognition Letters* **27**, 1537-1544.

Nelson, D. L. and Cox, M. M. (2005). *Principles of Biochemistry*. W. H. Freeman and Company.

Narayanan, A., Wu, X. and Yang, Z. R. (2002). Mining viral protease data to extract cleavage knowledge. *Bioinformatics* **18** (Suppl.1), S5-S13.

Rögnvaldsson, T. and You, L. (2004). Why neural networks should not be used for HIV-1 protease cleavage site prediction. *Bioinformatics* **20**, 1702-1709.

Taylor, W. R. (1986). The classification of amino acid conservation. *Journal of Theoretical Biology 119*, 205-18.

Thompson, T., Chou, K, and Zheng, C. (1995). Neural network prediction of the HIV-1 protease cleavage sites. *Journal of Theoretical Biology* **177**, 369-379.

Taha, I. A. and Ghosh, J. (1999). Symbolic interpretation of artificial neural networks. *IEEE Trans. on Knowledge Data Eng.* **11**, 443-463.

Thomson, R., Hodgman, T. C., Yang, Z. R. and Doyle, A. K. (2003). Characterizing proteolytic cleavage site activity using bio-basis function neural networks. *Bioinformatics* **19**, 1741-1747.

Yang, Z. R., Dalby, A. and Qiu, J. (2004). Mining HIV protease cleavage data using genetic programming with a sum-product function. *Bioinformatics* **20**, 3398-3405.

You, L., Garwicz, D. and Rögnvaldsson, T. (2005) Comprehensive bioinformatic analysis of the specificity of human immunodeficiency virus type 1 protease. *Journal of Virology* **79**, 12477-12486.

Received September 12, 2008; accepted October 29, 2008.

Stefan Jaeger
Shanghai Institutes for Biological Sciences
CAS-MPG Partner Institute for Computational Biology
Shanghai 200031, China
jaeger@picb.ac.cn

Su-Shing Chen
Shanghai Institutes for Biological Sciences
CAS-MPG Partner Institute for Computational Biology
Shanghai 200031, China
suchen@cise.ufl.edu