Available online at www.sciencedirect.com

SCIENCE DIRECT®

BioSystems 72 (2003) 159–176

Bio Systems

www.elsevier.com/locate/biosystems

Searching for discrimination rules in protease proteolytic cleavage activity using genetic programming with a min-max scoring function

Zheng Rong Yang[a,*], Rebecca Thomson[b], T. Charles Hodgman[c], Jonathan Dry[d], Austin K. Doyle[e], Ajit Narayanan[a], XiKun Wu[f]

[a] School of Engineering and Computer Science, Exeter University, Northcote House The Queen's Drive, Exeter EX4 4QJ, UK
[b] Division of Structural Biology, Oxford University, Oxford, UK
[c] GlaxoSmithKline, Stevenage SG1 2NY, UK
[d] Astra Zeneca, Cancer Bioscience, Alderley Park, SK10 4TG, UK
[e] GlaxoSmithKline, Biopharmaceuticals Development, Beckenham BR3 3BS, UK
[f] Department of Statistics, Oxford University, Oxford, UK

Abstract

This paper presents an algorithm which is able to extract discriminant rules from oligopeptides for protease proteolytic cleavage activity prediction. The algorithm is developed using genetic programming. Three important components in the algorithm are a min-max scoring function, the reverse Polish notation (RPN) and the use of minimum description length. The min-max scoring function is developed using amino acid similarity matrices for measuring the similarity between an oligopeptide and a rule, which is a complex algebraic equation of amino acids rather than a simple pattern sequence. The Fisher ratio is then calculated on the scoring values using the class label associated with the oligopeptides. The discriminant ability of each rule can therefore be evaluated. The use of RPN makes the evolutionary operations simpler and therefore reduces the computational cost. To prevent overfitting, the concept of minimum description length is used to penalize over-complicated rules. A fitness function is therefore composed of the Fisher ratio and the use of minimum description length for an efficient evolutionary process. In the application to four protease datasets (Trypsin, Factor Xa, Hepatitis C Virus and HIV protease cleavage site prediction), our algorithm is superior to C5, a conventional method for deriving decision trees.
© 2003 Elsevier Ireland Ltd. All rights reserved.

Keywords: Amino acid similarity matrix; Genetic programming; The reverse Polish notation; Proteolytic cleavage analysis

1. Introduction

Evolutionary algorithms (EAs) are inspired by Darwin's theory about evolution: *survival of the fittest*. The basic principle of EAs is to solve complicated problems by mimicking a natural evolutionary process: selecting against the worst solutions and keeping the remaining solutions (Fogel et al., 1966; Holland, 1975; Goldberg, 1989). In genetic algorithms (GAs), a form of EA, each solution is encoded by a binary string called a chromosome. A pool of chromosomes are then used to represent a population of possible solutions to a problem. The artificial evolutionary operations such as copy, mutation, and crossover are then used to eliminate the worst solutions and promote the best solutions. Generation by generation, GA will end up with a number of optimal

* Corresponding author.
E-mail address: z.r.yang@ex.ac.uk (Z.R. Yang).

0303-2647/$ – see front matter © 2003 Elsevier Ireland Ltd. All rights reserved.
doi:10.1016/S0303-2647(03)00141-2

or approximate optimal solutions to the system under investigation. The most useful characteristics of all EAs are that the objective function does not need to be differentiable and multiple solutions can be found. Because of this, EAs can be used for the optimization of many complicated problems covering many areas.

Genetic programming (GP), another form of EA, can be used to evolve programs to perform certain tasks (Koza, 1992). In GP, a binary chromosome is replaced by a complicated tree structure, in which a solution is an algebraic equation of the input variables. GP has also been applied to many areas, for instance, chemical process modeling (Gao and Loney, 2001; McKay et al., 1997) and superplastic-damage (Lin et al., 2002). GP has advantages over GA for some specific tasks, such as the use of variable data structures. The most important reason for using GP is that complicated algebraic equations of amino acids, which are non-numerical variables in protein sequences, can be handled without much computational cost.

Proteases are widely distributed in nature, where they perform a variety of different functions. Bacteria produce proteases to degrade extracellular proteins. Viral proteases are essential to the life cycle of many viruses by cleaving the precursor molecules of their coat proteins. Higher organisms use proteases for food digestion, cleavage of signal peptides. Proteases are also used in protein chemistry, proteomics research, and biopharmaceutical manufacture, for tasks such as protein identification by peptide mass fingerprinting, protein fragmentation, protein domain separation and analysis, and identification of protease activity type (Cheronis and Repine, 1993). Prediction and characterization of proteolytic cleavage sites are of great importance, both for research into the action of naturally-occurring proteases and to help practical research techniques. There have been intensive studies in this area using different techniques. As reviewed by Chou (1996). However, "protein fingerprint" predictions produced from cleavage site rules and regular expressions can often be ambiguous because of their low accuracy. As one of the most important application areas, analyzing biological sequences (particularly the analysis of proteolytic cleavage activity) is still challenging for GP because of the non-numerical attributes in oligopeptides and lack of previous related research.

Artificial neural networks (ANNs) are a class of advanced computing algorithms capable of approximating universal non-linear functions and have been widely used for analyzing oligopeptide data. Learning algorithms in training ANNs for use with oligopeptide data have included back-propagation neural networks (BPNNs) (Rumelhart and McClelland, 1986), self-organising maps (SOM) (Kohonen, 1989) and recurrent neural networks (RNNs) (Elman, 1991; Frasconi and Gori, 1996). Application of BPNNs to human immunodeficiency virus (HIV) protease cleavage site prediction gave a prediction accuracy of approximately 92% (Cai and Chou, 1998; Yang, 2001; Narayanan et al., 2002). BPNNs resulted in about 65–70% accuracy for secondary structure prediction (Reczko, 1993; Baldi et al., 2000; Pollastri et al., 2002) and RNNs have improved this prediction accuracy to 75% (Baldi et al., 2000). A self-organising map (SOM) has also been used to identify motifs and families in the context of unsupervised learning (Arrigo et al., 1991). The most common encoding of amino acids is through the "distributed method", in which 20 binary bits are used to represent each amino acid, with just one "bit" set to 1 and all others to 0 (Qian and Sejnowski, 1988). There are some problems with this method, the most important of which is that the use of Euclidean space has no theoretic basis in biology or chemistry and may reduce model accuracy. The distance between every pair of different amino acids in this distributed method is $\sqrt{2}$. This conflicts with biological differences between amino acids, for which various distance matrices have been defined and validated (Dayhoff et al., 1978; Johnson and Overington, 1993).

We have therefore developed a new ANN learning algorithm which is able to recognize amino acids directly using biological similarity measurements. The algorithm is referred to as a "bio-basis function neural network" (BBFNN) (Thomson and Yang, 2002; Thomson et al., 2003). The algorithm has improved performance in both learning speed and prediction robustness for the analysis of protease oligopeptides.

Despite the initial success of the BBFNN, it was not conducive to knowledge extraction. Moreover, our early work with both HIV and Hepatitis C Virus (HCV) showed that conventional decision tree methods do not outperform ANNs. The prediction accuracy from C5 (http://www.rulequest.com/) is much lower than that from a neural net model (Narayanan et al., 2002). This paper therefore aims to extract discriminant rules for protease cleavage site prediction using GP (Koza, 1992; Banzhaf et al., 1998). A discriminant rule is not necessarily a widely used pattern sequence such as *SFXXXXIT*, where *X* stands for any amino acid. Instead, a discriminant rule is a complicated algebraic equation of the amino acids occurring at the different positions in oligopeptides. For instance, $bT = cK = +cR = +$ represents a specific rule, where b and c stand for the second (P_2) and the third position (P_3) in an oligopeptide, respectively, T, K and R are three amino acids (threonine, lysine, and arginine), $=$ and $+$ denote *equality* and *or* operation, respectively. The rule is interpreted as: $P_2 = T$ or $P_3 = K$ or $P_3 = R$. An linguistic interpretation of this rule is as follows:

if the second position in a sequence is threonine (T) or the third position is lysine (K) or the third position is arginine (R), then . . . .

Using this kind of representation for rules will make evolutionary operations convenient, which will be explained in Section 2.3.

We have applied this algorithm to four datasets: Trypsin, Factor Xa, HCV and HIV protease cleavage site prediction, with success. The next section will discuss the method and the third section will describe the simulation results.

2. Method

2.1. Problem specification

For the task of proteolytic cleavage activity analysis, each oligopeptide is denoted by a vector $\boldsymbol{x} \in \mathcal{C}^D$, where $\mathcal{C}$ is a set of all the 20 amino acids (hence $|\mathcal{C}| = 20$) and D is the number of positions in the oligopeptides for analysis. A dataset of oligopeptides is denoted by $\Omega \subset \{\mathcal{C}^D \times \{0, 1\}\}$, where $\{0, 1\}$ is a target set containing class labels. The dataset can be divided into two parts: Ω_A and Ω_B for two classes. For proteolytic cleavage activity prediction, Ω_A contains oligopeptides without cleavage sites while Ω_B cleaved oligopeptides.

A rule set is denoted by Γ, in which each rule is a tree structure. Fig. 1 gives an example of a simple structure, where c and e are the third and the fifth positions (P_3 and P_5) in an oligopeptide, respectively, A and P are two amino acids (alanine and proline) and

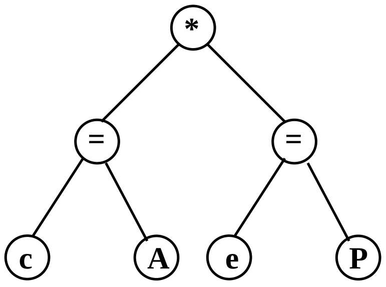

Fig. 1. A simple rule in a tree structure, where "c" and "e" are the third and the fifth positions in an oligopeptide, "A" and "P" are two amino acids, "=" and "$*$" mean the *equality* and *and* operation. The rule is therefore $P_c = A$ and $P_e = P$ meaning that two classes of oligopeptides can be well separated if the third and the fifth positions are occupied by the amino acids A and P, respectively.

$*$ is the *and* operation. The rule is then expressed as

$$P_3 = A \quad \text{and} \quad P_5 = P \tag{1}$$

A rule $\boldsymbol{r} \in \Gamma$ is defined as

$$\boldsymbol{r} = [[xy =]z] \tag{2}$$

where $[a]$ means that a can be repeated at maximum two times, $x \in \{a, b, c, \ldots, z\}$ indicates a position in an oligopeptide, $y \in \mathcal{C}$ represents an amino acid occurred at the position x and $z \in \{+, *\}$ denotes an operator. Note that $+$ and $*$ are the *or* and *and* operations.

2.2. Reverse Polish notation

A rule in the computer program is expressed using a RPN in this study similar to Kado et al. (1995). For instance, the rule in Fig. 1 is expressed as $cA = eP = *$. A reason for the use of RPN is to increase the ease of evolutionary reproduction operations. Without RPN, real tree structure would be more expensive computationally. For instance, crossover (see Section 2.3 for details) of two rules $cA = eP = *aF = +$ and $bF = dW = +fY = *$ at position of 7 produces two new rules $cA = eP = * \bigcup fY = \rightarrow cA = eP = *fY = *$ and $bF = dW = + \bigcup aF = + \rightarrow bF = dW = +aF = +$ with a little computational cost. We use $\mathcal{L}(\boldsymbol{r})$ to denote the length of a rule $\boldsymbol{r} \in \Gamma$ in a RPN and $\boldsymbol{r}[a : b]$ the subset of $\boldsymbol{r}$ from the position a to the position b in a RPN of the rule $\boldsymbol{r}$.

Table 1
RPN for rule expression

Rules	Expressions
(P_1 = A or P_6 = F or P_5 = P) and (P_7 = Y)	$aA = fF = +eP = +gY = *$
(P_1 = A or P_6 = F) and (P_5 = P or P_7 = Y)	$aA = fF = +eP = gY = +*$
(P_1 = A and P_6 = F) or (P_5 = P and P_7 = Y)	$aA = fF = *eP = gY = *+$

The left column shows three possible rules while the right column shows the corresponding RPN expressions.

In Table 1, there are some examples of using RPN to express rules. Suppose we have a rule as follows:

$$(P_1 = A \text{ or } P_6 = F \text{ or } P_5 = P) \quad \text{and} \quad P_7 = Y \tag{3}$$

a RPN for the rule is then $aA = fF = +eP = +gY = *$. Using a stack, this RPN can be easily processed (Fig. 2). The process is as in Table 2. For instance, the first token in RPN (a) is pushed into a stack since it is not an algebraic operator while the program pops two elements ("A" and "a") from the stack to form a rule, and then pushes the formulated rule ("$aA =$") into the stack when scanning the third token (=). Finally, the expression of the rule is as follows:

$$(P_1 = A \text{ or } P_6 = F \text{ or } P_5 = P) \quad \text{and} \quad P_7 = Y \tag{4}$$

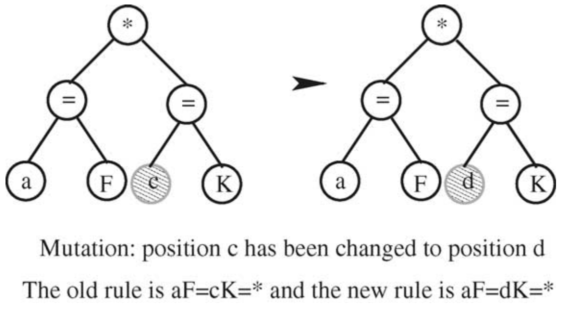
*
=
=
a
F
c
K
*
=
=
a
F
d
K
Mutation: position c has been changed to position d
The old rule is aF=cK=* and the new rule is aF=dK=*

Fig. 3. Mutation operation in GP. The node marked by a filled circle has changed its function from *or* operation to *and* operation after mutation is applied. Through a mutation, an old rule ($aF = cK = *$) has been used to reproduce a new rule ($aF = dK = *$).

2.3. Evolutionary operations

There are three common operators in GP: copy, mutation, and crossover. Fig. 3 shows a mutation operation while Fig. 4 shows a crossover operation. Suppose we have two rules $\boldsymbol{r}_m$ and $\boldsymbol{r}_f$ with their cutting points

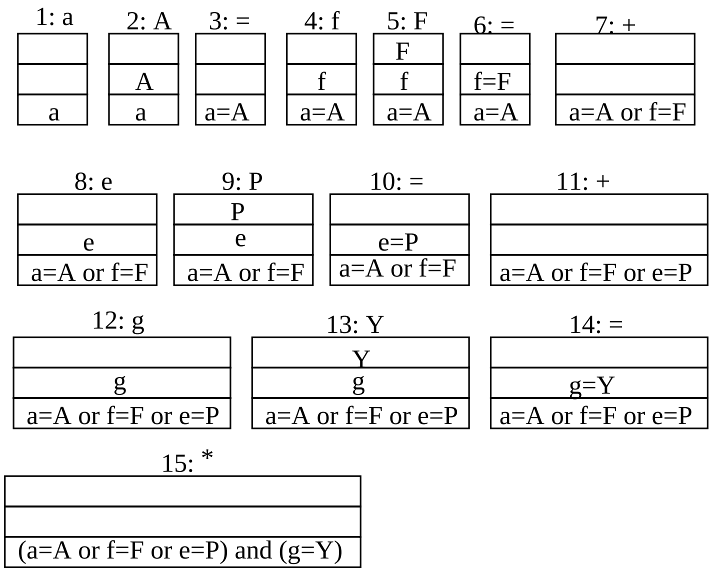

Fig. 2. Transformation of a RPN to a rule. A stack is expressed as a list of three boxes. The number before ":" is the step of scanning the RPN. The token after ":" is the token taken from the RPN. Whenever a token is taken from the RPN, a corresponding action is taken. The actions are to push into, pop from the stack and form a rule.

Table 2
RPN for rule recognition

Token	Actions			
a	*P*(*a*)			
A	*P*(*A*)			
=	*p*(*A*)	*p*(*a*)	*R*(*aA*=)	*P*(*aA*=)
f	*P*(*f*)			
F	*P*(*F*)			
=	*p*(*F*)	*p*(*f*)	*R*(*fF*=)	*P*(*fF*=)
+	*p*(*fF*=)	*p*(*aA*=)	*R*(*aA* = *fF* = +)	*P*(*aA* = *fF* = +)
e	*P*(*e*)			
P	*P*(*P*)			
=	*p*(*P*)	*p*(*e*)	*R*(*eP*=)	*P*(*eP*=)
+	*p*(*eP*=)	*p*(*aA* = *fF* = +)	*R*(*aA* = *fF* = +*eP* = +)	*P*(*aA* = *fF* = +*eP* = +)
g	*P*(*g*)			
Y	*P*(*Y*)			
=	*p*(*Y*)	*p*(*g*)	*R*(*gY*=)	*P*(*gY*=)
∗	*p*(*gY*=)	*p*(*aA* = *fF* = +*eP* = +)	*R*(*gY* = *aA* = *fF* = +*eP* = +∗)	*P*(*gY* = *aA* = *fF* = +*eP* = +∗)

15 steps is needed since there are 15 tokens in this RPN. $P(x)$ means that x is pushed into the stack. $p(x)$ means that x is popped from the stack. $R(x)$ means that a new rule x is formulated.

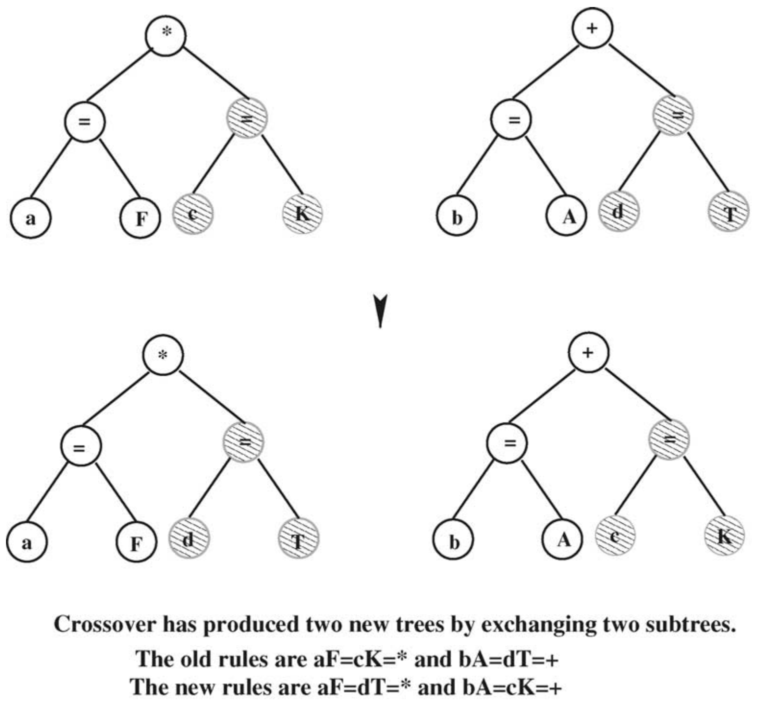
*
=
=
a
F
c
K
+
=
=
b
A
d
T
*
=
=
a
F
d
T
+
=
=
b
A
c
K
Crossover has produced two new trees by exchanging two subtrees.
The old rules are aF=cK=* and bA=dT=+
The new rules are aF=dT=* and bA=cK=+

Fig. 4. Crossover operation in GP. The sub-tree marked by the dark circles in left tree is exchanged with the sub-tree marked by the gray circles in the right tree after the crossover operation. Through a crossover, two old rules ($aF = cK = *$ and $bA = dT = +$) have been used to reproduce two new rules ($aF = dT = *$ and $bA = cK = +$).

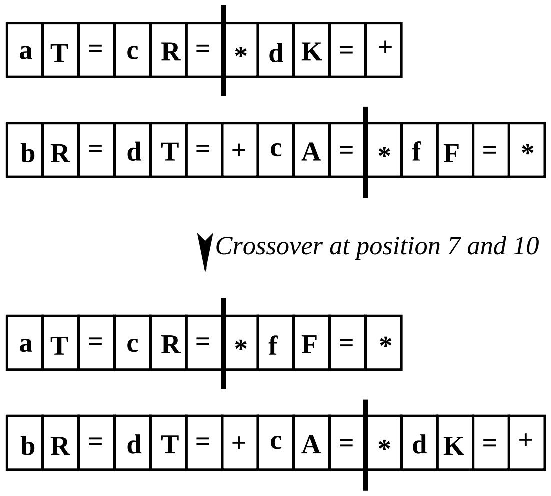
a T = c R = * d K = +
b R = d T = + c A = * f F = *
Crossover at position 7 and 10
a T = c R = * f F = *
b R = d T = + c A = * d K = +

Fig. 5. Crossover operation on two RPNs, where the cutting points are $c_m = 7$ and $c_f = 10$. Two old rules ($aT = cR = *dK = +$ and $bR = dT = +cA = *fF = *$) are used to reproduce two new rules ($aT = cR = *fF = *$ and $bR = dT = +cA = *dK = +$).

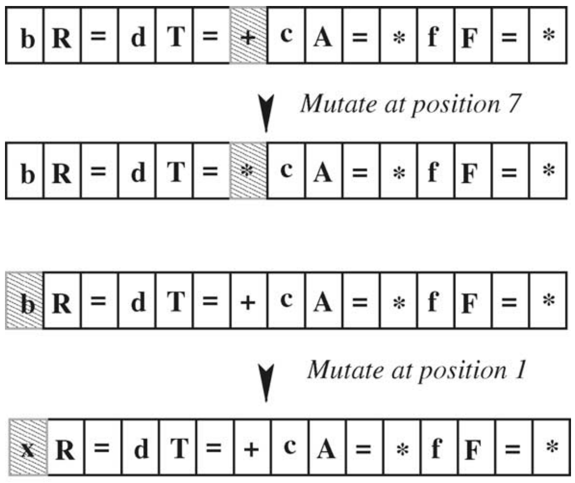
b R = d T = + c A = * f F = *
Mutate at position 7
b R = d T = * c A = * f F = *
b R = d T = + c A = * f F = *
Mutate at position 1
x R = d T = + c A = * f F = *

Fig. 6. Mutation operation on one RPN, where the cutting points are $c_m = 7$ and $c_m = 1$. An old rule ($bR = dT = +cA = *fF = *$) is used to reproduce two new rules ($bR = dT = *cA = *fF = *$ and $xR = dT = +cA = *fF = *$), where x means any amino acid.

in their RPNs (c_m and c_f) for mating, the crossover operation is defined as

$$\begin{aligned}\mathbf{r}_g &= \mathbf{r}_m[1 : c_m]\mathbf{r}_f[c_f + 1 : \mathcal{L}(\mathbf{r}_f)],\\ \mathbf{r}_b &= \mathbf{r}_f[1 : c_f]\mathbf{r}_m[c_m + 1 : \mathcal{L}(\mathbf{r}_m)]\end{aligned} \tag{5}$$

where $\mathbf{r}_g$ and $\mathbf{r}_b$ are two newly reproduced rules. Suppose we have two RPNs, $aT = cR = *dK+$ and $bR = dT = +cA * dK+$. Given two cutting points $c_m = 7$ and $c_f = 10$, two new RPNs will be $aT = cR = *dK+$ and $bR = dT = +cA * dK+$. The process for this is very simple and the time complexity is $\mathcal{O}(1)$. Fig. 5 shows this example.

Given a rule $\mathbf{r}_m$ with a cutting point in its RPN (c_m), the mutation operation is defined as

$$\mathbf{r}_g = \mathbf{r}_m[1 : c_m - 1]\pi(c_m)\mathbf{r}_m[c_m + 1 : \mathcal{L}(\mathbf{r}_m)] \tag{6}$$

where $\mathbf{r}_g$ is a newly reproduced rule and $\pi(\cdot)$ is a mutation function defined as

$$\pi(c_m) = \begin{cases} x = \mathcal{R}(\{a, b, c, \dots, z\}), \\ \quad \text{if } \mathbf{r}_m[c_m : c_m] \in \{a, b, c, \dots, z\} \\ y = \mathcal{R}(\mathcal{C}), \quad \text{if } \mathbf{r}_m[c_m : c_m] \in \mathcal{C} \\ z = \mathcal{R}(\{+, *\}), \quad \text{if } \mathbf{r}_m[c_m : c_m] \in \{+, *\} \end{cases} \tag{7}$$

where $\mathcal{R}(a)$ means a randomly selected element from the set a. Suppose we have one RPN, $bR = dT = +cA = *fF = *$. The new RPN can be $bR = dT = *cA = *fF = *$ if the cutting point is $c_m = 7$ while the new RPN is $xR = dT = +cA = *fF = *$, where x means any position in a sequence if the cutting point is $c_m = 1$. Fig. 6 shows this example.

2.4. Min-max scoring function

In order to evaluate a rule in terms of its ability to discriminate in an evolutionary process, the Fisher ratio was used. However, all the attributes in an oligopeptide are amino acids, which are non-numerical attributes. There is no direct way to measure the similarity between an oligopeptide and a rule, which is a complex algebraic equation of amino acids rather than a pattern sequence, for which a homology alignment can be applied without difficulty. As used in our previous study of BBFNN (Thomson and Yang, 2002; Thomson et al., 2003), the amino acid similarity matrices (Dayhoff et al., 1978; Johnson and Overington, 1993) are the solution. A min-max function is developed here to measure the similarity between an oligopeptide and a rule using amino acid similarity matrices in this study. The min-max function is as follows:

1. *and* operation:

$$s(\boldsymbol{x}_n \in \Omega, \boldsymbol{r}_m \in \Gamma) = \min\{g(\boldsymbol{x}_n(P_i), r_m(P_i)), g(\boldsymbol{x}_n(P_j), r_m(P_j))\} \tag{8}$$

2. *or* operation:

$$s(\boldsymbol{x}_n \in \Omega, \boldsymbol{r}_m \in \Gamma) = \max\{g(\boldsymbol{x}_n(P_i), r_m(P_i)), g(\boldsymbol{x}_n(P_j), r_m(P_j))\} \tag{9}$$

where $g(\cdot)$ performs a similarity measurement on a specific pair of amino acids (each from either an oligopeptide or a rule) using amino acid similarity matrix (Dayhoff et al., 1978; Johnson and Overington, 1993) while $s(\cdot)$ measures the similarity between the oligopeptide $\boldsymbol{x}_n$ and the discriminant rule $\boldsymbol{r}_m$.

2.5. Fitness function

The fitness function is composed of two parts: discriminant ability and rule complexity. The discriminant ability is quantified using the Fisher ratio:

$$J(\boldsymbol{r}_m) = \frac{|u_m^A - u_m^B|}{\sqrt{(\sigma_m^A)^2 + (\sigma_m^B)^2}} \tag{10}$$

where

$$u_m^A = \langle s(\boldsymbol{x}, \boldsymbol{r}_m) | \boldsymbol{x} \in \Omega_A \rangle$$

$$u_m^B = \langle s(\boldsymbol{x}, \boldsymbol{r}_m) | \boldsymbol{x} \in \Omega_B \rangle$$

$$(\sigma_m^A)^2 = \langle (s(\boldsymbol{x}, \boldsymbol{r}_m) - u_m^A)^2 | \boldsymbol{x} \in \Omega_A \rangle, \text{ and}$$

$$(\sigma_m^B)^2 = \langle (s(\boldsymbol{x}, \boldsymbol{r}_m) - u_m^B)^2 | \boldsymbol{x} \in \Omega_B \rangle$$

where $\langle\cdot\rangle$ means an expectation operation. Rule complexity is measured by minimum description length (Rissanen, 1978) and is defined as

$$\hat{\mathcal{L}}(\boldsymbol{r}_m) = \frac{\mathcal{L}(\boldsymbol{r}_m)}{\max\{\mathcal{L}(\boldsymbol{r}_k) | (\boldsymbol{r}_k) \in \Gamma\}} \tag{11}$$

where $\mathcal{L}(\boldsymbol{r}_m)$ is the equivalent length of the rule $\boldsymbol{r}_m$. The equivalent length is calculated by the recognition of the non-overlapped positions in a RPN. For instance, a RPN expressed as $aT = bK = bR = +$ has the equivalent length as 2 since only two positions in an oligopeptide are involved in decision-making. The fitness function is then defined as

$$\mathcal{F}(\boldsymbol{r}_m) = \frac{1}{1 + \exp(-J(\boldsymbol{r}_m) - (1 - \hat{\mathcal{L}}(\boldsymbol{r}_m))} \tag{12}$$

2.6. Evolutionary algorithm

Consider a space of M possible rules. In each evolutionary cycle, the best $M/2$ rules with the lowest fitness functions will be selected against with the remainder used for reproduction. Crossover and mutation operations were used to reproduce another $M/2$ new rules.

```
Step 1. Generate M rules, each of which has randomly selected positions and amino acids as well as the operators.
Step 2. Calculate the fitness function using Eq. (12). If the highest fitness function value is greater than a pre-determined threshold, goto Step 7, otherwise goto the next step.
Step 3. Sort all the rules in terms of their fitness function values from the highest to the lowest.
Step 4. Copy the top scoring M/2 rules to the next generation (the (t + 1)th generation).
Step 5. Randomly select an operation type: mutation or crossover:
    Mutation. Randomly select a rule and a cutting point from the (t + 1)th generation;
        If a token at the cutting point is a position token, mutate the position,
        If a token at the cutting point is an amino acid token, mutate the amino acid,
        If a token at the cutting point is an operator, mutate the operator,
        Insert this new produced rule to the (t + 1)th generation,
    Crossover. Randomly select a pair of rules called chromosome 𝒜 and chromosome ℬ from the (t + 1)th generation;
        Randomly select two cutting points from two rules,
        Merge the left part of the cutting point of 𝒜 and the right part of the cutting point of ℬ to produce a new chromosome (rule) 𝒳,
        Merge the left part of the cutting point of ℬ and the right part of the cutting point of 𝒜 to produce a new chromosome (rule) 𝒴,
```

Insert these two new rules ($\mathcal{X}$ and $\mathcal{Y}$) to the $(t+1)$th generation.

Step 6. Goto *Step 2*.

Step 7. Stop.

3. Results and analysis

3.1. Trypsin protease cleavage site prediction

Trypsin as a serine protease, is the most commonly used proteolytic enzyme in proteomics research. For example, it has been used to generate peptides from proteins for peptide mass fingerprinting, protein domain activity structural analysis, and for protein truncation in biopharmaceutical manufacture. Studies have shown that only the positions close to the cleavage site a significant influence on cleavage (Chou, 1996). The four most critical positions are denoted by P_3, P_2, P_1, $P_{1'}$, corresponding to the positions in trypsin S_3, S_2, S_1, $S_{1'}$ nomenclature (Schechter and Berger, 1968). Cleavage occurs between P_1 and $P_{1'}$.

Cleavage by trypsin occurs if a positively charged amino acid (lysine or arginine) is present in position P_1. If arginine is in position P_1, cleavage rates are much higher than if lysine is in P_1 (Wittinghofer et al., 1980; Barret et al., 1998). The negative impact of residues in positions P_2 and $P_{1'}$ was determined by Keil (1992).

Trypsin cleavage data was obtained from the literature (McRae et al., 1981; Kawabata et al., 1988; Pozsgay et al., 1981), and a set of non-cleaved tetrapeptides was created using Keil's rules (1992). All non-cleaved tetrapeptides were four residues long, including positions P_3–$P_{1'}$, as position $P_{1'}$ plays an important role in cleavage inhibition. All cleaved tetrapeptides obtained from papers showed positions P_3–P_1 only, therefore we expanded these tetrapeptides to include a residue in position $P_{1'}$.

Four hundred and thirteen tetrapeptides were used for training and testing. Among these tetrapeptides, 239 were cleaved tetrapeptides and the rest contained no cleavage sites. 350 tetrapeptides were used for training and the rest for testing. The population size was set to 1000 and the evolutionary cycles was 100. In each cycle, 50% offsprings were used for reproduction.

Table 3
Top 10 trypsin rules

No.	Rule	Fisher ratio	Testing accuracy (%)
1	$bT = cK = +cR = +$	2.5112	98
2	$cK = cN = +$	2.5013	98
3	$bH = cF = *cR = +$	2.5013	98
4	$cA = dT = *cR = +$	2.5013	98
5	$dT = cA = *cR = +$	2.5013	98
6	$cF = bH = *cR = +$	2.5013	98
7	$cF = bT = *cR = +$	2.4999	98
8	$cF = aQ = *cR = +$	2.4999	98
9	$cK = bT = +cR = +$	2.4997	98
10	$cF = bQ = *cK = +$	2.4997	98

We can see that $P_1 = R$ and $P_1 = K$ were the dominating sub-rules.

Table 3 lists the top 10 rules, where a, b, c, and d represent P_3, P_2, P_1, and $P_{1'}$, respectively. It was found that $P_1 = R$ and $P_1 = K$ were the dominating sub-rules. The rule with the highest fitness value is shown in Fig. 7. The Fisher ratio value and the testing accuracy on unseen trypsin tetrapeptides for the first rule were 2.0303 and 98%, respectively.

For the first rule $\boldsymbol{r}_1$, and an tetrapeptide $\boldsymbol{x}$, a scoring process using the min-max function is shown in Table 4. For the third rule $\boldsymbol{r}_3$, and an tetrapeptide $\boldsymbol{x}$, a scoring process using the min-max function is shown in Table 5. Note that $x(P_i)$ means the ith position in $\boldsymbol{x}$.

Table 6 shows the confusion matrix for the first rule. It can be seen that the predictive powers were 97% and 100% for the negavtive and positive classes. The total prediction accuracy was 98% on the unseen data. The false positive and false negative rates were 3% and 0%, respectively.

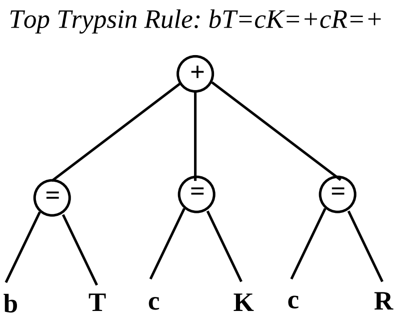

Fig. 7. The top scoring rule for the trypsin dataset. The rule was $P_2 = T$ *or* $P_1 = K$ *or* $P_1 = R$.

Table 4
Interpreting the first rule for the trypsin case

Scanning	Motif	Action
$x(P_2)$	T	Similarity $g_1 = s(x(P_2), T)$
$x(P_1)$	K	Similarity $g_2 = s(x(P_1), K)$
$x(P_1)$	R	Similarity $g_3 = s(x(P_1), R)$
Or		$s(\mathbf{x}, \mathbf{r}_1) = \max\{g_1, g_2, g_3\}$

For each sub-rule, such as *bT*=, a scanning process was needed to find the amino acid at a specified position in an oligopeptide ($\mathbf{x}$) under investigation. Similarity was calculated using Dayhoff matrix (Dayhoff et al., 1978).

Table 5
Interpreting the third rule for the trypsin case

Scanning	Motif	Action
$x(P_2)$	H	Similarity $g_1 = s(x(P_2), H)$
$x(P_1)$	F	Similarity $g_2 = s(x(P_1), F)$
And		$g_3 = \min\{g_1, g_2\}$
$x(P_1)$	R	Similarity $g_4 = s(x(P_1), R)$
Or		$s(\mathbf{x}, \mathbf{r}_3) = \max\{g_3, g_4\}$

It should be noted that the function of a discriminant rule was to separate the cleaved and the non-cleaved tetrapeptides. In other words, when the scoring values from two classes showed two well-separated distributions, the rule was regarded as having a good discriminant ability. For instance, the sub-rule $P_1 = R$ had a higher scoring values from the cleaved tetrapeptides than those from the non-cleaved tetrapeptides, while the sub-rule $P_1 = A$ had a higher scoring value from the non-cleaved tetrapeptides than those from the cleaved tetrapeptides. The reason was that few *Arginine* (R) residues occurred in the non-cleaved tetrapeptides while no *Alanine* (A) residues occurred in the cleaved tetrapeptides in the collected data. Table 7 shows these statistics.

Table 6
The confusion matrix of the best trypsin rule

	Negative	Positive	Percent (%)
Negative	31	1	97
Positive	0	31	100
Predictive power (%)	100	97	98

It can be seen that the predictive powers were 97% and 100% for the negative and positive classes. The total prediction accuracy was 98% on the unseen data. The false positive and false negative rates are 3% and 0%, respectively.

Table 7
An examination of the sub-rules in the trypsin case

Sub-rules	Non-cleaved	Cleaved
$P_1 = K$	8	178
$P_1 = R$	174	239
$P_1 = F$	10	0
$P_1 = A$	13	0
$P_1 = N$	6	0
$P_2 = H$	7	6
$P_{1'} = T$	13	16
$P_2 = T$	11	11

Each number represents the times that a specific amino acid in a specific position occurs in the oligopeptides.

By way of a comparison, C5 produced two different rules with different prediction accuracies. One of them, which had the highest accuracy (90%), is as follows:

$$\text{if } P_1 = K \text{ then cleave, otherwise not} \tag{13}$$

The second rule had an accuracy of 88% and is formulated as

$$\text{if } P_1 = R \text{ then cleave, otherwise not} \tag{14}$$

3.2. Factor Xa protease cleavage site prediction

Factor Xa is a serine protease. Earlier studies have shown that the positions close to the scissile bond have significant contribution to the cleavage activity. P_2, and P_1 and $P_{1'}$ are the most critical positions, with some influence shown at P_4 and P_3 positions (Bianchini et al., 2002). Cleavage is almost entirely dependent upon the presence of a positively charged residue arginine in the P_1 position (Harris et al., 2000). Most selection occurs at the P_2 position where there is strong preference for small non-polar amino acids such as glycine, or aromatic amino acids such as phenylalanine (McRae et al., 1981; Bianchini et al., 2002). Hydrophilic amino acids seem to be preferred at the P_3 position, and there is a minor preference for aliphatic P_4 and small $P_{1'}$ residues (McRae et al., 1981; Bianchini et al., 2002). Evidence suggests that cleavage is inhibited by the presence of a cysteine or proline residue at the P_1 and P_2 positions (Bianchini et al., 2002), or by the by the presence of bulky residues (valine, leucine, isoleucine, phenylalanine, tryptophan, tyrosine or histidine) at the $P_{1'}$ position (Harris et al., 2000). Factor Xa is integral to the clotting cascade in mammalian cells. Its presence in recombinant host-cells such as Chinese hamster ovary, however, has proven problematic to the biopharmaceutical industry. Proteolytic degradation of biopharmaceuticals before their recovery from the host-cell prevents their use as therapeutics (Murby et al., 1996). Locating cleavage sites within oligopeptides could enable improved stability via protein engineering (Flaschel and Friehs, 1993). To date, the PeptideCutter algorithm at ExPASy (http://www.expasy.ch) is the only tool available publicly for use in prediction of Factor Xa cleavage sites in proteins. This algorithm is based on sequence searches for matches to regular expressions for a vague primary sequence cleavage site motif of four amino acids, representing the P_1 to P_4 residues, with no consideration for effects of the $P_{1'}$ position. The data used to generate the regular expressions was collected from a limited number of substrates. These regular expressions prove inaccurate, predicting cleavage sites matching a vague motif that would not be cleaved, and also failing to locate cleavage motifs that exist outside of the motif. Such a tool can therefore not be used to predict Factor Xa cleavage sites for biopharmaceutical design with sufficient accuracy. These algorithms also give no indication as to the predicted rate of cleavage, enabling no analysis of which sites will be cleaved primarily when more than one cleavage motif is located in a protein. In this study 260 molecular oligopeptides, representing P_4 to $P_{1'}$ residues of known substrate cleavage sites, were used as a positive dataset to train, validate and test. A negative dataset of 251 oligopeptides was collected from experimentally verified non-cleavage sites, and expanded slightly with molecular oligopeptides generated using regular expressions from structurally accessible non-cleavage areas of known non-substrates and substrates.

In total, there were 481 oligopeptides, of which 122 were positive (with cleavage sites) and the rest were negative. 400 oligopeptides were for training and the rest were used for testing. The population size was set to 1000 and the evolutionary cycles was 100. Half of the offsprings were used for reproduction in each evolutionary iteration. Table 8 shows the top 10 rules, where a, b, c, d, and e represent P_4, P_3, P_2, P_1, and $P_{1'}$, respectively. Note that "="s were omitted to save space. It was found that $P_1 = K$, $P_1 = W$ and $P_1 = H$ were the dominating sub-rules. Fig. 8 shows the rule with the highest fitness value. Table 9 shows the confusion matrix for the factor dataset. It can be seen that the predictive powers were 93% and 71%, respectively. The total prediction accuracy on the unseen data was 86%. The false positive and false negative rates were 12% and 19%, respectively.

3.3. HCV protease cleavage site prediction

Identified by molecular cloning in 1989 (Choo et al., 1989; Kuo et al., 1989), the Hepatitis C Virus (HCV) is the major etiological agent of non-A and non-B hepatitis. Infection by HCV causes chronic liver disease, which is a serious health problem worldwide (Jenny-Avital, 1998). HCV-related liver disease may kill more people than AIDS (Cohen, 1999). There is currently no vaccine or effective therapy against

Table 8
Top 10 factor rules

No.	Rule	Fisher ratio	Testing accuracy (%)
1	$aVeS * bE + eA + eT + dW * aA + eA + dH*$	1.4221	86
2	$aVeS * bE + bD + eA + eT + dW * aA + dK*$	1.3203	86
3	$cGcY + eA + cY + aT + eT + dK * dH*$	1.3152	86
4	$cGcY + eA + cY + aT + eT + dK * dH*$	1.3152	86
5	$aVeS * bE + eA + eT + dW * aA + eA + dK*$	1.2824	86
6	$aVeS * bE + eA + eT + dW * aA + eA + dK*$	1.2824	86
7	$aVeS * bE + bD + eA + eT + dW * aA + eA + dK*$	1.2707	86
8	$cQcA * bV * cY + cL * eT + cF * eT + dK * dH*$	1.1969	81
9	$cQcI * bV * cY + cL * eT + cF * eT + dK * dH*$	1.1969	81
10	$cQcI * bP * cY + cL * eT + cF * eT + dK * dH*$	1.1969	81

We can see that $P_1 = K$, $P_1 = W$ and $P_1 = H$ were the dominating sub-rules.

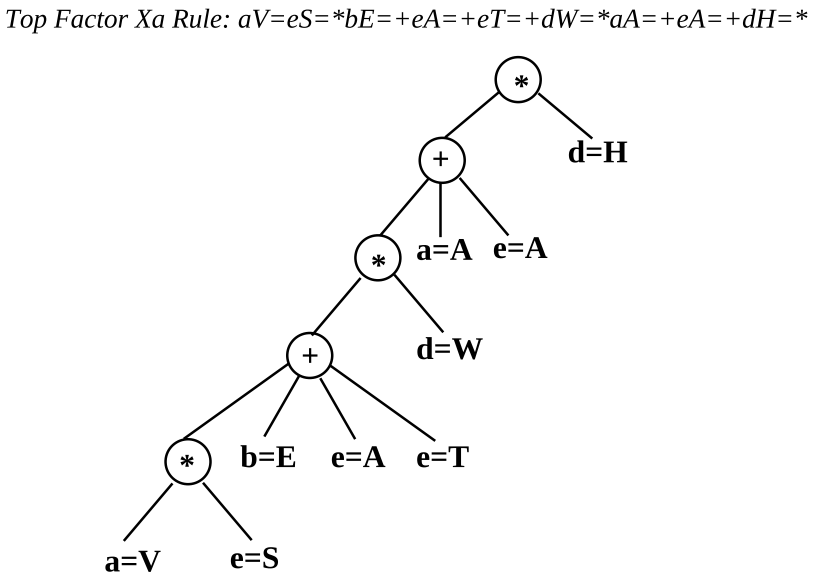

Fig. 8. The top scoring rule for the Factor Xa dataset. The rule was $((((P_4 = V \text{ and } P_{1'} = S) \text{ or } P_3 = E \text{ or } P_{1'} = A \text{ or } P_{1'} = T) \text{ and } P_1 = W) \text{ or } P_4 = A \text{ or } P_{1'} = A) \text{ and } P_1 = H$.

HCV, except through the use of general anti-viral agents which are not fully effective (Alter, 1997). As a member of the family Flaviviridae (Francki et al., 1991), HCV has a 9.5 kb single, positive-sense RNA genome. The N-terminus of the NS3 protein contains a serine protease which can perform four out of five cleavage events in the non-structural region: NS3/NS4A, NS4A/NS4B, NS4B/NS5A, and NS5A/NS5B (Bartenschlager et al., 1995; Eckard et al., 1993; Hijikata et al., 1993; Komoda et al., 1994; Tomei et al., 1993). Inhibition of the NS3 protease activity leads to the production of non-infectious viral particles (Chambers et al., 1990), and NS3 protease has therefore become one of the main targets for anti-HCV drugs.

Table 9
The confusion matrix of the best factor rule

	Negative	Positive	Percent (%)
Negative	53	7	88
Positive	4	17	81
Predictive power (%)	93	71	86

It can be seen that the predictive powers were 93% and 71%, respectively. The total prediction accuracy on the unseen data was 86%. The false positive and false negative rates are 12% and 19%, respectively.

Due to the flat and featureless substrate binding sites, NS3 requires a relatively long (at least decamer) peptide substrate (Steinkhler et al., 1996; Urbani et al., 1997; Zhang et al., 1997). The nomenclature of Schechter and Berger (1968) is used to designate the cleavage sites on the polyproteins, P_6, P_5, P_4, P_3, P_2, P_1, $P_{1'}$, $P_{2'}$, $P_{3'}$, $P_{4'}$ (P oligopeptide), the asymmetric scissile bond being between P_1 and $P_{1'}$. The binding sites on the enzyme corresponding to residues P_6, P_5, P_4, P_3, P_2, P_1, $P_{1'}$, $P_{2'}$, $P_{3'}$, $P_{4'}$ are indicated as S_6, S_5, S_4, S_3, S_2, S_1, $S_{1'}$, $S_{2'}$, $S_{3'}$, $S_{4'}$ (S oligopeptide). Unlike serine protease such as trypsin and chymotrypsin that can cleave small peptide substrates, the NS3 protease has an unusual substrate recognition mechanism, which makes it difficult as a target for small molecule inhibitors. Also due to the number of possible decapeptides formed from 20 amino acids (20^{10}), to test them one by one in the laboratory is impossible. Instead, a restricted number of oligopeptides have been obtained and the behavior of NS3 on these observed and recorded. The pattern recognition task is to use this restricted set to explore patterns in the P oligopeptide for which the S oligopeptide locks on, to express these patterns in such a way that allows generalizations to be made, and therefore to predict the behavior of NS3 on all

Table 10
Top 10 HCV rules

No.	Rule	Fisher ratio	Testing accuracy (%)
1	$jFgReI * fH + fR * hW * iW + +aD * fC +$	1.8726	93
2	$jFgRcI * fH * hF * iW + +aD * fC +$	1.8726	93
3	$jFgReI * fH * hF * iW + +aD * fC +$	1.8726	93
4	$jFgReI * fH * hF * iW + +aD * fC +$	1.8718	93
5	$jYfYiC * jF * gS * iW + +aD * fC +$	1.8718	93
6	$jYfYbC * jF * gS * iW + +aD * fC +$	1.8598	93
7	$jYfYbC * hF * iW + +aD * fC +$	1.8598	93
8	$jYfYbC * jM * gS * iW + +aD * fC +$	1.8548	93
9	$jYfIbC * hF * iW + +aD * fC +$	1.8548	93
10	$jYfYbC + hF * iW + +aD * fC +$	1.8110	93

We can see that $P_1 = C$, $P_6 = D$, and $P_{3'} = W$ were the dominatant sub-rules.

the remaining oligopeptides. Apart from our own preliminary studies, there has been little attempt to use pattern recognition techniques to predict HCV polyprotein cleavability through the recognition of patterns in P oligopeptides.

We have collected 920 oligopeptides, of which 168 were positive (with cleavage sites) and the rest were negative. We used 800 oligopeptides for training and 120 for testing. The population size was set to 1000 with 100 evolutionary cycles. We also used 50% offsprings for reproduction in each evolutionary iteration. Table 10 listed top 10 rules, where a, b, c, d, e, f, g, h, i, and j imply P_6, P_5, P_4, P_3, P_2, P_1, $P_{1'}$, $P_{2'}$, $P_{3'}$, and $P_{4'}$, respectively. Note that "="s were omitted for saving the space. It has been found that $P_1 = C$, $P_6 = D$, and $P_{3'} = W$ were the dominatant sub-rules. Fig. 9 shows the rule with the highest fitness value. Table 11 showed the confusion matrix for the HCV dataset. It can be seen that the predictive powers were 98% and 72%, respectively. The total prediction

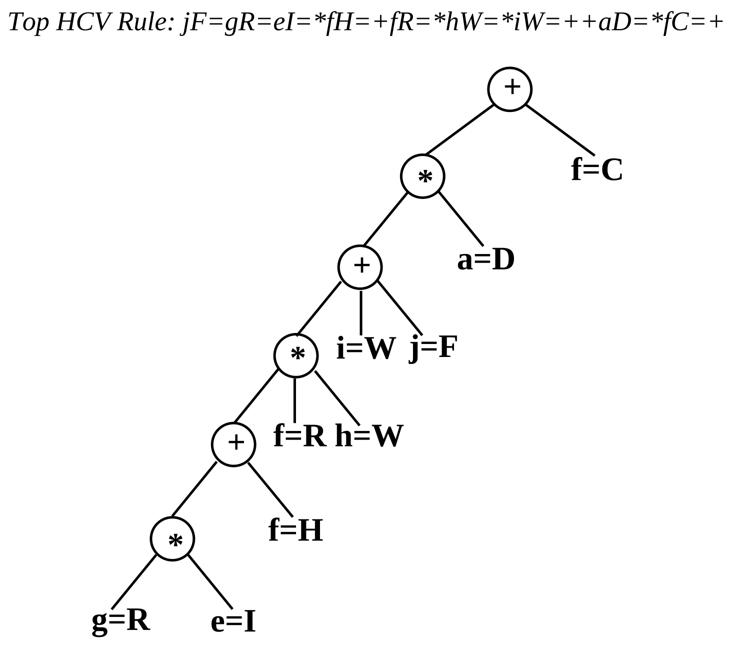

Fig. 9. The top scoring rule for the HCV dataset. The rule was ((((($P_{1'} = R$ and $P_2 = I$) or $P_1 = H$) and $P_1 = R$ and $P_{2'} = W$) or $P_{3'} = W$ or $P_{4'} = F$) and $P_6 = D$) or $P_1 = C$.

Table 11
The confusion matrix of the best HCV rule

	Negative	Positive	Percent (%)
Negative	100	5	95
Positive	2	13	87
Predictive power (%)	98	72	94

It can be seen that the predictive powers were 98% and 72%, respectively. The total prediction accuracy on the unseen data was 94%. The false positive and false negative rates are 5% and 13%, respectively.

accuracy on the unseen data was 94%. The false positive and false negative rates were 5% and 13%, respectively.

3.4. HIV protease cleavage site prediction

Proteolytic cleavage is an essential component of the HIV life cycle (Chou, 1996; Poorman et al., 1991; Ridly, 1996). HIV-1 protease has a recognized specificity consisting of eight positions around the cleavage site (Chou, 1996). The knowledge of the cleavage sites can be used in the development of antiviral drugs. The eight positions around cleavage sites in HIV octapeptides are denoted by P_4, P_3, P_2, P_1, $P_{1'}$, $P_{2'}$, $P_{3'}$ and $P_{4'}$ each corresponding to the eight positions in HIV protease S_4, S_3, S_2, S_1, $S_{1'}$, $S_{2'}$, $S_{3'}$ and $S_{4'}$. The cleavage occurs between P_1 and $P_{1'}$. HIV-1 protease is highly substrate and cleavage specific (Chou, 1996) and is essential for the replication and maturation of HIV. Cleavage often occurs at the site of defined pairs of large, virus-specific polypeptides (Hellen et al., 1989). There have been studies in designing effective HIV protease inhibitor (Ashorn et al., 1990). In order to design effective HIV protease inhibitors, there have been many algorithms designed for predicting the cleavage sites of HIV polyprotein, such as the h function (Poorman et al., 1991), Markov chain models (Chou, 1996), ANNs (Cai and Chou, 1998; Narayanan et al., 2002) and a binary probabilistic model (Yang, 2001). For a thorough review, see Chou (1996).

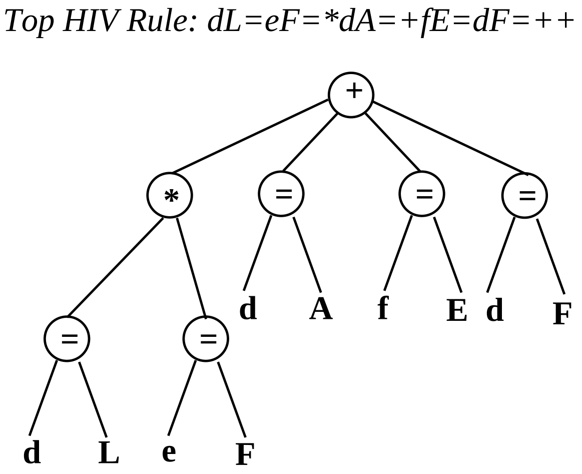

Fig. 10. The top scoring rule for the HIV dataset. The rule was $(P_1 = L \textit{ and } P_{1'} = F) \textit{ or } P_1 = A \textit{ or } P_{2'} = E \textit{ or } P_1 = F$.

Three hundred and sixty-two HIV oligopeptides were collected from (Cai and Chou, 1998), of which 114 were positive (with cleavage sites) and the rest were negative. We used 300 oligopeptides for training and 62 for testing. The population size was set to 1000

Table 12
Top 10 HIV rules

No.	Rule	Fisher ratio	Testing accuracy (%)
1	$dL = eF = *dA = +fE = dF = ++$	1.3261	90
2	$dL = eF = *dF = +fE = dN = ++dF = +$	1.3260	90
3	$dL = eF = *dT = +fE = dF = ++dF = +$	1.3260	90
4	$dL = eF = *dF = +fE = dF = ++$	1.3260	90
5	$dL = eF = *fE = dF = ++$	1.3258	90
6	$dL = eF = *gA = +fE = dT = ++dF = +$	1.3258	90
7	$dL = eF = *gS = +fE = dF = ++$	1.3253	90
8	$dL = eF = *hA = +fE = dF = ++$	1.3253	90
9	$dL = eF = *dF = +fE = dY = ++dF = +$	1.3253	90
10	$dL = eY = *dF = +fE = dY = ++$	1.3253	90

We can see that $P_1 = L$, $P_{2'} = E$, and $P_1 = F$ were the dominating sub-rules.

with 100 evolutionary cycles. Again, we used 50% offsprings for reproduction in each evolutionary cycle. Table 12 lists the top 10 rules, where a, b, c, d, e, f, g, and h represent P_4, P_3, P_2, P_1, $P_{1'}$, $P_{2'}$, $P_{3'}$, and $P_{4'}$, respectively. It can be seen that $P_1 = L$, $P_{2'} = E$, and $P_1 = F$ were the dominating sub-rules. Fig. 10 shows the rule with the highest fitness value. Table 13 shows the confusion matrix for the HIV dataset. It can be seen that the predictive powers were 100% and 86%, respectively. The total prediction accuracy on the unseen data was 95%. The false positive and false negative rates were 7% and 0%, respectively.

Table 13
The confusion matrix of the best HIV rule

	Negative	Positive	Percent (%)
Negative	40	3	93
Positive	0	19	100
Predictive power (%)	100	86	95

It can be seen that the predictive powers were 100% and 86%, respectively. The total prediction accuracy on the unseen data was 95%. The false positive and false negative rates are 7% and 0%, respectively.

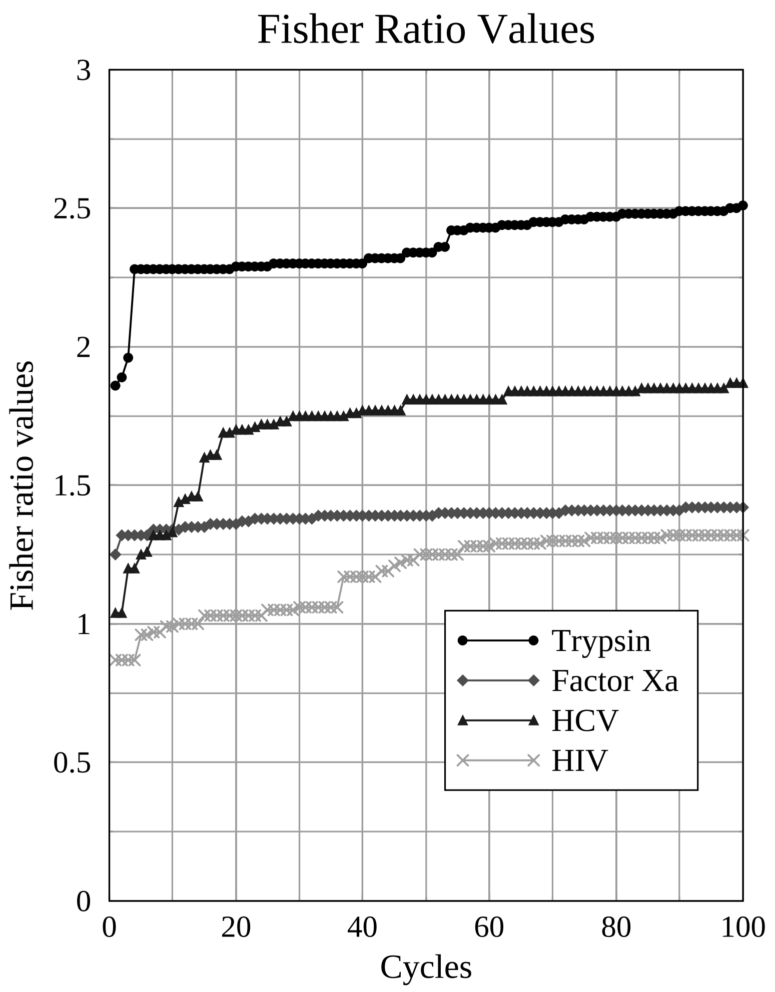

Fig. 11. The Fisher ratio values in the evolutionary cycles. The evolutionary cycles were 100 for all the datasets. They all approached a stable state within 100 evolutionary cycles.

Table 14
The GP parameters used for the four datasets

	Popsize	Iterations	CPU	Selection (%)	Total	Training
Trypsin	1000	100	1740	50	413	63
Factor Xa	1000	100	4361	50	481	81
HCV	1000	100	6218	50	920	120
HIV	1000	100	3107	50	362	62

"Popsize" is the number of offsprings used in a pool, "iterations" is the longest allowed simulation cycles, "CPU" is the CPU time in seconds, "selection" is the percentage of the offsprings used for reproduction, "total" is the total number of oligopeptides and "training" is the number of oligopeptides used for testing.

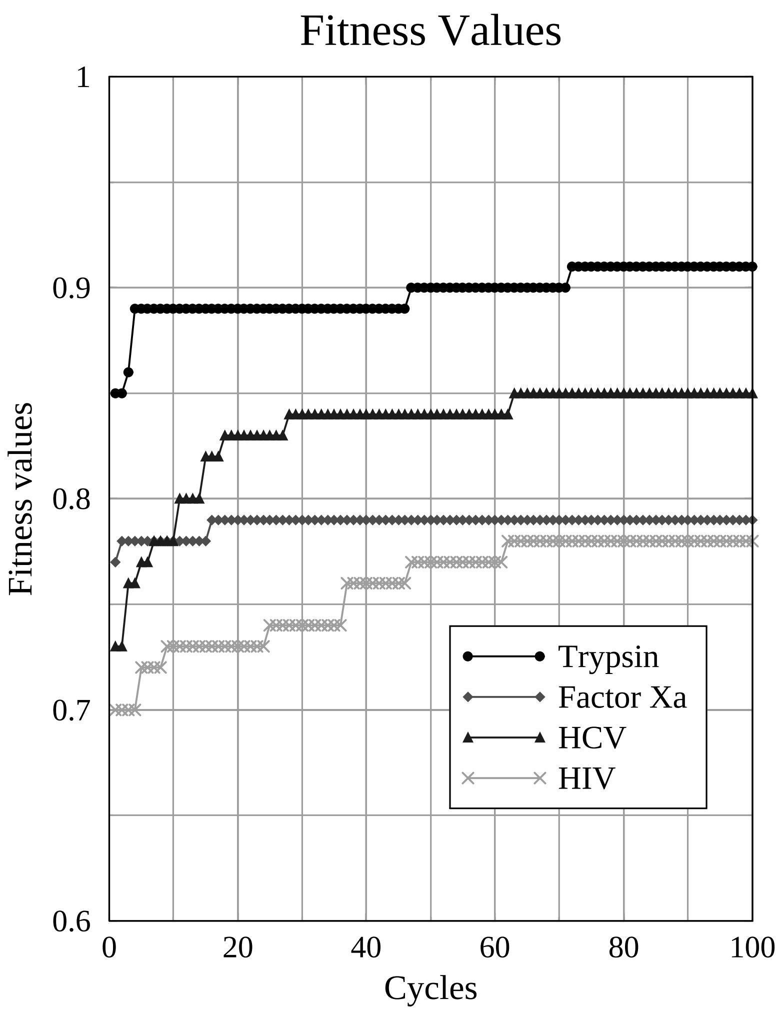

Fig. 12. The fitness values in the evolutionary cycles. The evolutionary cycles were 100 for all the datasets. They all approached a stable state within 100 evolutionary cycles.

3.5. The parameters used in the experiments

The experimental parameters on all four datasets are listed in Table 14. The population size was set to 1000 to ensure enough diversity during an evolutionary process. For each parent in each evolutionary cycle, crossover or mutation operation was determined randomly. Fig. 11 shows the Fisher ratio values through the evolutionary cycles and Fig. 12 the fitness values through the evolutionary cycles. It can be seen that the fitness functions for all four experiments were approaching stable values within 100 evolutionary cycles.

3.6. A comparison among four methods

Table 15 gives a comparison between our algorithm, C5, and ANNs. It can be seen that our algorithm was superior to the conventional decision tree method. For instance, GP's prediction accuracies for the trypsin, HCV and HIV cases were higher than C5. In our experiments, GP was also comparable with ANNs. In the trypsin case, the accuracy from GP was 98% compared with 90% from C5. In the HCV case, the accuracy from GP was 93% compared with 91% from C5. In the HIV case, the accuracy from GP was 95% compared with 85% from C5. We did not manage to complete the C5 experiment for the Factor Xa dataset before the license expired. For the Factor Xa case, ANNs did not work as well as GP. This phenomenon may arise from the fact that there were many "*X*" (unknown amino acids) in the dataset, which caused ANNs difficulty. The full cause why ANNs did not work well is still under the investigation.

Table 15
Comparison among different methods

	BPNN (%)	BBFNN (%)	C5 (%)	GP (%)
Trypsin	94	99	90	98
Factor Xa	75	77	–	86
HCV	96	97	91	94
HIV	90	93	85	95

The BPNN model was based on the distributed encoding method (Qian and Sejnowski, 1988) and using 20 hidden neurons. The BBFNN model used 300 bio-basis functions. GP model was based on a population size of 1000 with 100 iterations.

4. Summary and future work

Having understood that proteins have their own language (Benner and Gaucher, 2001), and that existing rule extraction tools do not outperform ANNs, we have presented a new method for extracting rules from oligopeptides using GP. Three contributions have been made towards a robust, reliable, and accurate algorithm. The first is the use of the reverse Polish notation, which makes the evolutionary operation much easier. The second is the use of minimum description length, which penalizes over-complicated rules so that possible overfitting can be avoided. The third is that a min-max function is developed to measure the similarity between an oligopeptide and a rule, which is a complex algebraic equation of amino acids. The Fisher ratio is then used for evaluating the discriminant ability of rules with the assistant of the widely used amino acid similarity matrices. The first two have made the task of knowledge extraction simpler and better in generalization. The last one is critically important since it avoids possible bias in recognizing non-numerical attributes (amino acids) and therefore enhances the robustness of the process of knowledge extraction. In the application to four proteolytic activity datasets, we have shown that our algorithm is superior to the conventional decision tree method and is comparable to ANN models. Future work will focus on the scoring functions other than min-max and the integration of multiple rules using the Bayes rule.

Acknowledgements

The authors thank the anonymous reviewers for their extremely valuable comments.

References

Alter, M.J., 1997. Epidemiology of Hepatitis C. Hepatology 26, 62S–65S.

Arrigo, P., Giuliano, F., Scalia, F., Rapallo, A., Damiani, G., 1991. Identification of a new motif on nucleic acid sequence data using Kohonen's self-organising map. CABIOS 7, 353–357.

Ashorn, P., McQuade, T.J., Thaisrivongs, S., Tomasselli, A.G., Tarpley, W.G., Moss, B., 1990. An inhibitor of the protease blocks maturation of human and simian immunodeficiency viruses and spread of infection. Proc. Natl. Acad. Sci. U.S.A. 87, 7472–7476.

Baldi, P., Pollastri, G., Andersen, C.A., Brunak, S., 2000. Matching protein beta-sheet partners by feedforward and recurrent neural networks. In: Proceedings of the International Conference on Intelligent Systems for Molecular Biology, vol. 8, pp. 25–36.

Banzhaf, W., Nordin, P., Keller, R.E., Francone, F.D., 1998. Genetic Programming, An Introduction. Morgan Kaufmann, Los Altos.

Barret, A., Rawlings, N.D., Woessner, J.F., 1998. Handbook of Proteolytic Enzymes. Academic Press, New York.

Bartenschlager, R., Ahlborn-Laake, L., Yasargil, K., Mous, J., Jacobsen, H., 1995. Substrate determinants for cleavage in *cis* and in *trans* by the Hepatitis C Virus NS3 protease. J. Virol. 69, 198–205.

Benner, S.A., Gaucher, E.A., 2001. Evolution, language, and analogy in functional genomics. Trends Genet. 17, 414–418.

Bianchini, E.P., Louvain, V.B., Marque, P.E., Juliano, M.A., Juliano, V., Le Bonniec, B.F., 2002. Mapping of the catalytic groove preferences of Factor Xa reveals an inadequate selectivity for its macromolecule substrates. J. Biol. Chem. Mar 29.

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

Chambers, T.J., Weir, R.C., Grakoui, A., McCourt, D.W., Bazan, F.F., Fletterick, R.J., Rice, C.M., 1990. Evidence that the N-terminal domain of nonstructural protein NS3 from Yellow Fever Virus is a serine proteins responsible for site-specific cleavages in the viral polyprotein. Proc. Natl. Acad. Sci. U.S.A. 87, 8898–8902.

Cheronis, J.C., Repine, J.E., 1993. Proteases, Protease Inhibitors and Protease-Derived Peptides. Birkhauser Verlag, Berlin.

Choo, Q.L., Kuo, G., Weiner, A.J., Overby, L.R., Bradley, D.W., Houghton, M., 1989. Isolation of a cDNA clone derived from a blood-borne non-A non-B viral hepatitis genome. Science 244, 359–362.

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

Cohen, J., 1999. The scientific challenge of Hepatitis C. Science 285, 26–30.

Dayhoff, M.O., Schwartz, R.M., Orcutt, B.C., 1978. A model of evolutionary change in proteins. matrices for detecting distant relationships. In: Dayhoff, M.O. (Ed.), Atlas of Protein Sequence and Structure, vol. 5. National Biomedical Research Foundation, Washington, DC, pp. 345–358.

Eckard, M.R., Selby, M., Masiarz, F., Lee, C., Berger, K., Crawford, K., Kuo, C., Kuo, G., Houghton, M., Choo, Q.L., 1993. The Hepatitis C Virus encodes a serine protease involved in processing of the putative nonstructural proteins from the viral polyprotein precursor. Biochem. Biophys. Res. Commun. 192, 399–406.

Elman, J.L., 1991. Distributed representations, simple recurrent networks, and grammatical structure. Machine Learn. 7, 195–225.

Flaschel, E., Friehs, K., 1993. Improvement of downstream processing of recombinant proteins by means of genetic engineering methods. Biotech. Adv. 11, 31–78.

Fogel, L.J., Owens, A.J., Walsh, M.J., 1966. Artificial Intelligence through Simulated Evolution. Wiley, New York.

Francki, R.I.B., Fauquet, C.M., Knudson, D.L., Brown, F., 1991. Classification and nomenclature of virus. Fifth Report of the International Committee on Taxonomy of Viruses. Arch. Virol. 2 (Suppl.), 223.

Frasconi, P., Gori, M., 1996. Computational capabilities of local-feedback recurrent networks acting as finite-state machines. IEEE Trans. Neural Netw. 7, 1521–1524.

Gao, L., Loney, N.W., 2001. Evolutionary polymorphic neural network in chemical process modelling. Comput. Chem. Eng. 25, 1403–1410.

Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading, MA.

Harris, J.L., Backes, B.J., Leonetti, F., Mahrus, S., Ellman, J.A., Craik, C.S., 2000. Rapid and general profiling of protease specificity by using combinatorial fluorogenic substrate libraries. Proc. Natl Acad. Sci. U.S.A. 97, 7754–7759.

Hellen, C.U.T., Krausslich, H.G., Wimmer, E., 1989. Proteolytic processing of polyproteins in the replication of RNA viruses. Biochemistry 28, 9881–9890.

Hijikata, M., Mizushima, H., Tanji, Y., Komoda, Y., Hirowatari, Y., Akagi, T., Kato, N., Kimura, K., Shimotohno, K., 1993. Proteolytic processing and membrane association of putative nonstructural proteins of Hepatitis C Virus. Proc. Natl. Acad. Sci. U.S.A. 90, 10773–10777.

Holland, H., 1975. Adaptation in Natural and Artificial Systems.

Jenny-Avital, E.R., 1998. Hepatitis C. Curr. Opin. Infect. Dis. 11, 293–299.

Johnson, M.S., Overington, J.P., 1993. A structural basis for sequence comparisons—an evaluation of scoring methodologies. J. Mol. Biol. 233, 716–738.

Kado, K., Ross, P.M., Corne, D., 1995. In: Eshelman (Ed.), Proceedings of the Sixth International Conference on Investigating Genetic Algorithms for Facility Layout Problems in Genetic Algorithms. Morgan Kaufmann, Los Altos.

Kawabata, S., Miura, T., Morita, T., Kato, H., Fujikawa, K., Sadaaki, I., Takada, K., Kimura, T., Sakakibara, S., 1988. Highly sensitive peptide-4-methylcoumaryl-7-amide substrates for blood-clotting proteases and trypsin. Eur. J. Biochem. 172, 17–25.

Keil, B., 1992. Specificity of Proteolysis. Springer, Berlin, pp. 66–69.

Kohonen, T., 1989. Self Organization and Associative Memory, 3rd ed. Springer, Berlin.

Komoda, Y., Hijikata, M., Tanji, Y., Hirowatari, Y., Mizushima, H., Kimura, K., Shimotohno, K., 1994. Processing of Hepatitis C viral polyprotein in *Escherichia coli*. Gene 145, 221–226.

Koza, J.R., 1992. Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge.

Kuo, G., Choo, Q., Alter, H.J., Gitnick, G.L., Redecker, A.G., Purcell, R.H., Myamura, T., Dienstag, J.L., Alter, M.J., Syevens, C.E., Tagtmeyer, G.E., Bonino, F., Colombo, M., Lee, W.S., Kuo, C., Berger, K., Shister, J.R., Overby, L.R., Bradley, D.W., Houghton, M., 1989. An assay for circulating antibodies to a major etiologic virus of human non-A non-B hepatitis. Science 244, 362–364.

Lin, J., Cheong, B.H., Yao, X., 2002. Universal multi-objective function for optimising superplastic-damage constitutive equations. J. Mater. Process. Technol. 125, 199–205.

McKay, B., Willis, M., Barton, G., 1997. Steady-state modelling of chemical process systems using genetic programming. Comput. Chem. Eng. 21, 981–996.

McRae, B.J., Kurachi, K., Helmark, R.L., Fujikawa, K., Davie, E.W., Powers, J.C., 1981. Mapping the active sites of bovine thrombin, Factor IXa, Factor Xa, Factor XIa, Factor XIIa, plasma kallikrein and trypsin with amino acid and peptide thioesters: development of new sensitive substrates. Biochemistry 20, 7196–7206.

Murby, M., Uhlen, M., Stahl, S., 1996. Upstream strategies to minimize proteolytic degradation upon recombinant production in *Escherichia coli*. Protein Expr. Purif. 7, 129–136.

Narayanan, A., Wu, X.K., Yang, Z.R., 2002. Mining viral protease data to extract cleavage knowledge. Bioinformatics 18, 1–18.

Pollastri, G., Przybylski, D., Rost, B., Baldi, P., 2002. Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles. Proteins 47, 228–235.

Poorman, R.A., Tomasselli, A.G., Heinrikson, R.L., Kezdy, F.J., 1991. A cumulative specificity model for protease from human immunodeficiency virus types 1 and 2, inferred from statistical analysis of an extended substrate data base. J. Biol. Chem. 22, 14554–14561.

Pozsgay, M., Szbabo, G.C.S., Bajusz, S., Simonsson, R., Gaspar, R., Elodi, P., 1981. Investigation of the substrate-binding site of trypsin by the aid of tripeptidyl-*p*-nitroanilide substrates. Eur. J. Biochem. 115, 497–502.

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

Reczko, M., 1993. Protein secondary structure prediction with partially recurrent neural networks. SAR and QSAR in Environmental Research 1, 153–159.

Ridly, T.W., 1996. Human immunodeficiency virus, type 1 protease substrate specificity is limited by interactions between substrate amino acids bond in adjacent enzyme. J. Biol. Chem. 271, 4709–4717.

Rissanen, J., 1978. Modeling by shortest data description. Automatica 14, 465–471.

Rumelhart, D.E., McClelland, J.L., 1986. Parallel Distributed Processing: Exploration in the Cognition. MIT Press, Cambridge, MA.

Schechter, I., Berger, A., 1968. On the active site of proteases. 3. Mapping the active site of papain; specific peptide inhibitors of papain. Biochem. Biophys. Res. Commun. 32, 898.

Steinkhler, C., Urbani, A., Tomei, L., Bisdiol, G., Sardana, M., Bianchi, E., Pessi, A., De Francesco, R., 1996. Activity of purified Hepatitis C Virus protease NS3 on peptide substrates. J. Virol. 70, 6694–6700.

Thomson, R., Yang, Z.R., 2002. A novel bio-basis function neural network. ICONIP02.

Thomson, R., Hodgman, T.C., Yang, Z.R., Austin K.D., 2003. Characterising proteolytic cleavage site activity using bio-basis function neural networks. Bioinformatics, in press.

Tomei, L., Failla, C., Santolini, E., De Francesco, R., La Monica, N., 1993. NS3 is a serine protease required for processing of Hepatitis C Virus polyprotein. J. Virol. 67, 4017–4026.

Urbani, A., Bianchi, E., Narjes, F., Tramontano, A., De Francesco, R., Steinkhler, C., Pessi, A., 1997. Substrate specificity of the Hepatitis C Virus serine protease NS3. J. Biol. Chem. 272, 9204–9209.

Wittinghofer, A., Frank, R., Leberman, R., 1980. Composition and properties of trypsin-like elongation factor Tu. Eur. J. Biochem. 108, 423–431.

Yang, Z.R., 2001. A binary probabilistic model and genetic algorithm for HIV protease cleavage sites prediction and search. ICONIP01.

Zhang, R., Durkin, J., Windsor, W.T., McNemar, C., Ramanathan, L., Le, H.V., 1997. Probing the substrate specificity of Hepatitis C Virus NS3 serine protease by using synthetic peptides. J. Virol. 71, 6208–6213.