Hindawi Publishing Corporation
EURASIP Journal on Applied Signal Processing
Volume 2006, Article ID 17195, Pages 1–9
DOI 10.1155/ASP/2006/17195

MASSP3: A System for Predicting Protein Secondary Structure

Giuliano Armano, Alessandro Orro, and Eloisa Vargiu

Department of Electrical and Electronic Engineering, University of Cagliari, Piazza d'Armi, 09123 Cagliari, Italy

Received 15 May 2005; Revised 22 September 2005; Accepted 1 December 2005

A system that resorts to multiple experts for dealing with the problem of predicting secondary structures is described, whose performances are comparable to those obtained by other state-of-the-art predictors. The system performs an overall processing based on two main steps: first, a "sequence-to-structure" prediction is performed, by resorting to a population of hybrid genetic-neural experts, and then a "structure-to-structure" prediction is performed, by resorting to a feedforward artificial neural networks. To investigate the performance of the proposed approach, the system has been tested on the RS126 set of proteins. Experimental results (about 76% of accuracy) point to the validity of the approach.

1. INTRODUCTION

Due to the strict relation between protein function and structure, the prediction of protein 3D structure has during recent years become one of the most important tasks in bioinformatics. In fact, notwithstanding the increase of experimental data on protein structures available in public databases, the gap between known sequences (165,000 entries in Swiss-Prot [1] in December 2004) and known tertiary structures (28,000 entries in PDB [2] in December 2004) is constantly increasing. The need for automatic methods has brought the development of several prediction and modeling tools, but despite the increase of accuracy a general methodology to solve the problem has not been yet devised. Building complete protein tertiary structure is still not a tractable task, and most methodologies concentrate on the simplified task of predicting their secondary structure. In fact, the knowledge of secondary structure is a useful starting point for further investigating the problem of finding protein tertiary structures and functionalities. In this paper, we concentrate on the problem of predicting secondary structures using a system that performs an overall processing based on two main steps: first, a "sequence-to-structure" prediction is performed, by resorting to a population of hybrid genetic-neural experts, and then a "structure-to-structure" prediction is performed, by resorting to a feedforward artificial neural network (ANN). Multiple experts are the underlying technology of the former subsystem, also rooted in two powerful soft-computing techniques, that is, genetic and neural. It is worth pointing out that here the term "expert" denotes a software module entrusted with the task of predicting protein secondary structure in combination with other experts of the same kind.

The remainder of this paper is organized as follows. In Section 2, some relevant work is briefly recalled. Section 3 introduces the architecture of the system that has been devised to perform secondary structure prediction. Section 4 reports experimental results. Section 5 draws conclusions and future work.

2. RELATED WORK

In this section, some relevant related work is briefly recalled, according to both an applicative and a technological perspective. The former is mainly focused on the task of secondary structure prediction, whereas the latter concerns the subfield of multiple experts, which the proposed system stems from.

2.1. Protein structure prediction

The secondary structure of protein is the local spatial arrangement of its main-chain atoms without regard to the conformation of its side chains or to its relationship with other segments. In practice, the problem of predicting the secondary structure of a protein basically consists of finding a linear labeling representing the conformation to which each residue belongs. Each residue is mapped into a secondary alphabet composed—in the simplest case—of three symbols: alpha helix (α), beta sheet (β), and random coil (c). Assessing the secondary structure can help in building the complete protein structure, and can be useful information for making hypotheses on the protein functionality. In fact, very often, active sites are associated with a particular conformation

or combination (motifs) of secondary structures conserved during the evolution.

There are a variety of secondary structure prediction methods proposed in the literature. Early prediction methods were based on statistics headed at evaluating, for each amino acid, the likelihood of belonging to a given secondary structure [3]. A second generation of methods exhibits better performance by exploiting protein databases, as well as statistic information about amino acid subsequences. Several methods exist in this category, which may be classified according to (i) the underlying approach including statistical information [4], graph theory [5], multivariate statistics [6], and linear discriminant analysis [7], (ii) the kind of information actually taken into account including physicochemical properties [8] and sequence patterns [9], or (iii) the adopted technique, including k-nearest neighbors [10] and ANNs [11].

The most significant innovation introduced in this field was the exploitation of evolutionary information contained in multiple alignments. The underlying motivation is that active regions of homologous sequences will typically adopt the same local structure, irrespective of local sequence variations. PHD [11] is one of the first successful methods based on ANNs that make use of evolutionary information to perform secondary structure prediction. In particular, after searching similar sequences using BLASTP [12], ClustalW [13] is invoked to identify which residues can actually be substituted without compromising the functionality of the target sequence. To predict secondary structure, the multiple alignment produced by ClustalW is given as input to a multilayer ANN. The first layer outputs a sequence-to-structure prediction, which is sent to a further ANN layer that performs a structure-to-structure prediction aimed at refining it.

Further improvements are obtained with both more accurate multiple alignment strategies and more powerful neural network architectures. For instance, PSI-PRED [14] exploits the position-specific scoring matrix (called "profile") built during a preprocessing performed by PSI-BLAST (see also [15]). This approach outperforms PHD thanks to the PSI-BLAST ability of detecting distant homologies. Other relevant works include DSC [7], PREDATOR [16, 17], NNSSP [10], and JPred [18, 19]. DSC combines the compositional features of multiple alignments with empirical rules that are found important for secondary structure prediction. The information is processed using linear statistics. PREDATOR owes its accuracy mostly to the incorporation of long-range interactions for β-strand prediction. NNSSP is the actual system that resorts to the k-nearest neighbors technique to perform prediction. JPred predicts secondary structure by combining a number of modern, high quality prediction methods to form a consensus. In more recent work [20, 21], recurrent ANNs (RANNs) are exploited to capture long-range interactions. The actual system that embodies such capabilities, that is, SSPRO [22], is characterized by (i) PSI-BLAST profiles for encoding inputs, (ii) bidirectional RANNs, and (iii) a predictor based on ensembles of RANNs.

2.2. Multiple experts

Divide and conquer is one of the most popular strategies aimed at recursively partitioning the input space until regions of roughly constant class membership are obtained. Several machine learning approaches, for example, decision lists (DL) [23, 24], decision trees (DT) [25], counterfactuals (CFs) [26], classification and regression trees (CART) [27] apply this strategy to control the search, thus yielding monolithic solutions. Nevertheless, a partitioning procedure can also be considered as a "tool" for generating multiple experts. Although with a different focus, this multiple experts' perspective has been adopted by the evolutionary computation and by the connectionist communities. In the former case, the focus was on devising suitable architectures and techniques able to enforce an adaptive behavior on a population of individuals (see, e.g., [28, 29]). Genetic algorithms (GAs) [30–33], learning classifier systems (LCSs) [34, 35], and extended classifier systems (XCSs) [36] fall in this specific category of metaheuristics (see also [37] for a description about evolutionary computation applied to bioinformatics). In the latter case, the focus was mainly on training techniques and output combination mechanisms; in particular, let us recall Jordan's mixtures of experts [38, 39] and Weigend's gated experts [40].

Further investigations are focused on comparing the behavior of a population of experts with respect to a single expert. Theoretical studies and empirical results, rooted in the computational and/or statistical learning theory (see, e.g., [41, 42]), have shown that the overall performance of a system can be significatively improved by adopting an approach based on multiple experts. Relevant studies in this subfield include ANN ensembles [43, 44] and DT ensembles [45, 46]. There has also been a great interest in combining evolutionary and connectionist approaches, giving rise to evolutionary ANNs (EANNs) [47]. In recent years, the focus of interest moved from single ANNs to ensembles of ANNs, yielding hybrid learning systems in which—typically—a population of ANNs is designed by exploiting the characteristics of an evolutionary process [48].

3. THE ARCHITECTURE OF MASSP3

This section introduces the two-tiered approach devised to perform protein secondary structure prediction. The corresponding system has been called MASSP3, standing for multiagent secondary structure predictor with postprocessing. As shown in Figure 1, the information flows according to a pipeline in which the first and the second modules are entrusted with performing a sequence-to-structure (P2S) and a structure-to-structure (S2S) predictions, respectively.

3.1. Sequence-to-structure prediction

In this subsection, the module that has been devised to perform the first step, which stems from the one proposed in [49, 50], is briefly described–focusing on the internal details that characterize an expert (microarchitecture) and on the

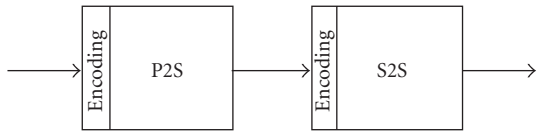

Figure 1: The overall architecture of MASSP3, consisting of a population of experts devised to perform sequence-to-structure prediction (P2S), followed by a postprocessor, devised to perform structure-to-structure prediction (S2S).

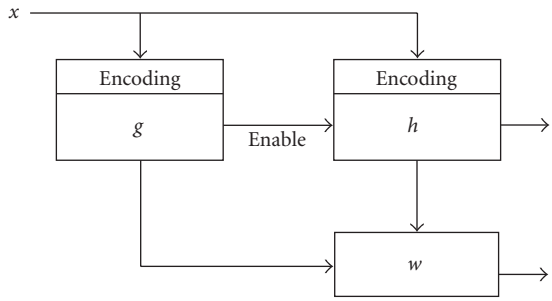

Figure 2: The microarchitecture of an expert.

behavior of the overall population of experts (macroarchitecture). Due to its impact on the overall accuracy of the system, the solution adopted to deal with the problem of how to encode inputs for embedded experts is briefly outlined in a separate section.

Microarchitecture

In its current formulation, the general structure of a single expert Γ is a quadruple $\langle l, g, h, w \rangle$, where l is a class label, g is a "guard", that is, a function devised to accept or discard inputs according to the value of some relevant features, h is an embedded expert whose activation depends on g, and w is a weighting function, used to perform output combination (see Figure 2). Hence, $\Gamma(x)$ coincides with $h(x)$ for any input x that matches $g(x)$, otherwise it is not defined. An expert Γ contributes to the final prediction according to the value $w(x)$ of its weighting function, which represents the expert strength in the voting mechanism.

As for the structure of guards, in the simplest case, the main responsibility of g is to split the input space into matching/nonmatching regions, with the goal of facilitating the training of h. In a typical evolutionary setting, each guard performs a "hard-matching" activity, implemented by resorting to an embedded pattern in $\{0, 1, \#\}^L$, where "#" denotes the usual "don't care" symbol and L denotes the length of the pattern. Given an input x, consisting of a string in the alphabet $\{0, 1\}$, the matching between x and g returns *true* if and only if all non-# values coincide (otherwise, the matching returns *false*). It is trivial to extend this definition by devising guards that map inputs to $[0, 1]$. Though very simple from a conceptual perspective, this relaxed interpretation requires the adoption of a flexible matching mechanism, which has been devised according to the following semantics: given an input x, a guard g evaluates the overall matching score $g(x)$, and activates the corresponding embedded expert h if and only if $g(x) \geq \theta$ (the threshold θ is a system parameter).

Let us assume that g embeds a pattern e, represented by a string in $\{0, 1, \#\}$ of length L, used to evaluate the distance between an input x and the guard. To improve the generality of the system, one may assume that a vector of relevant, domain-dependent features is provided, able to implement a functional transformation from x to $[0, 1]^L$. In so doing, the ith feature, denoted by $m_i(x)$, can be associated with the ith value, say e_i, of the embedded pattern e. Under these assumptions, the function $g(x)$ can be defined as (d denotes a suitable distance metrics)

$$g(x) = 1 - d(e, m(x)). \tag{1}$$

In our opinion the most natural choice for implementing the distance metrics should extend the hard-matching mechanism used in a typical evolutionary setting. In practice, the ith component of e controls the evaluation of the corresponding input features, so that only non-"#" features are actually taken into account. Hence, $H_g \neq \emptyset$ being the set of all non-"#" indexes in e, $g(x)$ can be defined, according to the Minkowski's L_∞ distance metrics, as

$$g(x) = 1 - \max_{i \in H_g} \{ | e_i - m_i(x) | \}. \tag{2}$$

Let us stress that the result should be interpreted as a "degree of expertise" of an expert over the given input x.

As for embedded experts, a simple multilayer perceptron (MLP) architecture has been adopted—equipped with a single hidden layer. The issue of the dependence between the number of inputs and the number of neurons in the hidden layer has also been taken into account. Several experiments addressed the problem of finding a good tradeoff between the need of limiting the number of hidden neurons and the need of augmenting it (to prevent overfitting and underfitting, resp.). Let us stress in advance that overfitting has been greatly reduced by experimenting a novel type of encoding which performs a kind of multiple alignment by resorting to the substitution matrix [51] (e.g., *Blosum80* [52]). As a consequence, the underfitting problem has also become more tractable, due to the fact that the range of "reasonable" choices for ANN architectures has increased. In particular, an embedded expert with a complete visibility of the input space is equipped with 35 hidden neurons, whereas experts enabled by 10%, 20%, and 30% of the input space are equipped with 10, 15, and 20 neurons, respectively.

Macroarchitecture

Experts are trained in two steps, which consist of (1) discovering a population of guards aimed at soft partitioning the input space, and (2) training the embedded experts of the resulting population.

In the first step, experts are generated concentrating only on the "partitioning" capability of their guards (let us recall that a guard is aimed at identifying a context able to facilitate

the prediction performed by the corresponding embedded expert). In particular, the system starts with an initial population of experts equipped with randomly generated guards, and then further experts are created according to covering, crossover, or mutation mechanisms. In this phase, embedded experts play a secondary role, their training being deferred to the second step. Until then, their output is steadily "1," meaning that the class label l is asserted with the highest strength. It is worth pointing out that, at the end of the first step, for each class label a globally scoped expert (i.e., equipped with a guard whose embedded pattern contains only "#") is inserted in the population, to guarantee that the input space is completely covered in any case.[1] From this point on, no further creation of experts is performed.

In the second step the focus moves to embedded experts, which, turned into MLPs, are trained using the backpropagation algorithm on the subset of inputs acknowledged by their corresponding guard. Let us note that each embedded predictor h is actually equipped with a "complementary" output, independently trained and denoted by $\overline{h}(x)$. This choice allows to easily evaluate the reliability $r(x)$ of the prediction (see below), estimated by $|h_\Gamma(x) - \overline{h}_\Gamma(x)|$ (see also [11]). In the current implementation of the system, all MLPs are trained in parallel, until a convergence criterion is satisfied or the maximum number of epochs has been reached. The training of MLPs follows a special technique, explicitly devised for this specific application. In particular, given an expert consisting of a guard g and its embedded expert h, h is trained on the whole training set in the first five epochs, whereas the visibility of the training set is restricted to the inputs matched by the corresponding guard in the subsequent epochs. In this way, a mixed training strategy has been adopted, whose rationale lies in the fact that experts must find a suitable tradeoff between the need of enforcing diversity (by specializing themselves on a relevant subset of the input space) and the need of preventing overfitting.

As for the output combination policy, let us recall that—by hypothesis—experts do not have complete visibility of the input space (i.e., they typically operate on different regions). In the implementation designed for predicting protein secondary structures, regions exhibit a particular kind of "soft" boundaries, in accordance with the selected flexible matching mechanism. Given an input x, all selected experts form the match set, denoted by $M(x)$, which in turn can be partitioned into three separate subsets: $M_\alpha(x)$, $M_\beta(x)$, and $M_c(x)$. Each subset contains only experts that support α, β, and c, respectively.

Given an input x, for each expert $\Gamma \in M(x)$, let us denote with $g_\Gamma(x)$ its degree of expertise over x, with $h_\Gamma(x)$ its prediction, and with $w_\Gamma(x)$ its strength. It is worth noting that, in the current implementation, $w_\Gamma(x)$ depends (i) on the degree of expertise, (ii) on the fitness, and (iii) on the reliability of the prediction. Under these hypotheses, the P2S module "annotates" each input x with a triple of values (one for each class label) according to the following policy:

$$O^k(x) = \frac{\sum_{\Gamma \in M_k(x)} h_\Gamma(x) \cdot w_\Gamma(x)}{\sum_{\Gamma \in M_k(x)} w_\Gamma(x)}, \quad k \in \{\alpha, \beta, c\},$$
$$w_\Gamma(x) = f_\Gamma \cdot g_\Gamma(x) \cdot r_\Gamma(x), \tag{3}$$

where $M_k(x) \subseteq M(x)$ contains only experts that assert k, and f_Γ and $r_\Gamma(x) = |h_\Gamma(x) - \overline{h}_\Gamma(x)|$ denote the fitness and the reliability of the expert Γ. In so doing, the P2S module outputs three separate "signals," which estimate, along the given sequence, the likelihood of each amino acid to be labeled as α, β, or c.

Input encoding

The list of features handled by guards (adopted for soft partitioning the input space) is reported in Table 1, which represents a first attempt to inject into the system useful domain knowledge.

As for embedded predictors, we propose a solution based on the *Blosum80* substitution matrix, which enforces a sort of "low-pass" filtering with respect to the typical encoding based on multiple alignment. Some preliminary definition follows.

(i) Each amino acid is represented by an index in [1–3, 12, 15, 18–21, 24, 27–31, 44, 49–51, 53] (i.e., 1/Alanine, 2/Arginine, 3/Asparagine, ..., 19/Tyrosine, 20/Valine). The index 0 is reserved for representing the gap.

(ii) $\mathbf{P} = \langle P_i, i = 0, 1, \ldots, n\rangle$ is a list of sequences where (i) P_0 is the protein to be predicted (i.e., the primary input sequence), containing L amino acids, and (ii) P_i, $i = 1, \ldots, n$, is the list of sequences related with P_0 by means of similarity-based metrics, retrieved using BLAST. Being multialigned with P_0, these sequences usually contain gaps, so that their length still amounts to L. Furthermore, let us denote with $P(j)$, $j = 1, 2, \ldots, L$, the jth column of the multialignment, and with $P_i(j), j = 1, 2, \ldots, L$, the jth residue of the sequence P_i.

(iii) $\mathbf{B}$ is a 21×21 matrix obtained by normalizing the *Blosum80* matrix in the range [0,1]. Thus, B_k denotes the row of $\mathbf{B}$ that encodes the amino acid k ($k = 1, 2, \ldots, 20$), whereas $B_k(r)$ represents the degree of substitability of the rth amino acid with the kth amino acid. The row and the column identified by the 0th index represent the gap, set to a null vector in both cases—except for the element $B_0(0)$ which is set to 1.

(iv) $\mathbf{Q}$ is a matrix of $21 \times L$ positions, representing the final encoding of the primary input sequence P_0. Thus, $Q(j)$ denotes the jth column of the matrix, which is intended to encode the jth amino acid (i.e., $P_0(j)$) of the primary input sequence (i.e., P_0), whereas $Q_r(j), r = 0, 1, \ldots, 20$, represents the contribution of the rth amino acid in the encoding of $P_0(j)$ (the index $r = 0$ is reserved for the gap).

The normalization of the *Blosum80* matrix in the range [0,1], yielding the $\mathbf{B}$ matrix, is performed according to the

[1] The weighting function of such kind of experts always returns a constant and negligible value, to prevent them from affecting the result of output combination in presence of locally scoped experts.

Table 1: Features used for "soft" partitioning the input space: each feature is evaluated on a window of length r and centered around the residue to be predicted.

	Feature	Conjecture
1	Check whether hydrophobic amino acids occur in the current window ($r = 15$) according to a clear periodicity (i.e., one every 3-4 residues)	Alpha helices may sometimes fulfil this pattern
2	Check whether the current window ($r = 13$) contains numerous residues in {A,E,L,M} and few residues in {P,G,Y,S}	Alpha helices are often evidenced by {A,E,L,M} residues, whereas {P,G,Y,S} residues account for their absence
3	Check whether the left side of the current window ($r = 13$) is mostly hydrophobic and the right part is mostly hydrophilic (or vice-versa)	Transmembrane alpha helices may fulfil this feature
4	Check whether, on the average, the current window ($r = 11$) is positively charged or not	A positive charge might account for alpha helices or beta sheets
5	Check whether, on the average, the current window ($r = 11$) is negatively charged or not	A negative charge might account for alpha helices or beta sheets
6	Check whether, on the average, the current window ($r = 11$) is neutral	A neutral charge might account for coils
7	Check whether the current window ($r = 11$) mostly contains "small" residues	Small residues might account for alpha helices or beta sheets
8	Check whether the current window ($r = 11$) mostly contains polar residues	Polar residues might account for alpha helices or beta sheets

following guidelines:

(1) μ and σ being the mean and the standard deviations of the *Blosum80* matrix, respectively, calculate the "equalized matrix" **E** by applying a suitable sigmoid function, whose zero crossing is set to μ and with a range in $[-\sigma, \sigma]$; in symbols,

$$\begin{aligned}\forall k = 1, 2, \ldots, 20 : \forall j = 1, 2, \ldots, 20 : E_k(j)\\ \longleftarrow \sigma \cdot \tanh\left(Blosum80_k(j) - \mu\right),\end{aligned} \tag{4}$$

(2) E^m and E^M being the minimum and the maximum values of the equalized matrix **E**, respectively, build the normalized matrix **B**; in symbols (the 0th row and column of **B** are used to encode gaps),

$$B_0 \longleftarrow \langle 1, 0, \ldots, 0\rangle, \quad B(0) \longleftarrow \langle 1, 0, \ldots, 0\rangle^T$$
$$\forall k = 1, 2, \ldots, 20 : \forall j = 1, 2, \ldots, 20 : B_k(j) \longleftarrow \frac{E_k(j) - E^m}{E^M - E^m}. \tag{5}$$

The algorithm used for encoding the primary input sequence P_0 is the following.

(1) Initialize **Q** with the *Blosum80*-like encoding of the primary sequence P_0 (B_s^T represents the vector B_s transposed); in symbols,

$$\forall j = 1, 2, \ldots, L : s \longleftarrow P_0(j), \quad Q(j) \longleftarrow B_s^T. \tag{6}$$

(2) Update **Q** according to the *Blosum80*-like encoding of the remaining sequences $P_1, P_2, \ldots, P_n$; in symbols,

$$\begin{aligned}\forall i = 1, 2, \ldots, n : \forall j = 1, 2, \ldots, L : s \longleftarrow P_i(j),\\ Q(j) \longleftarrow Q(j) + B_s^T.\end{aligned} \tag{7}$$

(3) Normalize the elements of **Q**, column by column, in [0,1]; in symbols,

$$\begin{aligned}\forall j = 1, 2, \ldots, L : \gamma \longleftarrow \sum_s Q_s(j),\\ \forall r = 0, 1, 2, \ldots, 20 : Q_r(j) \longleftarrow \frac{Q_r(j)}{\gamma}.\end{aligned} \tag{8}$$

According to our experimental results, the encoding defined above greatly contributes to reduce overfitting and produces an improvement of about 1.5% in the prediction performance. It is worth noting that also in this case a mixed strategy—in a sense similar to the one adopted for training ANNs—has been enforced, where the information contained in the *Blosum80* matrix and in multiple alignment represent the "global" and the "local" parts, respectively. As a final remark, let us stress that a comparison between *Blosum80* and PSI-BLAST encoding exhibited only negligible differences.

3.2. Structure-to-structure prediction

It is well known that protein sequences have a high correlation in their secondary structure, which may be taken into

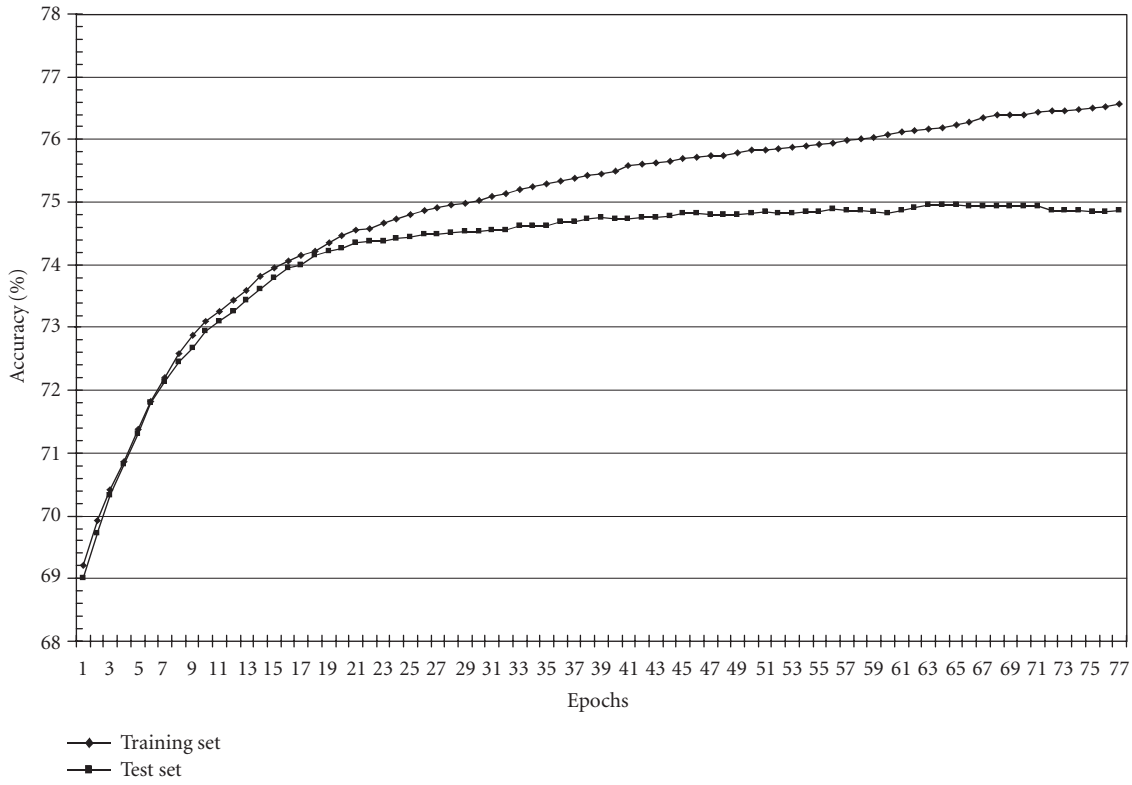

Figure 3: The second step of the training activity of the population of experts that characterize the P2S module. Its overall performance, obtained while training embedded MLP predictors, is reported at different epochs. The plot highlights that a limited amount of overfitting occurred—also due to the specific encoding, based on the *Blosum80* matrix that has been devised and adopted.

account to improve the prediction accuracy. Technologies that adopt a simple residue-centric approach, in which secondary structures are predicted independently, often generate inconsistent and unrealistic secondary structure assignment, for example, isolated alpha helices. To deal with this problem, a suitable postprocessing is usually performed. The postprocessing module can be either hand coded or automatically generated. In the former case, it follows the guidelines of suitable empirical rules, whereas in the latter an architecture typically based on ANNs is devised and trained on the inputs generated by the subsystem responsible for the P2S prediction. In the implementation of MASSP3, we adhered to the latter type of postprocessing techniques, and a preliminary "low-pass" filtering is also performed on the prediction produced by the population of experts. For each class label, it calculates a value averaged over windows of three residues, according to the profile of a suitable Gaussian shape. The actual postprocessing is performed by an MLP, trained on the signals obtained for α, β, and c after running the aforementioned "low-pass" filtering. For each position of the sequence the MLP takes as input the resulting three-dimensional signal on a window of 21 residues (i.e., 63 inputs) and generates three outputs in $[0, 1]$—to be considered as pseudo-probabilities. Each amino acid of the given sequence is then labeled with α, β, or c according to a criterion of maximum likelihood.

4. EXPERIMENTAL RESULTS

To assess the performance of the predictor, also facilitating a comparison with other systems, we adopted the TRAIN and the R126 datasets, for training and testing, as described in [22]. The TRAIN dataset has been derived from a PDB selection obtained by removing short proteins (less than 30 amino acids) and by keeping proteins with a resolution of at least 2.5 Å. This dataset underwent a homology reduction, aimed at excluding sequences with more than 50% of similarity. Furthermore, proteins in this set have less than 25% identity with the sequences in the set R126. The resulting training set consists of 1180 sequences, corresponding to 282,303 amino acids. The distribution of α, β, c in the training set is 35.41%, 22.75%, 41.84%. The R126 test dataset is derived from the historical Rost and Sander's protein dataset (RS126) [11], and corresponds to a total of 23,363 amino acids (the overall number has slightly varied over the years, due to changes and corrections in the PDB). The distribution of α, β, c in the test set is 31.78%, 23.14%, 45.08%.

In the experiments carried out on the P2S subsystem, the population was composed of 600 experts, with about 20 experts (on average) involved in the match set. The threshold θ has been set to 0.4. As for MLPs, the learning rate has been set to 0.07 and the number of epochs to 80. Results obtained by the P2S module allow to reach a performance of

Table 2: Experimental results, obtained from the RS126 dataset.

System	Q_3
SSPRO	76.6
MASSP3	**76.1**
JPred	74.8
PHD	73.5
NNSSP	72.7
DSC	71.1
PREDATOR	70.3

Table 3: Detailed results obtained by using MASSP3 on the RS126 test set using the *Blosum80* encoding and postprocessing.

	α	β	c	*overall*
Q_3	78.2	63.1	81.3	76.1
SOV	77.8	73.2	70.3	71.8

about 75%. Figure 3 illustrates the second step of the training process, which occurred after generating a suitable population of guards able to "soft" partitioning the input space. The S2S module allowed to improve the alignment of about 1%, so that the overall accuracy of more than 76% has been obtained. To facilitate the comparison with other relevant systems, MASSP3 has also been assessed according to the guidelines described in [19]. In particular, the programs NNSSP, PHD, DSC, PREDATOR, and JPred have been considered concerning performance against the commonly used RS126 dataset.

Having trained MASSP3 using the same dataset (i.e., TRAIN), we reported also the performance of SSPRO [22]. Experimental results are summarized in Table 2.

To give a better insight on the characteristics of MASSP3. Table 3 reports the accuracy of the system and the SOV scores [54] for the three secondary structure labels, as well as the overall Q_3 and SOV score (let us recall that SOV measures the accuracy of the prediction in terms of secondary structure segments rather than on individual residues).

As a final remark on experimental results, let us point out that the fact that SSPRO obtains better results is not surprising, this system being based on a technology (i.e., recurrent ANNs—see, e.g., [53]) which is deemed more adequate than MLPs for processing sequences. Nevertheless, in our opinion, the proposed system has still great potentiality to improve its performances, due to its ability of taking into account suitable domain knowledge and to the possibility of adopting more powerful techniques (e.g., RANNs, HMMs) for implementing embedded experts.

5. CONCLUSIONS AND FUTURE WORK

In this paper, an approach for predicting protein secondary structures has been presented, which relies on two-tiered architecture, consisting of a sequence-to-structure predictor, followed by a structure-to-structure predictor. The former resorts to a multiple-expert architecture, in which a population of hybrid experts—embodying a genetic and a neural part—has been suitably devised to perform the given application task. The latter consists of an MLP, fed with the first-stage prediction suitably encoded by a "low-pass" filter. Experimental results, performed on sequences taken from well-known protein databases, improve those obtained with most state-of-the-art predictors. As for the future work, in collaboration with a biologist, we are trying to devise more "biologically based" features—to be embedded in genetic guards—able to improve their ability of performing context identification. The adoption of RANNs is also being investigated as the underlying technology for implementing embedded experts.

REFERENCES

[1] A. Bairoch and R. Apweiler, "The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000," *Nucleic Acids Research*, vol. 28, no. 1, pp. 45–48, 2000.

[2] H. M. Berman, J. Westbrook, Z. Feng, et al., "The protein data bank," *Nucleic Acids Research*, vol. 28, no. 1, pp. 235–242, 2000.

[3] P. Y. Chou and U. D. Fasman, "Prediction of protein conformation," *Biochemistry*, vol. 13, pp. 211–215, 1974.

[4] B. Robson and E. Suzuki, "Conformational properties of amino acid residues in globular proteins," *Journal of Molecular Biology*, vol. 107, no. 3, pp. 327–356, 1976.

[5] E. M. Mitchell, P. J. Artymiuk, D. W. Rice, and P. Willett, "Use of techniques derived from graph theory to compare secondary structure motifs in proteins," *Journal of Molecular Biology*, vol. 212, pp. 151–166, 1992.

[6] M. Kanehisa, "A multivariate analysis method for discriminating protein secondary structural segments," *Protein Engineering*, vol. 2, no. 2, pp. 87–92, 1988.

[7] R. D. King and M. J. E. Sternberg, "Identification and application of the concepts important for accurate and reliable protein secondary structure prediction," *Protein Science*, vol. 5, pp. 2298–2310, 1996.

[8] O. B. Ptitsyn and A. V. Finkelstein, "Theory of protein secondary structure and algorithm of its prediction," *Biopolymers*, vol. 22, no. 1, pp. 15–25, 1983.

[9] W. R. Taylor and J. M. Thornton, "Prediction of super-secondary structure in proteins," *Nature*, vol. 301, pp. 540–542, 1983.

[10] A. A. Salamov and V. Solovyev, "Prediction of protein secondary structure by combining nearest neighbor algorithms and multiple sequence alignment," *Journal of Molecular Biology*, vol. 247, pp. 11–15, 1995.

[11] B. Rost and C. Sander, "Prediction of protein secondary structure at better than 70% accuracy," *Journal of Molecular Biology*, vol. 232, no. 2, pp. 584–599, 1993.

[12] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *Journal of Molecular Biology*, vol. 215, no. 3, pp. 403–410, 1990.

[13] D. Higgins, J. Thompson, T. Gibson, J. D. Thompson, D. G. Higgins, and T. J. Gibson, "CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice," *Nucleic Acids Research*, vol. 22, no. 22, pp. 4673–4680, 1994.

[14] D. T. Jones, "Protein secondary structure prediction based on position-specific scoring matrices," *Journal of Molecular Biology*, vol. 292, no. 2, pp. 195–202, 1999.

[15] S. F. Altschul, T. L. Madden, A. A. Schaeffer, et al., "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs," *Nucleic Acids Research*, vol. 25, no. 17, pp. 3389–3402, 1997.

[16] D. Frishman and P. Argos, "Incorporation of long-distance interactions into a secondary structure prediction algorithm," *Protein Engineering*, vol. 9, pp. 133–142, 1996.

[17] D. Frishman and P. Argos, "75% accuracy in protein secondary structure prediction," *Proteins*, vol. 27, pp. 329–335, 1997.

[18] J. A. Cuff, M. E. Clamp, A. S. Siddiqui, M. Finlay, and G. J. Barton, "Jpred: a consensus secondary structure prediction server," *Bioinformatics*, vol. 14, pp. 892–893, 1998.

[19] J. A. Cuff and G. J. Barton, "Evaluation and improvement of multiple sequence methods for protein secondary structure prediction," *PROTEINS: Structure, Function and Genetics*, vol. 34, pp. 508–519, 1999.

[20] P. Baldi, S. Brunak, P. Frasconi, G. Soda, and G. Pollastri, "Exploiting the past and the future in protein secondary structure prediction," *Bioinformatics*, vol. 15, no. 11, pp. 937–946, 1999.

[21] P. Baldi, S. Brunak, P. Frasconi, G. Pollastri, and G. Soda, "Bidirectional dynamics for protein secondary structure prediction," in *Sequence Learning: Paradigms, Algorithms, and Applications*, R. Sun and C. L. Giles, Eds., pp. 80–104, Springer, New York, NY, USA, 2000.

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

[23] R. L. Rivest, "Learning decision lists," *Machine Learning*, vol. 2, no. 3, pp. 229–246, 1987.

[24] P. Clark and T. Niblett, "The CN2 induction algorithm," *Machine Learning*, vol. 3, no. 4, pp. 261–283, 1989.

[25] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[26] S. A. Vere, "Multilevel counterfactuals for generalizations of relational concepts and productions," *Artificial Intelligence*, vol. 14, no. 2, pp. 139–164, 1980.

[27] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*, Wadsworth, Belmont, Calif, USA, 1984.

[28] T. Back, D. Fogel, and Z. Michalewicz, *Handbook of Evolutionary Computation*, Oxford University Press, New York, NY, USA, 1997.

[29] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*, Springer, New York, NY, USA, 2003.

[30] H. J. Bremmerman, "Optimization through evolution and recombination," in *Self-Organizing Systems*, M. C. Yovits, G. T. Jacobi, and G. D. Goldstine, Eds., pp. 93–106, Spartan Books, Washington, DC, USA, 1962.

[31] L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence Through Simulated Evolution*, John Wiley & Sons, New York, NY, USA, 1966.

[32] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Mich, USA, 1975.

[33] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, Mass, USA, 1989.

[34] J. H. Holland, "Adaption," in *Progress in Theoretical Biology*, R. Rosen and F. M. Snell, Eds., vol. 4, pp. 263–293, Academic Press, New York, NY, USA, 1976.

[35] J. H. Holland, "Escaping brittleness: the possibilities of general purpose learning algorithms applied to parallel rule based systems," in *Machine Learning, An Artificial Intelligence Approach*, R. S. Michalski, J. Carbonell, and M. Mitchell, Eds., vol. 2, chapter 20, pp. 593–623, Morgan Kaufmann, Los Altos, Calif, USA, 1986.

[36] S. W. Wilson, "Classifier fitness based on accuracy," *Evolutionary Computation*, vol. 3, no. 2, pp. 149–175, 1995.

[37] G. B. Fogel and D. W. Corne, Eds., *Evolutionary Computation in Bioinformatics*, Morgan Kaufmann, San Francisco, Calif, USA, 2003.

[38] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991.

[39] M. I. Jordan and R. A. Jacobs, "Hierarchies of adaptive experts," in *Advances in Neural Information Processing Systems*, J. Moody, S. Hanson, and R. Lippman, Eds., vol. 4, pp. 985–993, Morgan Kaufmann, San Mateo, Calif, USA, 1992.

[40] A. S. Weigend, M. Mangeas, and A. N. Srivastava, "Nonlinear gated experts for time series: discovering regimes and avoiding overfitting," *International Journal of Neural Systems*, vol. 6, no. 4, pp. 373–399, 1995.

[41] L. Valiant, "A theory of the learnable," *Communications of the ACM*, vol. 27, pp. 1134–1142, 1984.

[42] V. N. Vapnik, *Statistical Learning Theory*, John Wiley & Sons, New York, NY, USA, 1998.

[43] A. Krogh and J. Vedelsby, "Neural network ensembles, cross validation, and active learning," in *Advances in Neural Information Processing Systems*, G. Tesauro, D. Touretzky, and T. Leen, Eds., vol. 7, pp. 231–238, MIT Press, Cambridge, Mass, USA, 1995.

[44] L. Breiman, "Stacked regressions," *Machine Learning*, vol. 24, pp. 41–48, 1996.

[45] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer Science and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[46] R. E. Schapire, "A brief introduction to boosting," in *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pp. 1401–1406, Stockholm, Sweden, 1999.

[47] X. Yao, "Evolving artificial neural networks," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1423–1447, 1999.

[48] X. Yao and Y. Liu, "Evolving neural network ensembles by minimization of mutual information," *International Journal of Hybrid Intelligent Systems*, vol. 1, no. 1, pp. 12–21, 2004.

[49] G. Armano, G. Mancosu, and A. Orro, "A multi agent system for protein secondary structure prediction," in *The 4th International Workshop on Network Tools and Applications in Biology "Models and Metaphors from Biology to Bioinformatics Tools" (NETTAB '04)*, Camerino, Italy, 2004.

[50] G. Armano, "NXCS experts for financial time series forecasting," in *Applications of Learning Classifier Systems*, L. Bull, Ed., pp. 68–91, Springer, New York, NY, USA, 2004.

[51] G. Armano, A. Orro, and M. Saba, "Encoding multiple alignments by resorting to substitution matrices," DIEE - Tech. Rep., University of Cagliari, Cagliari, Italy, May 2005.

[52] S. Henikoff and J. G. Henikoff, "Amino acid substitution matrices from protein blocks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 89, no. 2, pp. 10915–10919, 1992.

[53] A. Cleeremans, *Mechanisms of Implicit Learning Connectionist Models of Sequence Processing*, MIT Press, Cambridge, Mass, USA, 1993.

[54] A. Zemla, C. Vencolvas, K. Fidelis, and B. Rost, "A modified definition of Sov, a segment-based measure for protein secondary structure prediction assessment," *Proteins*, vol. 34, no. 2, pp. 220–223, 1999.

Giuliano Armano obtained his Ph.D. degree in electronic engineering from the University of Genoa, Italy, in 1990. He is currently Associate Professor of computer engineering at the Department of Electrical and Electronic Engineering (DIEE), University of Cagliari, leading also the IASC (Intelligent Agents and Soft Computing) group. His educational background ranges over expert systems and machine learning, whereas his current research activity is focused on (i) proactive and adaptive behavior of intelligent agents and (ii) hybrid genetic-neural architectures and systems. The above research topics are mainly experimented in the field of bioinformatics, in particular for designing and implementing algorithms for multiple alignment and protein secondary structure prediction.

Alessandro Orro received his Ph.D. degree in electronics and computer engineering in February 2005, after a three-year course at the University of Cagliari, Italy, under the supervision of Professor G. Armano. He is currently working at ITB-CNR, Milan, Italy. His main research interests are in the field of Bioinformatics; in particular he is investigating multiple alignment algorithms and techniques for protein secondary structure prediction. The underlying techniques and tools, such as genetic algorithms and artificial neural networks, fall into the category of soft computing.

Eloisa Vargiu obtained her M.S. and Ph.D. degrees in electronic and computer engineering from the University of Cagliari, Italy, in 1999 and 2003, respectively. Since 2000, she collaborates with the Intelligent Agents and Soft Computing (IASC) group at the Department of Electrical and Electronic Engineering (DIEE), University of Cagliari. Her educational background is mainly focused on intelligent agents, in particular on their proactive and adaptive behavior. Her research interests are currently in the field of artificial intelligence; in particular, intelligent agents and bioinformatics.