ARTÍCULO DE INVESTIGACIÓN ORIGINAL

REVISTA MEXICANA DE INGENIERÍA BIOMÉDICA
SOMIB
Vol. XXXII, Núm. 2
Diciembre 2011
pp 109 - 118

Identification of functional sequences using associative memories

Román-Godínez I*
Garibay-Orijel C**
Yáñez-Márquez C***

* Laboratorio de Redes Neuronales y Computo no Convencional, Centro de Investigación en Computación, Instituto Politécnico Nacional, México, D.F.
** Laboratorio de Bioingeniería, Unidad Profesional Interdisciplinaria de Biotecnología, Instituto Politécnico Nacional, México, D.F.

Correspondence:
Israel Román Godínez
Av. Juan de Dios Bátiz s/n casi esquina Miguel Othón de Mendizábal, Unidad Profesional «Adolfo López Mateos», Edificio CIC,
Col. Nueva Industrial Vallejo, 07738, Del. Gustavo A. Madero, México, D.F.
E-mail: israelromang@hotmail.com

Received article: 06/junio/2011
Accepted article: 09/septiembre/2011

ABSTRACT

The identification and discrimination of functional sequences or mutations is very helpful in the medical area. Promoter and splice-junction identification, gene finding, DNA or Aminoacid database searching are some examples. Pattern recognition algorithms are candidates to perform this tasks. In this work we present a model, based on AlphaBeta associative memory and NeedlemanWunsch algorithm, to correctly recall altered version of learning patterns with one or more of the following modifications: insertions, deletions, and mutations, very common alterations in DNA and Aminoacid sequences. Moreover, this model preserve one of the most important advantages in associative memories, the correct recall of the fundamental set. To test the performance of the algorithm on bioinformatics and biomedical applications, the model presented here was tested using two datasets; one from the UCI repository; refered to promoter identification and the second one to using the genome of the *Variovorax paradoxus* organism obtained from the NCBI repository.

Key Words: Promoters, aminoacid sequences retrieval, DNA sequence retrieval, associative memories, biomedical, bioinformatics, Variovorax paradoxus.

RESUMEN

La identificación y discriminación de secuencias funcionales son de mucha ayuda en la investigación en el área biomédica. Identificación de promotores, identificación de zonas de empalme, búsqueda de genes y búsqueda de secuencias de ADN y aminoácidos en bases de datos son algunos ejemplos de aplicaciones en dicha área de investigación. Dada la naturaleza del problema, los algoritmos de reconocimiento de patrones son candidatos naturales para llevar a cabo las tareas antes mencionadas. En el presente trabajo se propone un nuevo modelo de memorias asociativas Alfa-Beta, basadas en el modelo original de memorias y el algoritmo global de alineamiento de secuencias desarrollado por Needleman-Wunsch, que permiten la recuperación de patrones alterados con respecto de los patrones de aprendizaje con alguna de las siguientes alteraciones: mutaciones, inserciones y borrados; alteraciones comunes en secuencias de DNA y aminoácidos. El presente modelo preserva una de las más importantes ventajas en memorias asociativas, la recuperación completa del conjunto fundamental. Para probar el desempeño del modelo en aplicaciones tanto de bioinformática como biomédica, se utilizaron dos bases de datos; una obtenida del repositorio de la Universidad de California en Irvine; sobre secuencias que contienen promotores y la

segunda del genoma del organismo *Variovorax paradoxus* obtenida del repositorio de la NCBI.

Palabras clave: Promotores, recuperación de secuencias de aminoácidos, recuperación de secuencias de ADN, memorias asociativas, bioinformática, Variovorax paradoxus.

INTRODUCTION

In the later decades, very important scientific advances in the field of molecular biology have been achieved. Thanks to the enormous ammounts of information derived from these advances, there has arisen a need to process such information in a faster manner and just as effectively, or more, than by an expert. This gives birth to a new branch of science, known as *Bioinformatics*: a multidisciplinary field which combines, among others, two important fields of science, *molecular biology* and *computer sciences*[1]. Its main objective is to apply the advantages offered by computer sciences to molecular biology, given the fast development, efficiency and efficacy of the algorithms of the former[2].

Among the first and foremost problems boarded by Bioinformatics are: the development of databases, protein sequence alignment, DNA string sequencing, protein structure prediction, protein structure classification, promoter identification, splice-junction zone localization, and filogenetic relationships determining[3,4].

Deoxyribonucleic acid (DNA) and *proteins* are biological macromolecules made up of long chains of chemical components. On one hand, DNA is made up of *nucleotides*, of which there are four: *adenine* (A), *cytosine* (C), *guanine* (G), and *thymine* (T), denoted by their initials. Also, DNA plays a fundamental role in different biochemical processes of living organisms, such as protein synthesis and hereditary information transmission from parents to children[5].

Promoters are the regions in the DNA that regulates the expression of the proteins and are regularly before each gene[6-8].

On the other hand, proteins are polypeptides formed inside cells as sequences of 20 different *aminoacids*[9], which are denoted by 20 different letters. Each of these 20 aminoacids is coded by one or more *codons*[5]. The chemical properties differentiating the 20 aminoacids make them group together to conform proteins with certain tridimentional structures, defining the specific functions of a cell[8].

Several diseases are generated by point mutations, insertions or deletions in the DNA. Thus, pattern recognition plays an important role in medical area. The recognition of the mutations leads to a better understanding of the disease and the development of new techniques and equipment. An example was the Influenza outbreak in México City, where the use of Bioinformatics gave important information about the pandemia, and lead to the development of different techniques and equipment for its identification and vaccination[10,11].

The topic of associative memories has been an active field of scientific research for some decades, attracting the attention in some research areas for the great power they offer despite the simplicity of its algorithms. The most important characteristic and, at the same time, fundamental purpose of an associative memory, is to correctly recall whole output patterns from input patterns, with the possibility of having the latter altered, either by an additive, subtractive, or mixed alteration[12-14]. This kind of alterations could be classified as mutation based on their definition, but the insertions and deletions was not treatable by this algorithm.

An associative memory has two phases: the learning phase, which is the process that allows the memory to be built by learning associations of patterns, and the recalling phase, in which the memory is presented with input patterns that can be present in the fundamental set or not, and the memory output, the corresponding associated pattern, according to the associations learned[13,14].

Associative memories, and specifically alpha-beta associative memories, are a powerful computational tool in pattern recognition due to the simplicity of their algorithms, their strong mathematical foundation, and the high efficacy shown by them in pattern recalling and classification[15].

In this paper we propose the use of a robust alpha-beta associative memory model for retrieval of sequences from a Aminoacid-database. This model has the capability of managing learning and recalling patterns of different dimensions. Moreover, this model can handle insertion, deletion and mutation in sequences, keeping its capability of complete

recall of the fundamental set. These characteristics are not present in the models found in our research of the state-of-art.

This paper is organized as follows. *Tools* is focused on explaining the Alpha-Beta heteroassociative memory model which is the main tool of this paper. *Robust Retrieval Associative Memory* contains the core proposal and its theoretical support. *Results* is devoted to the experimental results and finally, *Conclusion and Future work* addresses thoughts derived from this work.

TOOLS

Alpha-Beta associative memories

Here we introduce the basic notation of associative memories as presented in[15]. An associative memory $\mathbf{M}$ is a system that relates input and outputs patterns. Each input vector $\mathbf{x}$ forms an association with a corresponding output vector $\mathbf{y}$. The k -th association will be denoted as $(\mathbf{x}^k, \mathbf{y}^k)$. Associative memory $\mathbf{M}$ is represented by a matrix whose component in the i-th row and j-th column is denoted m_{ij}. The m_{ij} is generated from a set of *a priori* known associations, called the fundamental set. The fundamental set is defined as follows: $\{(\mathbf{x}^\mu, \mathbf{y}^\mu) \mid \mu = 1,2,..., p\}$ where p the cardinality of the fundamental set. The patterns that belong to the fundamental set are called fundamental patterns. If it holds that $\mathbf{x}^\mu = \mathbf{y}^\mu \ \ \forall \mu \in \{1,2,..., p\}$, then $\mathbf{M}$ is *autoassociative*, otherwise it is *heteroassociative*. In this latter case it is possible to establish that $\exists \mu \in \{1,2,..., p\}$ for which $\mathbf{x}^\mu \neq \mathbf{y}^\mu$. When feeding an unknown fundamental pattern $\mathbf{x}^\omega$ with $\omega \in \{1,2,..., p\}$ to an associative memory $\mathbf{M}$, it happens that the output corresponds exactly to the associated pattern $\mathbf{y}^\omega$, it is said that recall is correct.

The heart of the mathematical tools used in the Alpha-Beta model, are two binary operators designed specifically for these memories. These operators are defined in[15] as follows: First, it is defined the sets $A = \{0,1\}$ and $B = \{0,1,2\}$, then the operators $\alpha : A \times A \to B$ and $\beta : A \times B \to A$ are defined in Table 1.

Table 1. Alpha and Beta operators.

x	y	$\alpha(x,y)$	x	y	$\beta(x,y)$
0	0	1	0	0	0
0	1	0	0	1	0
1	0	2	1	0	0
1	1	1	1	1	1
			2	0	1
			2	1	1

There exist two types of heteroassociative Alpha-Beta memories, these are: type Max $(\vee)$ and type Min $(\wedge)$. The main difference of this two types is their tolerance to different kinds of alterations. For the generation of both types it will used the operator $\boxtimes$, which is defined as follows:

$$[\ \mathbf{y}^\mu \boxtimes (\mathbf{x}^\mu)^t\ ]_{ij} = \alpha(y_i^\mu, x_j^\mu) \text{ where}$$

$$\mu \in \{1,2,\ldots, p\}, i \in \{1,2,\ldots, m\}, \text{ and } j \in \{1,2,\ldots, n\}.$$

Alpha-Beta heteroassociative memories with correct recall

Alpha-Beta heteroassociative memories, unlike original and many other models[15,13], guarantee the correct recall of the fundamental set[16,17]. This section shows the Alpha-Beta heteroassociative memory type min, with which the complete recall of the fundamental set is guaranteed[16]. The Alpha-Beta heteroassociative memory type max is obtained by duality.

Alpha-Beta heteroassociative memory type min

Let Λ be an Alpha-Beta heteroassociative memory type Min and $\{(\mathbf{x}^\mu, \mathbf{y}^\mu) \mid \mu = 1,2,..., p\}$ its fundamental set with $\mathbf{x}^\mu \in A^n$ and $\mathbf{y}^\mu \in A^p, A = \{0,1\}, B = \{0,1,2\}, n, p \in Z^+$. The number of the components with value equal to zero of the i-th row of Λ is given by: $r_i = \sum_{j=1}^{n} T_j$ where $T \in B^n$ and its components are defined as:

$$T_i = \begin{cases} 1 & \leftrightarrow \quad \lambda_{ij} = 0 \\ 0 & \leftrightarrow \quad \lambda_{ij} \neq 0 \end{cases} \forall j \in \{1,2,..., n\} \qquad (1)$$

and the r_i components conform the min sum vector with $\mathbf{r} \in Z^p$[16].

a. Learning phase

Let $\mathbf{x} \in A^n$ and $\mathbf{y} \in A^p$ be input and output vectors, respectively. The corresponding fundamental set is denoted by $\{(\mathbf{x}^\mu, \mathbf{y}^\mu) \mid \mu = 1,2,..., p\}$. Which is built according with the following conditions: the $\mathbf{y}$ vectors are built with the *zero-hot* codification: assigning for the output binary pattern $\mathbf{y}$ the following values: $y_k = 0$, and $y_j = 1$ for

$j = 1,2,\ldots,k-1,k+1,\ldots,p$ where $k \in \{1,2,\ldots,p\}$. And, to each $\mathbf{y}^{\mu}$ vector correspond *one and only one* $\mathbf{x}^{\mu}$ vector.

For each $\mu \in \{1,2,...p\}$, from the pair $(\mathbf{x}^{\mu}, \mathbf{y}^{\mu})$ build the matrix: $[\ \mathbf{y}^{\mu} \boxtimes (\mathbf{x}^{\mu})^t\ ]_{p\times n}$ then the min binary operator $(\wedge)$ is applied to the resulting matrices. Therefore, the Λ matrix is obtained as follow:

$\Lambda = \bigwedge_{\mu=1}^{p} [\mathbf{y}^{\mu} \boxtimes (\mathbf{x}^{\mu})^t\ ]$ where the component in the i-th row and j-th column is given by:

$$\lambda_{ij} = \bigwedge_{\mu=1}^{p} \alpha(y_i^{\mu}, x_j^{\mu}).$$

b. Recalling phase

A pattern $\mathbf{x}^{\varpi}$ is presented to Λ the ∇_{β} operation is done and the resulting vector is assigned to a vector called $\mathbf{z}^{\varpi}$: $\mathbf{z}^{\varpi} = \Lambda \nabla_{\beta} \mathbf{x}^{\varpi}$. The i-th component of the resulting column vector are: $\mathbf{z}_i^{\varpi} = \bigvee_{j=1}^{n} \beta(\lambda_{ij}, x_j^{\varpi})$

It is necessary to build the *min sum vector* $\mathbf{r}$, therefore the corresponding $\mathbf{y}^{\varpi}$ is given as:

$$\mathbf{y}_i^{\varpi} = \begin{cases} 0 & if \quad r_i = r_k \quad \wedge \quad z_i^{\varpi} = 0 \\ 1 & otherwise \end{cases} \tag{2}$$

where $\theta = \{i \mid z_i^{\varpi} = 0\}$.

Robust retrieval associative memory

In this section, we describe a model, merging Alpha-Beta associative memories[16] and Needleman–Wunsch algorithm[18]. With this it is possible to handle input patterns of different sizes for both learning and recall phase, keeping the main property: correct recall of the fundamental set.

First at all is important to define this: Let $x^{\alpha} \in A^n$ with $A = \{0,1\}$, $n \in Z^+$, and $\alpha \in \{1,2,...,p\}$ be a row vector. Let $q \in Z^+$ be the dimension of the new smaller vectors extracted from x^{α}. The *vectorial partition operation* $\rho(x^{\alpha}, q)$ is defined as the set of binary row vectors q-dimensional and is denoted as follow:

$$\rho(x^{\alpha l}, q) = \left\{ x^{\alpha 0}, x^{\alpha 1}, ..., x^{\alpha \frac{n}{q}-1} \right\} \tag{3}$$

such that $x_r^{\alpha l} = x_{l+r}^{\alpha}$ where $x^{\alpha l} \in A^q$

with $l \in \left\{0,1,2,...,\frac{n}{q}-1\right\}$ and $r \in \{1,2,...,q\}$.

Robust Alpha-Beta heteroassociative memory type min

1. Learning phase

Let $\mathbf{x} \in A^n$ and $\mathbf{y} \in A^p$ with $n, p \in Z^+$, be an input and output vectors, respectively. The corresponding fundamental set is denoted by $\{(\mathbf{x}^{\mu}, \mathbf{y}^{\mu}) \mid \mu = 1,2,..., p\}$ such that $\exists \delta, \sigma \in \{1,2,..., p\}$ where $\mathbf{x}^{\delta} \in A^b$, $\mathbf{x}^{\sigma} \in A^c$ with $b, c \in Z^+$ and $b \neq c$. Moreover, the $\mathbf{y}$ vectors are built with the *zero-hot* codification: assigning for $\mathbf{y}^{\mu}$ the following values: $y_k^{\mu} = 0$, and $y_j^{\mu} = 1$ for $j = 1,2,\ldots,k-1,k+1,\ldots,p$ where $k \in \{1,2,\ldots,p\}$. And to each $\mathbf{y}^{\mu}$ vector correspond *one and only one* $\mathbf{x}^{\mu}$ vector.

For each $\mu \in \{1,2,...p\}$, from the couple $(\mathbf{x}^{\mu}, \mathbf{y}^{\mu})$ build the matrix: $[\mathbf{y}^{\mu} \boxtimes (\mathbf{x}^{\mu})^t\ ]_{p\times n}$ then, the min binary operator is applied to the matrices. Therefore, the Λ matrix is obtained as follow:

$\Lambda = \bigwedge_{\mu=1}^{p} [\mathbf{y}^{\mu} \boxtimes (\mathbf{x}^{\mu})^t\ ]$ where the ij-th component is given by: $\lambda_{ij} = \bigwedge_{\mu=1}^{p} \alpha(y_i^{\mu}, x_j^{\mu})$.

2. Recall phase

A fundamental pattern $\mathbf{x}^{\varpi}$, that can or not be of different size from other fundamental patterns, is presented to Λ, then the vector $\mathbf{y} \in A^p$ is built as follows:

First, the vectorial partition operation is applied to each $\lambda_i \in B^a$ and to $\mathbf{x}^{\varpi} \in A^b$ where α and β belong to the fundamental set.

$$\rho(x^{\varpi l}, q) = \left\{ x^{\varpi 0}, x^{\varpi 1}, ..., x^{\varpi \frac{b}{q}-1} \right\} \tag{4}$$

$$\rho(\lambda_i^l, q) = \left\{ \lambda_i^0, \lambda_i^1, ..., \lambda_i^{\frac{a}{q}-1} \right\} \tag{5}$$

Given $i \in \{1,2,..., p\}$, $c \in \left\{0,1,2,...\frac{a}{q}-1\right\}$, $h \in \left\{0,1,2,...,\frac{b}{q}-1\right\}$ with $C = \frac{a}{q}-1$ and $H = \frac{b}{q}-1$, the i-th component of the output vector $\mathbf{z}^{\omega}$ is:

$$z_i^{\omega} = \begin{cases} 0, & F_{CH}^i = \bigvee_{k=1}^{p} F_{CH}^k \quad and \quad F_{CH}^i > 0 \\ 1, & otherwise \end{cases} \tag{6}$$

then the $\mathbf{F}^i$ is the matrix built as follow:

$$F_{ch}^{i} = \bigvee \left(F_{c-1,h-1} + (S(\lambda_i^{h-1}, x^{\omega c-1}) * \eta), F_{c-1,h} + d, F_{c,h-1} + d \right) \quad (7)$$

where:

$$S(\lambda_i^{h-1}, x^{\omega c-1}) = \begin{cases} -1, & \bigvee_{j=1}^{q} \beta(\lambda_{ij}^{h-1}, x_j^{\omega c-1}) \neq 0 \\ 1, & otherwise \end{cases} \quad (8)$$

With $F_{c,0} = c * d, F_{0,h} = h * d$ with $d \in \mathbf{Z}^-$ being a penalization known as gap, and η being a factor to increase the result of Alpha-Beta memory recognition.

Once the z vector has been built, the *sum min vector* $r \in Z^p$ is computed. It contain in its $i-$ th component the amount of zeros of the i-th row of the Λ matrix.

$$r_i = \sum_{j=1}^{n} T_j \quad (9)$$

where $T \in B^n$ and its components are defined as:

$$T_i = \begin{cases} 1 & \leftrightarrow \quad \lambda_{ij} = 0 \\ 0 & \leftrightarrow \quad \lambda_{ij} \neq 0 \end{cases}$$

$\forall j \in \{1,2,...,n\}$. Therefore the corresponding $\mathbf{y}^{\varpi}$ is given as:

$$\mathbf{y}_i^{\varpi} = \begin{cases} 0 \quad \text{if} \quad r_i = \bigwedge_{k \in \theta} r_k \quad \wedge \quad z_k^{\varpi} = 0 \\ 1 \qquad \text{otherwise} \end{cases}$$

where $\theta = \{ i \mid z_i^{\varpi l} = 0 \}$.

Example 1: Let x^1, x^2, x^3, x^4 *be the input patterns*

$$x^1 = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix}, x^2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, x^3 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}, x^4 = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

and the corresponding output vectors

$$y^1 = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}, y^2 = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix}, y^3 = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \end{pmatrix}, y^4 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

the output vectors are built with the Zero-Hot codification, and to each output pattern corresponds *one and only one* input pattern, therefore the fundamental set is expressed as follow:

$$\{ (x^1, y^1), (x^2, y^2), (x^3, y^3), (x^4, y^4) \}$$

Once the fundamental set is made, the learning phase of the new algorithm is applied:

$$\begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \end{pmatrix} \boxtimes \begin{pmatrix} 1 & 0 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 2 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 1 & 2 & 1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} \boxtimes \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 2 & 1 & 2 & 2 & 2 \\ 1 & 0 & 1 & 1 & 1 \\ 2 & 1 & 2 & 2 & 2 \\ 2 & 1 & 2 & 2 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} \boxtimes \begin{pmatrix} 1 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 2 & 1 \\ 1 & 2 & 2 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 2 & 2 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \end{pmatrix} \boxtimes \begin{pmatrix} 1 & 1 & 0 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

The binary operator min $\bigwedge$ is applied to the matrices obtained before to build the matrix Λ:

$$\Lambda = \begin{pmatrix} 0 & 1 & 0 & 0 & \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Once the Λ matrix is generated, to recall x^{ϖ} with $\varpi \in \{1,2,3,...,p\}$, particularly $\varpi = 4$, x^4 is presented to Λ. First, the vectorial partition operator is applied to vector x^{ϖ} and λ_i with $q = 3$:

$$\rho(x^4,3) = \{(1,1,0),(1,0,1),(0,1,1)\}$$

$$\rho(\lambda^1,3) = \{(0,1,0),(1,0,0)\}$$

$$\rho(\lambda^2,3) = \{(1,0,1),(0,1,1),(1,1,1)\}$$

$$\rho(\lambda^3,3) = \{(0,1,1),(1,1,0)\}$$

$$\rho(\lambda^4,3) = \{(0,0,1),(0,1,0),(1,0,0)\}$$

Then, for each $i \in \{1,2,...,p\}$ and $d = -1$ and $\eta = 3$, the F^i matrices are built as follow:

$$F_{11}^1 = \vee(F_{0,0} + S(\lambda_1^0, x^{4(1-1)}) * 3, F_{0,1} + (-1), F_{1,0} + (-1)) = \vee(0+0*3,-1-1,-1-1) = \vee(0,-2,-2) = 0$$

$$F_{12}^1 = \vee(F_{0,1} + S(\lambda_1^1, x^{4(1-1)}) * 3, F_{0,2} + (-1), F_{1,1} + (-1)) = \vee(-1+0*3,-2-1,0-1) = \vee(-1,-2,-1) = -1$$

$$F_{21}^1 = \vee(F_{1,0} + S(\lambda_1^0, x^{4(2-1)}) * 3, F_{1,1} + (-1), F_{2,0} + (-1)) = \vee(-1+1*3,0-1,-2-1) = \vee(0,-1,-3) = 0$$

$$F_{22}^1 = \vee(F_{1,1} + S(\lambda_1^1, x^{4(2-1)}) * 3, F_{1,2} + (-1), F_{2,1} + (-1)) = \vee(0+0*3,-1-1,0-1) = \vee(0,-2,-1) = 0$$

$$F_{31}^1 = \vee(F_{2,0} + S(\lambda_1^0, x^{4(3-1)}) * 3, F_{2,1} + (-1), F_{3,0} + (-1)) = \vee(-2+0*3,0-1,-3-1) = \vee(-2,-1,-4) = -1$$

$$F_{32}^1 = \vee(F_{2,1} + S(\lambda_1^1, x^{4(3-1)}) * 3, F_{2,2} + (-1), F_{3,1} + (-1)) = \vee(0+1*3,0-1,-1-1) = \vee(3,-1,-2) = 3$$

Then, the matrix F^1 is:

$$F^1 = \begin{bmatrix} & & \lambda_1^0 & \lambda_1^1 \\ & 0 & -1 & -2 \\ x^{40} & -1 & 0 & -2 \\ x^{41} & -2 & 0 & 0 \\ x^{42} & -3 & -1 & 3 \end{bmatrix}$$

The calculus of each component for the matrices F^2, F^3, F^4 is not explicitly expressed, however the matrices are shown here:

$$F^2 = \begin{bmatrix} & & \lambda_1^0 & \lambda_1^1 & \lambda_1^2 \\ & 0 & -1 & -2 & -3 \\ x^{40} & -1 & 3 & 2 & 1 \\ x^{41} & -2 & 2 & 6 & 5 \\ x^{42} & -3 & 1 & 5 & 9 \end{bmatrix}$$

$$F^3 = \begin{bmatrix} & & \lambda_1^0 & \lambda_1^1 \\ & 0 & -1 & -2 \\ x^{40} & -1 & 3 & 2 \\ x^{41} & -2 & 2 & 6 \\ x^{42} & -3 & 1 & 5 \end{bmatrix}$$

$$F^4 = \begin{bmatrix} & & \lambda_1^0 & \lambda_1^1 & \lambda_1^2 \\ & 0 & -1 & -2 & -3 \\ x^{40} & -1 & 3 & 2 & 1 \\ x^{41} & -2 & 2 & 6 & 5 \\ x^{42} & -3 & 1 & 5 & 9 \end{bmatrix}$$

the resulting vector could be or not an output pattern from the fundamental set, in other words, it could be an ambiguous pattern. According with the recall phase, the resulting vector is known as z^4 then the *min sum vector r* must be built:

$$z^4 = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}, r = \begin{pmatrix} 3 \\ 1 \\ 2 \\ 4 \end{pmatrix}$$

after that, the output pattern y^4:

$$y^4 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

due to the minimum value of r_j where $z_j^4 = 0 \forall j \in \{1,2,3,4\}$ is 4.

RESULTS

This section reports an experimental study of the model. The experimentation was made for both aminoacid and DNA sequences. For aminoacid sequences the dataset was created from NCBI repository and for DNA sequences the datasource was obtained from the Machine Learning Repository of the University of California in Irvine[20]. The proposed model requires that η and q are given. A simple implementation of the model is used to test it. In the following test, $\eta = 5$ and $q = 1$ are used along some small-scale data sources.

As mentioned before the learning data sources of aminoacid sequences was obtained from the NCBI. The organism selected was *Variovorax paradoxus* S110 chromosome 1. In order to use the proposed model, it is necessary to relate the aminoacid characters into binary sequences, in Table 2 shows such relation. They were created using the known blosum62 substitution matrix[19]. Actually, there are two mappings, one for coding the learning patterns and other for recalling patterns. The mappings were built by sorting the 24 characters (20 aminoacids and 4 wildcard) and assigning to each one a twenty four dimensional vector, each component correspond to one of the 24 characters of the aminoacids. Then it was assigned a number one in the component where in the blosum62 matrix the character pair has a positive value, On otherwise. Its main objective is to give information to the model about the most probably changes between aminoacids. To experimentally show that this proposal fulfills the correct recall property of the fundamental set, we have built four different fundamental sets of cardinality p. With these fundamental sets the associative memories are built. Then, the same set of learning patterns are presented to the memory. The recall percentages are shown in Table 3.

Table 2. Relational table.

Aminoacid character	Binary learning sequence	Binary recalling sequence
F	100000010000000100000000	100000000000000000000000
S	011100000000001000000000	010000000000000000000000
T	011000000000000000000000	001000000000000000000000
N	010100000000000000111000	000100000000000000000000
K	000010100110000000000100	000010000000000000000000
*	000000000000000000000000	111111111111111111111111
E	000010100110000000101000	000000100000000000000000
Y	100000010000000100010000	000000010000000000000000
V	000000001001010000000010	000000001000000000000000
Z	000010100110000000101000	000000100000000000000000
Q	000010100110000000010100	000000000010000000000000
M	000000001001010000000010	000000000001000000000000
C	000000000000100000000000	000000000000100000000000
L	000000001001010000000010	000000000000010000000000
A	010000000000001000000000	000000000000001000000000
W	100000010000000100000000	000000000000000100000000
X	000000000000000000000000	011000000000001000000000
P	000000000000000001000000	000000000000000001000000
B	000100100100000000101000	000000000000000000101000
H	000100010010000000010000	000000000000000000010000
D	000100100100000000101000	000000000000000000001000
R	000010000010000000000100	000000000000000000000100
I	000000001001010000000010	000000000000000000000010
G	000000000000000000000001	000000000000000000000001

It is also important to know how the algorithm behave when altered patterns are presented to the associative memory. Based on a fundamental set of the file p50.txt, six altered versions of it were built. The alterations of the fundamental set are shown in Table 5 and the results are shown in Table 4.

Table 5 shows the alteration made randomly to the original fundamental set.

1. Alteration: indicates the percentage of changes in the sequence. The changes could be mutation, insertion, and deletion
2. Mutation: percentage of substitution of an aminoacid by other
3. Insertion: percentage of insertion of an aminoacid in the sequence
4. Deletion: percentage of deletion of an aminoacid in the sequence

In the other hand, to test the performance of the model with DNA sequences, the promoters and splice-junction samples were taken from the «*E. coli* promoter gene sequences (DNA) with associated imperfect domain theory» and «Primate splice-junction gene sequences (DNA) with associated imperfect domain theory» datasources, consecutively.

The promoter database has 106 instances split into two classes, promoters and non-promoters, 53 instances to each one. The sequences are formed by 57 nucleotides and its binary codification is shown in Table 6. According with[21] the One-Hot codification is one of the most benefical.

Table 7 shows the percentage of recall on DNA datasets altered with some percentage of the alteration defined before. It is clear that, even when the alterations change the original sequence in both composition and dimension, the new model support this kind of modification and preserve its recall capacity.

Table 8 shows the alterations made to the DNA sequences from the original datasource.

Finally, it could be interesting to use the experimental datasources with the original model of associative memories. However, by the nature of the original model it is not possible due to the fact that the input and output vectors should be all the same dimension.

Table 3. Correct recall of the fundamental set.

Data source	*p*	% recalled
p50.txt	50	100
p100.txt	100	100
p150.txt	150	100
p200.txt	200	100

Table 4. Altered pattern recall.

Data source	*p*	% recalled
P70B50M50.txt	50	2
P70M50I50.txt	50	0
P70M100.txt	50	100
P90M100.txt	50	100
P90B50M50.txt	50	94
P90M50I50.txt	50	94

Table 5. Percentage of alteration to patterns per file.

Data Source	% alteration	% mutation	% deletion	% insertion
P70B50M50.txt	30	50	50	0
P70M50I50.txt	30	50	0	50
P90B50M50.txt	10	50	50	0
P90M50I50.txt	10	50	0	50

Table 6. DNA binary codification.

Nucleotide	Code
A	1000
T	0100
C	0010
G	0001
D	1011
N	1111
S	0101
R	1001

Table 7. Percentage of DNA sequence recall.

Data source	% recalled
Promoter.txt	100
PromoterP70M100.txt	100
PromoterP90M50I50.txt	98
PromoterP90M100.txt	100
PromoterP90M50B50.txt	100

Table 8. Percentage of alteration to patterns per file.

Data Source	% alteration	% mutation	% deletion	% insertion
Promoter.txt	0	0	0	0
PromoterP70M100.txt	30	100	0	0
PromoterP90M50I50.txt	10	50	0	50
PromoterP90M50B50.txt	10	50	50	0

CONCLUSION AND FUTURE WORK

In this work, a model for retrieval of aminoacid and DNA sequences from a data sources is proposed. The model ensures the correct recall of the fundamental set, this is the complete set of patterns learned. Moreover, unlike previous models of associative memories, it is capable of supporting some degree of the three type of alterations on the patterns: *mutation*, *deletion*, and *insertion*, as shown on Table 4. To do so, a relational table for aminoacids character to binary sequences is proposed. It is possible to use this model in any medical task that requires analysis of DNA or aminoacid sequences; no matter if some sequences has alterations. This model is capable of handling patterns of different sizes for learning and recall.

As future work it is important to develop an efficient software that implements the given model. It might be helpful to use the heteroassociative memory type Max to compare the advantages and disadvantages against the proposed model. Test a modified version of the model using the Smith-Waterman algorithm for local alignment. Develop experiments with several ranges of evolutionary proximity.

ACKNOWLEDGEMENTS

The authors would like to thank the Instituto Politécnico Nacional (Secretaría Académica, COFAA, SIP, and CIC), the CONACyT, the SNI, and the ICyTDF (grants PIUTE10-77 and PICSO10-85) for their economical support to develop this work.

REFERENCES

1. Baldi P, Brunak S. Bioinformatics: the machine learning approach. MIT Press, (Cambridge) 2001.
2. Von Heijne G. Sequence analysis in molecular biology: Treasure trove or trivial pursuit. Academic Press (London), 1987.
3. Doolittle RF. Of URFs and ORFs: A primer on how to analyze derived amino acid sequences. University Science Books, Mill Valley (California) 1986.
4. Wolfsberg TG, Wetterstrand KA, Guyer MS, Collins FS, Baxevanis AD. «A user's guide to the human genome». Nature Genetics 2002; 32: 1-79.
5. Setubal J, Meidanis J. Introduction to computational molecular biology. International Thomson Publishing (Boston, MA), 1999.
6. Lesk AM. Introduction to Bioinformatics. Oxford University Press, 2008.
7. Ray SS, Bandyopadhyay S, Mitra P, Pal SK. «Bioinformatics in neurocomputing framework». Circuits, Devices and Systems, IEE Proceedings 2005; 152(5): 556–564.
8. Mitra S, Hayashi Y. «Bioinformatics with soft computing». Systems, man, and cybernetics, Part C. Applications and Reviews, IEEE Transactions 2006; 36(5): 616-635.
9. Salzberg SL, Searls DB, Kasif S. Computational methods in molecular biology. Elsevier Science 1998.
10. Zepeda HM, Perea-Araujo L, Zarate-Segura PB, Vázquez-Pérez JA, Miliar-García A, Garibay-Orijel C, Domínguez-López A, Badillo-Corona JA, López-Orduna E, García-González OP, Villasenor-Ruiz I, Ahued-Ortega A, Aguilar-Faisal L, Bravo J, Lara-Padilla E, García-Cavazos RJ. Identification of influenza: A pandemic (H1N1) 2009 variants during the first 2009 influenza outbreak in Mexico City. Journal of Clinical Virology: the Official Publication of the Pan American Society for Clinical Virology 2010; 48: 36-39.
11. Zepeda-López HM, Perea-Araujo L, Miliar-García AA, Domínguez-López B, Xoconostle-Cazarez E, Lara-Padilla JA, Ramírez-Hernández E, Sevilla-Reyes ME, Orozco A, Ahued-Ortega I, Villasenor-Ruiz RJ, García-Cavazos, Teran LM. Inside the outbreak of the 2009 influenza A (H1N1) virus in Mexico. PloS One 2010; 5: e13256.
12. Hassoun MH. Associative neural memories: Theory and implementation. Oxford University Press (New York), 1993.
13. Ritter GX, Sussner P, Díaz-de-León JL. «Morphological Associative Memories». IEEE Transactions on Neural Networks 1998; 9(2): 281-293.
14. Hopfield JJ. «Neural networks and physical systems with emergent collective computational abilities». Biophysics 1982; 79: 2554-2558.
15. Yáñez-Márquez C. Associative memories based on order relations and binary operators. PhD Thesis, Center for Computing Research, National Polytechnic Institute, Mexico, D.F., 2002.
16. Román-Godínez I, Yáñez-Márquez C. «Complete Recall on Alpha-Beta Heteroassociative Memory». Lecture Notes. Computer Science 2007; 4827: 193-202.
17. Román-Godínez I, López-Yáñez I, Yáñez-Márquez C. «Classifying patterns in bioinformatics databases by using Alpha-Beta associative memories». In: Amandeep S, Sidhu-Tharam SD, editors. Biomedical Data and Applications in Studies in Computational Intelligence. Springer 2009; 187-210.
18. Needleman SB, Wunsch CD. «A general method applicable to the search for similarities in the amino acid sequence of two proteins». Journal of Molecular Biology 1970; 48(3): 443-453.

19. Henikoff S, Henikoff JG. «Amino acid substitution matrices from protein blocks». Proc Natl Acad Sci U SA 1992; 89(22): 10915-10919.
20. Asuncion A, Newman DJ. UCI Machine Learning Repository, Irvine, CA: University of California, Department of Information and Computer Science. Available at: http://www.ics.uci.edu/~mlearn/MLRepository.html
21. Brunak S, Engelbrecht J, Knudsen S. «Prediction of human mRNA donor and acceptor sites from the DNA sequence». J Mol Biol 1991; 220: 49-65.