A Comparative Study of Adaptive and Evolutionary Fuzzy Cognitive Maps

Wojciech Froelich\textsuperscript{1} and Przemyslaw Juszczuk\textsuperscript{2}

\textsuperscript{1} Institute of Computer Science, Silesian University, ul.Bedzinska 39, Sosnowiec, Poland, wojciech.froelich@us.edu.pl
\textsuperscript{2} Institute of Computer Science, Silesian University, ul.Bedzinska 39, Sosnowiec, Poland, przemyslaw.juszczuk@gmail.com

Abstract. Fuzzy cognitive maps (FCMs) represent the decision process in the form of a graph that is usually easy to interpret and, therefore, can be applied as a convenient decision-support tool. In the first part of this chapter, we explain the motivations for the research on FCMs and provide a review of the research in this area. Then, as stated in the title of the chapter, we concentrate our attention on the comparative study of adaptive and evolutionary FCMs. The terms adaptive and evolutionary refer to the type of learning applied to obtain a particular FCM. Despite many existing works on FCMs, most of them concentrate on one type of learning method. The purpose of our research is to learn FCMs using diverse methods on the basis of the same dataset and apply them to the same prediction problem. We assume the effectiveness of prediction to be one of the quality measures used to evaluate the trained FCMs. The contribution of this chapter is the theoretical and experimental comparison of adaptive and evolutionary FCMs. The final goal of our research is to determine which of the analyzed learning methods should be recommended for use with respect to the considered prediction problem. Additionally, due to the available scope of the book chapter, we provide some technical details that can be useful for the implementation of FCM-based prediction systems.

1 Problem formulation

For the comparative analysis of adaptive and evolutionary FCMs, we will consider a prediction problem formulated in the following way. Let $T = \{t_0, t_1, \ldots, t_n\}$ be an ordered set of time labels, where $\forall i. t_i - t_{i-1} = \Delta t$ is a constant parameter that will be referred to as “base time.” Let $C$ represent the set of symbolic labels pointing to the subsets of observations (concepts) acquired from the raw, temporal data. Every concept label $c_i \in C$ is uniquely identified by its subscript. For every concept label, we define a mapping $a : T \times C \rightarrow [0, 1]$ that assigns the state (activation level) of a concept at time $t$. The state of the $i^{th}$ concept at time step $t$ will be denoted as $a_i(t) = a(t, c_i)$. The concepts for which $a_i(t) > 0$ will be referred to as “active concepts at time $t$.” The $A(t)$ denotes the state vector for all concepts. Let us assume now that the set $T$ is divided into the known historical
sequence (pointing to learning data), $T_H = \{t_0, t_1, \ldots, t_{s-1}\}$, and the unknown sequence representing the unknown future, $T_F = \{t_s, t_{s+1}, \ldots, t_e\}$ (pointing to testing data), where $\text{card}(T_H) = \text{card}(T_F)$. The sequence $T_H$ constitutes the time horizon that will be used to learn a model. Let us assume that the state of concepts at (starting) time $A(t_s)$ can be observed. We would like to forecast the state of concepts $A(t)$ at some time $t_e > t_s$ by the reconstruction of the unknown sequence $A(t_s), A(t_{s+1}) \ldots A(t_e)$, where the length of the prediction horizon $t_e - t_s = t_{\text{pred}}$ is a constant parameter. The prediction will be produced using a model based on FCM. On the basis of the prediction errors, we will estimate the quality of the obtained FCMs and, indirectly, the effectiveness of the applied learning method. For the purposes of this research, we make the simplification (as is usually done by researchers) of assuming that the concept labels are simply the identifiers of attributes (columns) within the data table stored in the relational database.

2 An introduction to the theory of cognitive maps

The inspiration for the development of cognitive maps originated from biological experiments [8, 20]. The initial model of a cognitive map, originally represented by a graph, was proposed and applied in the social sciences [3]. The entire graph was prepared by domain experts and represented the structure of dependencies between concepts. The nodes within the graph were annotated in natural language and represented intuitively-understood concepts. The arcs represented a binary relationship labeled with “+” or “−” signs, denoting intuitively-understood, positive or negative causality (cause-and-effect relationship), respectively. The intuitively-understood notions of concepts and the cause-and-effect relationship among them could not be used for the construction of any computational decision support system, therefore were attempts to formalize this idea. For the purpose of this chapter, we define a cognitive map as an order pair:

$$CM = < C, W >,$$

where $C$ is the set of concept labels, and $W$ is the connection matrix representing the relationship between concepts. The concepts are understood as described in our problem formulation. In simplest case, it is possible to distinguish binary cognitive maps (BCM) for which the concept labels are mapped to binary states $a_i(t) \in \{0, 1\}$. The value 0 means no activation, and the value 1 means full activation of a concept. The nodes will be indexed by subscripts $i$ (cause node) and $j$ (effect). The matrix $W$ stores the weights assigned to the pairs of concepts. The weights of BCM are usually mapped to the crisp set, i.e., $w_{ij} \in \{-1, 0, 1\}$. The value 1 represents, positive causality, e.g. as described by Huerga [5], that the activation (change from 0 to 1) of concept $c_i$ occurs concurrently with the same activation of concept $c_j$ or that deactivation (change from 1 to 0) of $c_i$ occurs concurrently with the same deactivation of concept $c_j$. The value $−1$ represents the opposite situation, in which the activation of $c_i$ deactivates the concepts $c_j$ or vice versa. The $w_{ij} = 0$ means that there are no concurrently occurring
changes of the states of the concepts. Some researchers assume that a concept cannot have impact on itself and that the elements on the diagonal of matrix $W$ are equal to 0.

The most controversial problem is the interpretation of weights between concepts. Even the intuitive differentiation of positive or negative causality (as has been proposed in generic FCM models) can raise many doubts. As we will show later, the computational interpretation of FCM weights is usually done in two phases: 1) during the exploitation of FCM - the weights substantially influence the reasoning process and 2) during the learning process - the weights are interpreted differently, depending on the method applied to automatically obtain the FCM.

The extension of CM in the form of fuzzy cognitive maps was proposed by Kosko [7]. In FCM, the concepts are represented by fuzzy sets, and the concepts labels are mapped to a real-valued activation level from the closed continuous interval $a_i(t) \in [0, 1]$, where 0 means no activation and 1 means full activation. The arcs between concepts are also labeled with real-value weights $w_{ij} \in [-1, 1]$.

The state of the FCM at time $t$ is fully described by the state vector $A(t)$. In most known approaches to learning FCMs, the set of concept labels $C$ is provided a-priori by expert, and only the matrix $W$ is learned from raw data. The example

\begin{figure}[h]
\centering
\includegraphics[width=0.8\textwidth]{fcm.png}
\caption{Example of a fuzzy cognitive map: a) the graphical representation b) the connectivity matrix $W$}
\end{figure}

of FCM is shown in Fig. 1. There have been numerous enhancements to the initial model of FCMs. The addition of memory to the concept nodes within the HO-FCMs [16] enables the representation of high-order dynamics of the modeled decision process. The rule-based fuzzy cognitive maps RBFCM [4] assigned rules to the arcs of FCMs, significantly enhancing their reasoning capabilities. There
IV

are also the extensions that enable the incorporation of the variable time delays between concept activation in the FCMs [14]. FCMs can be applied in economics (e.g., stock market prediction), medical science (e.g., medical diagnosis) [19], or even intrusion detection systems [1].

As mentioned before, FCMs can be learned from raw data. Diverse adaptive learning methods have been applied by many authors [9], and the most recent proposals involve Hebbian learning [10] and non-linear Hebbian learning [13]. Evolutionary learning methods have been investigated by many researchers [21]. A successful application of an evolutionary FCM to the prediction problem is presented in [15]. A survey of research papers on FCMs can be found in [2].

3 Forward reasoning with the use of FCM

The standard exploitation form of FCM is usually the same for most applications and is independent of the applied learning method. The goal of forward reasoning is the simulation of the temporal behavior of FCM in order to reproduce the unknown sequence of the state vectors \( A(t_s), A(t_{s+1}), \ldots, A(t_e) \). The exploitation of a given FCM starts from the acquisition of a new observation of concept activations \( A(t_s) \) and then performing a simulation (understood as a kind of numerical forward reasoning) of the modeled process. The multi-step reasoning process, which will be described in detail later, allows to predict future concept activation values.

There are many papers that describe the reasoning process in FCMs, however, to our knowledge, there is no one that explicitly specifies their diverse variants. The justification of the applied reasoning method is usually done experimentally. In almost all cases, the researchers use the scaled summation of factors, each representing the linear dependency of concept activations. For all computational experiments described in this chapter, we chose to use for reasoning in FCM the equation (2):

\[
a_j(t+1) = \gamma(\sum_{i=1, i \neq j}^{n} w_{ij}a_i(t)),
\]

where \( \gamma(x) = 1/(1 + e^{-cx}) \) is the threshold function that serves to confine unbounded values to a strict range, \( c \) is a constant (usually \( c = 5 \)), and \( n = \text{card}(C) \). Some of the researchers report the use of equation (3), involving a full impact of every concept on itself:

\[
a_j(t+1) = \gamma(a_j(t) + \sum_{i=1, i \neq j}^{n} w_{ij}a_i(t)),
\]

The other possibility is to use an arbitrarily assumed scaling coefficient, \( k < 1 \), as is done in equation (4):

\[
a_j(t+1) = \gamma(ka_j(t) + \sum_{i=1, i \neq j}^{n} w_{ij}a_i(t)),
\]
or use the non-zero diagonal values of matrix $W$ and apply equation (5):

$$a_j(t + 1) = \gamma(\sum_{i=1}^{n} w_{ij}a_i(t)),$$

The simulation of the FCM-based reasoning process can lead to three types of behavior of the state vector $A(t)$:

1. fixed-point attractor, known also as hidden pattern (the state vector becomes fixed after some simulation steps);
2. limit cycle (the state vector keeps cycling); and
3. chaotic attractor (the state vector changes in a chaotic way).

The type of behavior of state vector depends on the characteristics of the raw data and on the algorithm applied during the learning phase of the FCM.

4 Learning FCM

As mentioned before, it is possible to distinguish two general approaches to learning FCMs, i.e., adaptive and evolutionary.

4.1 Adaptive learning of FCM

The main idea of the most adaptive learning methods of FCM is to modify its weights incrementally as the new learning data in $T_H$ become available. For the purposes of this chapter, we recall and use the two best-known adaptive algorithms. Historically, the first approach was inspired by the Hebbian algorithm used for learning artificial neural networks. It was adapted by Kosko [7] to FCMs in the form of the differential Hebbian learning (DHL) algorithm. Assuming that the $i^{th}$ causal concept changes, i.e., $\Delta a_i \neq 0$. The DHL algorithm modifies the weight between concepts $i$(cause) and $j$(effect), using the equation (6):

$$w_{ij}(t + 1) = w_{ij}(t) + c(t)[\Delta a_i \Delta a_j - w_{ij}(t)]$$ (6)

If $\Delta a_i = 0$, then the corresponding weight is not changed, i.e., $w_{ij}(t + 1) = w_{ij}(t)$. The coefficient $c(t)$ used in equation (6) plays a key role during adaptive learning. It can be assumed that:

$$c(t) = 0.1[1 - t/1.1q]$$ (7)

changes over time [5]. The constant parameter $q \in \mathbb{R}$ should ensure that the value of the weight holds in interval $[-1, 1]$. Therefore, $q = card(T_H)$ can be assigned to the number of steps of the learning period. Notice that using equation (6) updates the weight between two concepts only on the basis of changes in their activation values. The drawback of such an assumption, i.e., its negative influence on predictive capabilities of BCM, was shown in [5]. The balanced differential learning algorithm (BDA) [5] takes into account the changes of activation values of other nodes (not directly connected by the considered arc). The equations
given in [5] adapted to our notation (with the exchange of cause and effect indices) look as follows. Let us assume that \( n = \text{card}(T_H) \) is the number of temporal steps in the training set. The diagonal of matrix \( W \), which reflects the self-impact of concepts, is computed as in equation (8):

\[
w_{jj}(t+1) = w_{jj}(t) + \frac{a_j(t)}{n} \tag{8}
\]

For two different concepts, the update equations look as follows.

For \( \Delta a_i \Delta a_j > 0 \):

\[
w_{ij}(t+1) = w_{ij}(t) + c(t) \left[ \frac{\Delta a_j/\Delta a_i}{\sum_{k=1}^{n} \frac{\Delta a_j/\Delta a_k}{\Delta a_j/\Delta a_k}} - w_{ij}(t) \right] \tag{9}
\]

For \( \Delta a_i \Delta a_j < 0 \):

\[
w_{ij}(t+1) = w_{ij}(t) + c(t) \left[ \frac{-\Delta a_j/\Delta a_i}{\sum_{k=1}^{n} \frac{\Delta a_j/\Delta a_k}{\Delta a_j/\Delta a_k}} - w_{ij}(t) \right] \tag{10}
\]

The process of learning is thus extended to the scope of the entire FCM. In our opinion, one of the other important enhancements of the BDA algorithm is that it involves the factors in form of division \( \Delta a_j/\Delta a_i \) instead of multiplication, \( \Delta a_i \Delta a_j \), as used in equation (6). The explanation of this opinion follows later in this chapter. As opposed to Huerga [5], who used the BDA algorithm for binary cognitive maps, we decided to try it for learning FCMs.

### 4.2 Evolutionary algorithms

The other branch of computational methods for learning FCMs involves the application of evolutionary algorithms. For our purposes in this chapter, we chose to recall and use the RCGA [21] and differential evolution (DE) [18] algorithms. Both require the transformation of connection matrix \( W \) to the chromosome vector, which can be done by taking the following rows from \( W \), as shown in Fig. 2. The number of genes in a single genotype can be calculated in accordance with the formula: \( \text{Dim}(W) = n \cdot (n-1) \), where: \( n \) is the number of concepts in FCM. According to the definition of FCM, the values of each individual gene are limited to the range \( x \in [0, 1] \).

The RCGA algorithm applies a real-coded genetic algorithm to develop FCM from the set of historical data. In each iteration of the algorithm, the historical data are compared with the data generated by the FCM model. In RCGA, each chromosome is a vector of floating point values that correspond to FCM weights. The core of this algorithm is the fitness function, which can be assumed as in equation (11), given in [21]. This is the sum of the differences between the activations of concepts generated from the network, and the values taken from historical data.

\[
f = \frac{1}{(t_e - 1) \cdot n} \cdot \sum_{t=1}^{t_e} \sum_{i=1}^{n} |a_i(t) - a_i'(t)|^p, \tag{11}
\]

where:
- $t_l$ is the lower bound of the time window and specifies the initial index of the learning data,
- $t_u$ is the upper bound of the time window and determines the final index of the learning data,
- $n = \text{card}(C)$ is the number of concepts,
- $p$ is the constant parameter; for our experiments we assumed $p = 1$,
- $a_n(t)$ is the observed value of the $i^{th}$ concept activation at the time moment $t$, and
- $a_n'(t)$ is the activation value of the $i^{th}$ concept at the time moment $t$ generated by the FCM.

![Diagram of concepts and connections]

**Fig. 2.** Creation of the genotype on the basis of connection matrix

It is usually convenient to scale the fitness function 11 using equation (13) so that the lower bound of its value set is closer to value of 1.

$$Fit = \frac{1}{(100 \cdot f) + 1}$$  \hspace{1cm} (12)

One of the main advantages of the differential evolution method [17] is its ability to designate the optimal direction and speed of convergence on the basis of the position of individuals in the current population. It is necessary to initialize the initial population $P_0$, providing an even distribution of individuals in exploration space. Mutation used in the DE is a much more complex process than is the use of traditional genetic algorithms, but, thanks to the use of a mechanism known as differential vectors, it is possible to perform a directed search.

Let us denote $X_{ig}$ as the $i^{th}$ genotype within the population of the $g^{th}$ generation. Let the constant $n$ represent the number of genotypes, and $m$ is the number of genes in the genotype. The following algorithm describes the basic steps of the differential evolution [17] algorithm.

1. Create the initial population of genotypes $P_0 = \{X_{1,0}, X_{2,0}, \ldots, X_{n,0}\}$,
2. Set the generation number $g = 0$,
3. When the stop criterion is not met:
VIII

(a) Compute the fitness function for every genotype in the population 
\{f(X_{1,g}), f(X_{2,g}), ..., f(X_{n,g})\}
(b) Create the population of trial genotypes (algorithm 2) \(V_g\) based on \(P_g\)
(c) Make crossover of genotypes from the population \(P_g\) and \(V_g\) to create population \(U_g\)
(d) Choose the genotypes with the highest fitness function from population \(U_g\) and \(P_g\) for the next population
(e) If \(g = g + 1\), go to step a.

The subroutine called in step 3b refers to the creation of trial genotypes, and it is described as follows:

1. Set the \(\beta\) parameter
2. For \(i\) to \(n\):
   (a) Generate two different random numbers \(r_1, r_2\) between range \((1, ..., n)\)
   (b) for \(j\) to \(m\), do
       \[v_{ij} = v_{ij} + \beta \cdot (v_{r_1j} - v_{r_2j}),\]

The parameter \(\beta \in [0, \infty]\) specifies the strength of impact of the difference vector (between the two genotypes from the population \(P_g\)) on the value of the newly-created genotype belonging to population \(V_g\), and the numbers \(r_1\) and \(r_2\) determine the indices of the genotypes; \(j\) is the index of the selected gene; and \(v_{ij}\) is a new value of the \(i^{th}\) gene of the \(j^{th}\) genotype.

We would like also to mention here two other approaches to evolutionary learning of FCMs: the particle swarm optimization algorithm [11, 12], and the algorithm proposed by Khan and Chong [6].

5 Theoretical comparison of adaptive and evolutionary FCMs.

In this section, we present some discussion of the theoretical aspects of adaptive and evolutionary FCMs. We start from the analysis of their predictive capabilities, taking into account the problem stated in the first section of this chapter.

Let us notice here a trivial case that can lead to better understanding of equations (2) through (5). Let us temporarily assume that the scaling function \(\gamma(x) = 1\) is trivial and analyze the mutual impact of only two different concepts without self-dependency (i.e., the weights on the diagonal of \(W\) are equal to 0). Thus, independent of whether we apply equation (2), (3), (4), or (5), we use a simple linear dependency for reasoning between concept activations: \(a_j(t + 1) = w_{ij}a_i(t)\). On the other hand, we would like to compute the weight \(w_{ij}\) on the basis of raw data available in two consecutive time steps, \(< t_1, t_2 >\). In the case of linear dependency, we can do this easily by using the equation (13):

\[w_{ij} = \frac{\Delta a_j}{\Delta a_i}\]  \(13\)
where: $\Delta a_j = a_j(t_2) - a_j(t_1)$, $\Delta a_i = a_i(t_2) - a_i(t_1)$. For the causal concept with $i$th subscript, we assume $\Delta a_i \neq 0$. This way, we achieve a simple FCM - trivial but perfect from the point of view of one-step prediction at time step $t_1$. Of course, the model can be valid for the next time steps $t_2, t_3, \ldots, t_n$ only when assuming the existence of the same linear dependency between the activations of the two concepts. Another notice is that, if the activation of $i$th concept grows and $w_{ji} > 0$, then the activation of $j$th concept also increases. Moreover, the application of scaling function $\gamma(x) = \frac{1}{1+e^{-cx}}$ does not change this fact. The conclusion is that, if the effect concept should drop and we want to achieve this using any one of equations 2 through 5, we need a negative impact of other concepts (with negative value of weights). However, this negative influence, in turn, will change the original perfect prediction achieved for time steps $t_1, t_2$. Now, you can see how the weights of FCM are mutually dependent and that their values can be difficult to determine considering only two concepts.

5.1 The temporal aspects of adaptive and evolutionary types of learning

The adaptive algorithm based on DHL method operates in the current learning step on only two concepts. The difference between the activation values of the concept in the current and previous moments of time is used. Let us look once more at equation (6) used in the DHL learning algorithm, and let us consider another trivial case. If the learning coefficient $c(t) = 1$, equation (6) would assume the form as in Equation (14):

$$w_{ij}(t + 1) = \Delta a_i \Delta a_j$$

(14)

Thus, we would lose the incremental learning feature of 6. The weight $w_{ij}$ assumes a value that is dependent only on the last two time steps from the learning sequence $T_H$. If we assume that the activation values of both concepts change equally and quite strongly, e.g., $\Delta a_i = 0.5$ and $\Delta a_j = 0.5$, the maximal impact (assuming $c(t) = 1$) of this single change $\Delta a_i \Delta a_j = 0.25$ on $w_{ij}$ is only a quarter as high as the theoretical weight computed using equation (13), namely $\frac{\Delta a_i \Delta a_j}{\Delta a_i + \Delta a_j} = 0.5 = 1$. The application of $c(t) < 1$ will make this influence even weaker.

Now, you can see clearly that the intention of using 6 and the value of $c(t) < 1$ is to perform a gradual modification of the weight $w_{ij}$ as the new data are observed. The lower the $c(t)$ coefficient is, the weaker is the influence of a singular change of concepts’ activation differences on the weight $w_{ij}$. At the same time, the longer the temporal interval is that affects the learning process. If the coefficient $c(t)$ changes in time, the adaptive learning algorithms become specifically sensitive to the order of raw data. If the same changes of concepts’ activations occur concurrently earlier in the sequence $T_L$, then the corresponding weight modification is stronger.

Using evolutionary algorithms for learning FCM, it is possible to set the time window within which the FCMs are evaluated. In fact, in each iteration of the algorithm, the difference between the concepts’ activations generated by the
network and the activations of the concepts observed on the basis of learning data are compared. The temporal aspects of time windows for adaptive and evolutionary types of learning are briefly presented in Fig. 3.

5.2 The dynamics of the learning process

According to the previous analysis, it can be noticed that, within the first few steps of the adaptive learning algorithm, large changes in network weights are more likely to occur. Further iterations are used to gradually stabilize the values of the weights of FCM. The changes in concept activations that occurred earlier have greater influence on the FCM weights than the changes in activations during the final stages of the learning period.

In comparison, evolutionary algorithms use the fitness function to evaluate the quality of the entire FCM. In the following iterations, the value of the fitness function decreases to zero, which means it approaches the global optimum. However, a small improvement of the evaluated FCM very often means large fluctuations of the network weights.

The evolutionary method allows the assessment of the quality threshold and makes possible the cessation of the learning process when the desired threshold is reached. In adaptive methods, the evaluation of FCM is possible only after the entire process of learning is finished. This has to do with the fact that, in
adaptive methods, there is no function that is used for the evaluation of the FCM that can examine the difference between the solution obtained and the expected optimum solution.

While learning large FCMs with only a few concepts changing at some time step, the adaptive algorithms theoretically have some advantage over the evolutionary technique. The adaptive algorithm modifies the mutual weight for the arc between two concepts that have changed their values. In contrast, when evolutionary algorithms are applied, all the weights of the FCM are processed (even if only one concept value has changed). This is due to the application of crossover and mutation operators (used in genetic and differential evolution algorithms), which operate on the entire population of genotypes. As a result of genetic operators, completely new individuals are produced - each of which may have different values of genes in the genotype.

5.3 Sensitivity to incompleteness of data

For both types of learning methods, the data must be complete. Especially in case of adaptive algorithms, note that the lack of information on concept activation value at a particular time step does not imply that the concept value has not changed.

One of the simplest possible solutions to this situation is to supplement the learning data. The concepts that are not observed at a given moment are transcribed from the previous time step. The situation is shown in Fig. 4, and the missing values are denoted by the letter 'X.' However, the intuitively understood confidence in such ‘repair’ can be limited and can cause a decrease in the quality of the obtained FCM.

5.4 Complexity of learning

The complexity of the problem can be reduced by providing the values of some weights by an expert. It allows the reduction of the dimensions of the search space. In the case in which we know nothing about FCM, we must assume that increasing the number of concepts significantly increases the number of
connections between them. Assuming that we know nothing a-priori about the desired FCM (and especially do not know the connections between concepts), the FCM is initially represented by a full graph. In this case, the number of edges is $n^2$, where $n = \text{card}(C)$ is the number of concepts.

The main drawback of the evolutionary methods is the time required to generate the solution. This is due to the fact that, in each iteration, the algorithm evaluates $m$ number of FCMs (where $m$ is the number of genotypes in the population). The total number of cognitive networks created in the whole algorithm is $m \cdot g$ (where $g$ is number of generations). The result of a smaller size population or reducing the size of the time-window is the learning time is shortened. Unfortunately, this reduces the quality of the obtained FCM.

The adaptive methods do not have restrictions related to the amount of concepts in the FCM. Using adaptive methods, only one FCM is generated, and its weights are then modified in subsequent learning steps. The function used to modify weights assigned to the edges is less complicated than the fitness function used in evolutionary methods. In addition, there are no operations, such as crossover, mutation, or selection, that increase the time needed to find the satisfactory solution.

### 5.5 Intuitive interpretation of the relationship between concepts

Adaptive methods allow much more intuitive understanding of how the process of learning FCM weights takes place. A change in the activation of the cause concept occurs concurrently with a change of the activation value of the effect concept. These changes are reflected in the value of weight of the edge between two concepts. In our opinion, the ease of such interpretation is highly desirable in an FCM-based decision support system. Note that the FCM learned using adaptive DHL method is easier for a human expert to interpret. Even when the automatic reasoning process based on FCM (e.g., using equation (2)) is not performed, the learned FCM can be helpful for the expert as the illustration of the possible changes of concept activations within the concept space.

### 6 Experimental comparison of adaptive and evolutionary FCMs

Due to the specific inductive features of FCMs, we were interested in knowing whether the predictive capabilities of the obtained FCM-based models depend on the type of raw data used to learn the model. For the experimental evaluation of adaptive and evolutionary learning algorithms, we decided to use three classes of learning data, according to the dynamic features of the FCMs, i.e., 1) fixed point attractor, 2) limit cycle, and 3) chaotic attractor. The exemplary raw data will always be shown in the chart before presenting the results of the experiment.

In many practical applications, we have raw data available in the form of observations stored in a relational database. The construction of complex concepts on the basis of such data is not trivial and usually requires sophisticated
machine-learning techniques, e.g., learning of fuzzy membership functions for all concepts. The substantial support of a domain expert may also be necessary. Therefore, for the purposes of this chapter, we decided to simplify the construction of concepts, as is often done by researchers. We assumed that the concept labels correspond to the attributes of the data table. Instead of the complex construction of fuzzy concepts, we simply normalized the value sets of the attributes to the $[0, 1]$ range. The minima and maxima of value sets are computed as a data processing step.

6.1 Experimental settings

For all experiments described in this section we assume the following settings:

- the base time is $\Delta t = 1$;
- $\text{card}(T_H) = \text{card}(T_F) = t_{\text{pred}} = 10$;
- the fitness function (11) is used in all our experiments, the parameter $p = 1$;
- we use the equation (2) for reasoning.

We define the prediction error for the $i^{th}$ concept at time step $t$ according to equation (15):

$$
\text{err}(t) = |a_i(t) - a'_i(t)|,
$$

where $a_i(t)$ denotes the activation of concept observed from real data, and $a'_i(t)$ is the predicted activation of concept using the FCM-based model. The cumulative prediction error for $n$ time steps is defined as in equation (16).

$$
\text{err}(n) = \sum_{t=1}^{n} \text{err}(t)
$$

In the following experiments we test the ability of FCMs to perform two particular prediction tasks:

1. reproducing the original learning data assuming: $A(t_s) = A(t_0)$
2. predicting the unknown future data assuming: $A(t_s) \neq A(t_0)$

6.2 Quality of data reproduction

In this section we would like to test experimentally the ability of the FCMs to mimic the historical data used for learning. In Fig. 6.2 we show the raw data used to learn FCM in the first experiment. The matrix $W$ learned using 4 different algorithms is shown in table 6.2. Due to the assumed reasoning method 2, the values of weights on the diagonal of $W$ are equal to 0.

For the fixed-point attractor data, in the case of large fluctuations in the concept activations, the adaptive algorithms are not able to reflect changes within the full learning sequence. In Figure 6.2a, we can see that the DHL algorithm cannot learn weights properly, and the FCM cannot duplicate historical data well. The DHL algorithm assumes that, in any given time step, only two concepts
Fig. 5. Learning data of the ‘fixed-point attractor’ type

Table 1. The weights of the learned FCMs for ‘fixed-point attractor’ data
Fig. 6. The results of reproduction for ‘fixed-point attractor’ data

Fig. 7. Reproduction errors for ‘fixed-point attractor’ data
influence each other. Therefore, it is not possible to properly identify relationships between all concepts. In the subsequent time moments, the multiplication of two concept changes with values lower than 1.0 causes the weight values to gradually approach zero. At the same time, during the prediction, concept values that are close to zero scaled with the sigmoidal function are set to the value 0.5. Therefore after a few iterations, almost all concepts reach steady state.

The BDA adaptive algorithm deals with learning FCMs much better than DHL. Due to the use of more concepts in each time step, the values of the weights are not close to zero. In Figure 6b it can be seen that, for fixed data, the value of concepts $c_1$ is predicted quite well and that the prediction of the activation value of concept $c_2$ contains a relatively small error. Unfortunately, the other two concepts reach an erroneous steady state.

Another proposed algorithm, real coded genetic algorithm RCGA, is categorized as an evolutionary method, and it deals quite well with the prediction problem. Only slight errors were seen in earlier iterations. In subsequent time steps, when the data reach steady state, the error rate becomes very small (lower than 0.01).

The differential evolution algorithm is the second evolutionary method that is able to give very good results in the process of FCM learning. The prediction errors are comparable to those achieved using the RCGA algorithm. Initially, the prediction error was about 0.05, and, in the next iterations, it was reduced to a value of 0.01.

\begin{tabular}{|c|c|c|c|c|}
\hline
    & $A_1$ & $A_2$ & $A_3$ & $A_4$ \\
\hline
$t=1$ & 0.48 & 0.2 & 0.15 & 0.1 \\
$t=2$ & 0.44 & 0.3 & 0.3 & 0.2 \\
$t=3$ & 0.4 & 0.2 & 0.15 & 0.1 \\
$t=4$ & 0.44 & 0.3 & 0.3 & 0.2 \\
$t=5$ & 0.48 & 0.2 & 0.15 & 0.1 \\
$t=6$ & 0.44 & 0.3 & 0.3 & 0.2 \\
$t=7$ & 0.4 & 0.2 & 0.15 & 0.1 \\
$t=8$ & 0.44 & 0.3 & 0.3 & 0.2 \\
$t=9$ & 0.48 & 0.2 & 0.15 & 0.1 \\
$t=10$ & 0.44 & 0.3 & 0.3 & 0.2 \\
\hline
\end{tabular}

Fig. 8. Learning data of the "limit cycle" type

A similar situation occurs in the case of using cycle data for learning. The DHL and BDA algorithms cannot mimic cycle data. The application of BDA leads to a situation in which concept activations reach steady state in subsequent time steps. The application of the RCGA algorithm effectively facilitates learning the FCM. The sum of the prediction errors is quite small.

Chaotic data cannot be learned by DHL and BDA either, during the reasoning phase every concept activation reaches steady state.
Table 2. The weights of the learned FCMs for "limit cycle" data

<table>
<thead>
<tr>
<th></th>
<th>DHL</th>
<th>BDA</th>
<th>RCGA</th>
<th>DE</th>
</tr>
</thead>
<tbody>
<tr>
<td>W12</td>
<td>-0.000019</td>
<td>-0.07289</td>
<td>0.318</td>
<td>-0.78921</td>
</tr>
<tr>
<td>W13</td>
<td>-0.000032</td>
<td>-0.07289</td>
<td>-0.948</td>
<td>0.069206</td>
</tr>
<tr>
<td>W14</td>
<td>-0.001376</td>
<td>-0.27613</td>
<td>0.624</td>
<td>0.995486</td>
</tr>
<tr>
<td>W21</td>
<td>-0.000019</td>
<td>-0.02479</td>
<td>-0.12402</td>
<td>-0.11887</td>
</tr>
<tr>
<td>W23</td>
<td>0.005158</td>
<td>0.10609</td>
<td>0.172278</td>
<td>0.159555</td>
</tr>
<tr>
<td>W24</td>
<td>0.000052</td>
<td>0.024793</td>
<td>-0.79234</td>
<td>-0.94093</td>
</tr>
<tr>
<td>W31</td>
<td>-0.000032</td>
<td>-0.01653</td>
<td>0.172278</td>
<td>0.159555</td>
</tr>
<tr>
<td>W32</td>
<td>0.005158</td>
<td>0.070726</td>
<td>-0.75146</td>
<td>-0.74325</td>
</tr>
<tr>
<td>W34</td>
<td>0.000077</td>
<td>0.016528</td>
<td>-0.95552</td>
<td>-0.97217</td>
</tr>
<tr>
<td>W41</td>
<td>-0.001376</td>
<td>-0.23481</td>
<td>-0.06398</td>
<td>-0.13686</td>
</tr>
<tr>
<td>W42</td>
<td>0.000052</td>
<td>-0.10393</td>
<td>-0.85167</td>
<td>-0.4455</td>
</tr>
<tr>
<td>W43</td>
<td>0.000077</td>
<td>-0.10393</td>
<td>-0.51697</td>
<td>-0.81763</td>
</tr>
</tbody>
</table>

Fig. 9. Reproduction results for 'limit cycle' data
Fig. 10. Reproduction errors for 'limit cycle' data

<table>
<thead>
<tr>
<th>t</th>
<th>$A_1$</th>
<th>$A_2$</th>
<th>$A_3$</th>
<th>$A_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>t=1</td>
<td>0.57609</td>
<td>0.65286</td>
<td>0.98</td>
<td>1</td>
</tr>
<tr>
<td>t=2</td>
<td>0.5563</td>
<td>0.49524</td>
<td>0.5</td>
<td>0.78788</td>
</tr>
<tr>
<td>t=3</td>
<td>0.45</td>
<td>0.43</td>
<td>0.59</td>
<td>0.27273</td>
</tr>
<tr>
<td>t=4</td>
<td>0.65434</td>
<td>0.75056</td>
<td>0.69</td>
<td>0.60607</td>
</tr>
<tr>
<td>t=5</td>
<td>0.6413</td>
<td>0.69</td>
<td>0.66</td>
<td>0.25212</td>
</tr>
<tr>
<td>t=6</td>
<td>0.80435</td>
<td>0.83928</td>
<td>0.75</td>
<td>0.33333</td>
</tr>
<tr>
<td>t=7</td>
<td>0.76304</td>
<td>0.81</td>
<td>0.71</td>
<td>0.27273</td>
</tr>
<tr>
<td>t=8</td>
<td>0.756</td>
<td>0.9</td>
<td>0.81</td>
<td>0.21212</td>
</tr>
<tr>
<td>t=9</td>
<td>0.78319</td>
<td>0.88381</td>
<td>0.75</td>
<td>0.24697</td>
</tr>
<tr>
<td>t=10</td>
<td>0.7963</td>
<td>0.9112</td>
<td>0.77</td>
<td>0.28234</td>
</tr>
</tbody>
</table>

Fig. 11. Learning data of the 'chaotic attractor' type
Table 3. The weights of the learned FCMs for "chaotic attractor" data

<table>
<thead>
<tr>
<th></th>
<th>DHL</th>
<th>BDA</th>
<th>RCGA</th>
<th>DE</th>
</tr>
</thead>
<tbody>
<tr>
<td>W12</td>
<td>0.005068</td>
<td>0.091599</td>
<td>0.795016</td>
<td>0.147448</td>
</tr>
<tr>
<td>W13</td>
<td>0.001729</td>
<td>0.05207</td>
<td>-0.60798</td>
<td>0.32865</td>
</tr>
<tr>
<td>W14</td>
<td>0.007769</td>
<td>0.159386</td>
<td>0.122014</td>
<td>-0.32865</td>
</tr>
<tr>
<td>W21</td>
<td>0.005068</td>
<td>0.044876</td>
<td>0.777665</td>
<td>0.382542</td>
</tr>
<tr>
<td>W23</td>
<td>0.006718</td>
<td>0.036777</td>
<td>-0.12653</td>
<td>0.341903</td>
</tr>
<tr>
<td>W24</td>
<td>0.010871</td>
<td>0.044876</td>
<td>-0.50406</td>
<td>-0.56919</td>
</tr>
<tr>
<td>W31</td>
<td>0.001729</td>
<td>-0.00967</td>
<td>0.221459</td>
<td>0.814872</td>
</tr>
<tr>
<td>W32</td>
<td>0.013431</td>
<td>0.023119</td>
<td>0.093826</td>
<td>-0.3876</td>
</tr>
<tr>
<td>W34</td>
<td>0.006718</td>
<td>-0.00967</td>
<td>-0.13379</td>
<td>-0.22659</td>
</tr>
<tr>
<td>W41</td>
<td>0.007769</td>
<td>0.04869</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>W42</td>
<td>0.010871</td>
<td>0.027376</td>
<td>0.153213</td>
<td>-0.21901</td>
</tr>
<tr>
<td>W43</td>
<td>0.005799</td>
<td>0.019217</td>
<td>0.58101</td>
<td>0.999605</td>
</tr>
</tbody>
</table>

Fig. 12. Reproduction results for 'chaotic attractor' data.
It may be seen in Fig. 12 that, for chaotic data, the application of the RCGA algorithm during learning also leads to reaching the equilibrium state during reasoning. In this case, it may be assumed that concept values can be duplicated with some reasonable approximation, but, at the end of the reasoning process, the generated data have the tendency to reach steady state. Chaotic data were also difficult to duplicate by an FCM learned by the DE algorithm, but the prediction errors were much less than those for the RCGA algorithm. Note that the scales on the vertical axes of figures 13c and 13d differ.

In any case, the evolutionary methods of learning FCMs seem to be much better than adaptive. Therefore, we decided to present the cumulative error, defined in equation (16), only for the evolutionary learning method. The cumulative error $err(n)$ is summed for all three types of data, where the number of time steps $n = 10$. Every test understood as learning-and-testing cycle were run 10 times. The first column in Table 4 denotes the number of the particular test.

Evolutionary algorithms allow the learning of FCMs of quite good quality. Even for cases in which there are large variations in the concept activation values, problems are not created for the RCGA or the differential evolution algorithms.

6.3 Quality of data prediction

In this section we would like to test the behavior of the achieved FCMs assuming that the initial state vector $A(t_s)$ is changed i.e. $A(t_s) \neq A(t_0)$. For the following
Table 4. Cumulative reproduction error for cycle data

<table>
<thead>
<tr>
<th>Test</th>
<th>Chaotic DE</th>
<th>RCGA</th>
<th>Cycle DE</th>
<th>RCGA</th>
<th>Fixed DE</th>
<th>RCGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.595297</td>
<td>5.823462</td>
<td>0.173247</td>
<td>0.173247</td>
<td>0.279861</td>
<td>1.682506</td>
</tr>
<tr>
<td>2</td>
<td>1.31491</td>
<td>1.924132</td>
<td>0.17417</td>
<td>0.17417</td>
<td>0.279845</td>
<td>4.513751</td>
</tr>
<tr>
<td>3</td>
<td>2.110496</td>
<td>1.59704</td>
<td>0.1766</td>
<td>0.1766</td>
<td>0.279843</td>
<td>0.45272</td>
</tr>
<tr>
<td>4</td>
<td>1.513919</td>
<td>2.126377</td>
<td>0.166013</td>
<td>0.166013</td>
<td>0.27989</td>
<td>0.45272</td>
</tr>
<tr>
<td>5</td>
<td>2.315513</td>
<td>2.918895</td>
<td>0.17155</td>
<td>0.17155</td>
<td>0.279894</td>
<td>2.394499</td>
</tr>
<tr>
<td>6</td>
<td>2.213198</td>
<td>1.878982</td>
<td>0.168795</td>
<td>0.168795</td>
<td>0.279847</td>
<td>3.684865</td>
</tr>
<tr>
<td>7</td>
<td>1.6112</td>
<td>2.942499</td>
<td>0.17147</td>
<td>0.17147</td>
<td>0.279856</td>
<td>1.019418</td>
</tr>
<tr>
<td>8</td>
<td>1.455</td>
<td>4.888239</td>
<td>0.165302</td>
<td>0.165302</td>
<td>0.279871</td>
<td>2.620143</td>
</tr>
<tr>
<td>9</td>
<td>1.617157</td>
<td>2.210706</td>
<td>0.167254</td>
<td>0.167254</td>
<td>0.279872</td>
<td>2.227744</td>
</tr>
<tr>
<td>10</td>
<td>1.345905</td>
<td>3.923134</td>
<td>0.1708</td>
<td>0.1708</td>
<td>0.279887</td>
<td>4.89989</td>
</tr>
</tbody>
</table>

experiments we assume: \( a_1(t_s) = 0.3 \), \( a_2(t_s) = 0 \), \( a_3(t_s) = 0.1 \), \( a_4(t_s) = 0.3 \). We decided to compare the predicted sequence of state vectors with respect to the original learning data. The aim is to check whether the original type of data (fixed point attractor, cycles or chaotic behavior) are preserved. You can see in Fig. 14 that also this time, there were problems with the prediction of state vector by the adaptive FCMs. In case of using the BDA algorithm the stability of the state vector was not properly achieved. The activation value of concept \( c_3 \) differs significantly from its original stability value.

In case of evolutionary method you can notice that the large initial error (due to the forced initial state) (Fig. 15) is quickly reduced. The values of all concepts in stability state are also properly achieved.

Unfortunately, in case of cyclic data the problems with adaptive algorithms were similar. The state vector generated by the obtained FCM, reaches the stable state (Fig. 16) that is a negative property in this case. The results generated by evolutionary algorithms seems to be unexpected. The RCGA learning algorithm was in this case much better than the DE method. As you can see in Fig. 17 the data generated by DE cannot be identified as cyclic.

The results for chaotic data are quite interesting. Independant on the applied learning algorithm the generated state vector reaches the stable state Fig. 19. In case of using chaotic data it is not possible to learn properly the FCM.

### 6.4 Scaling up the evolutionary learning methods

On the basis of our previous experiments we have decided to choose the DE algorithm as the best for learning small FCMs consisting of only 4 concepts. It was interesting for us to check experimentally the possibility to scale up the algorithms for larger FCMs. In Fig. 20 you can see the chart of raw data for 8 concepts. The dimension of the optimization problem (the length of chromosome) is equal to 56. The obtained FCM is of quite good quality. The reconstructed data and reconstruction errors are shown in figures 21 and 22 respectively.
Fig. 14. The results of prediction for ‘fixed-point attractor’ data

Fig. 15. Prediction errors for ‘fixed-point attractor’ data
Fig. 16. The results of prediction for 'limit cycle' data

Fig. 17. Prediction errors for 'limit cycle' data
Fig. 18. The results of prediction for 'chaotic attractor' data

Fig. 19. Prediction errors for 'chaotic attractor' data
Fig. 20. Learning data of the 'fixed-point attractor' type - 8 concepts

Fig. 21. Prediction results for 'fixed-point attractor' data - 8 concepts
7 Conclusions and recommendations

In this chapter, we conducted a comparative analysis of adaptive and evolutionary learning methods of FCMs. In our opinion, the main advantage of evolutionary methods is the relative high quality of the obtained FCM with respect to the considered prediction task. This seems to be valid for any considered types of raw data used for learning. Using the standard forward reasoning method for FCM, the predictive capabilities of the adaptive FCMs are completely unsatisfactory. In our opinion, the cumulative and linear features of the standard reasoning process seem to be inappropriate for use with adaptive learning methods. On the other hand, the intuitive interpretation of adaptive FCMs by humans seems to be better. In conclusion, we definitely recommend evolutionary methods for learning FCMs that are intended for use in making predictions. In some cases in which the quality of prediction is not crucial, the existing adaptive methods of learning FCMs can be also useful.

References