COMPETITIVE EXCEPTION LEARNING USING FUZZY FREQUENCY DISTRIBUTIONS

W-M. van den Bergh, J. van den Berg

ERIM REPORT SERIES *RESEARCH IN MANAGEMENT*	
ERIM Report Series reference number	ERS-2000-06-LIS
Publication status / version	draft / version February 2000
Number of pages	12
Email address first author	Vandenbergh@few.eur.nl
Address	Erasmus Research Institute of Management (ERIM) Rotterdam School of Management / Faculteit Bedrijfskunde Erasmus Universiteit Rotterdam PoBox 1738 3000 DR Rotterdam, The Netherlands Phone: # 31-(0) 10-408 1182 Fax: # 31-(0) 10-408 9020 Email: info@erim.eur.nl Internet: www.erim.eur.nl

Bibliographic data and classifications of all the ERIM reports are also available on the ERIM website:
www.erim.eur.nl

ERASMUS **R**ESEARCH **I**NSTITUTE OF **M**ANAGEMENT

REPORT SERIES
RESEARCH IN MANAGEMENT

BIBLIOGRAPHIC DATA AND CLASSIFICATIONS		
Abstract	A competitive exception learning algorithm for finding a non-linear mapping is proposed which puts the emphasis on the discovery of the important exceptions rather than the main rules. To do so,we first cluster the output space using a competitive fuzzy clustering algorithm and derive a fuzzy frequency distribution describing the general, average system's output behavior. Next, we look for a fuzzy partitioning of the input space in such away that the corresponding fuzzy output frequency distributions `deviate at most' from the average one as found in the first step. In this way, the most important `exceptional regions' in the input-output relation are determined. Using the joint input-output fuzzy frequency distributions, the complete input-output function as extracted from the data, can be expressed mathematically. In addition, the exceptions encountered can be collected and described as a set of fuzzy if-then-else-rules. Besides presenting a theoretical description of the new exception learning algorithm, we report on the outcomes of certain practical simulations.	
Library of Congress Classification (LCC)	5001-6182	Business
	5201-5982	Business Science
	5691-5716	Business Mathematics
Journal of Economic Literature (JEL)	M	Business Administration and Business Economics
	M 11 R 4	Production Management Transportation Systems
	C6	Mathematical Methods and Programming
European Business Schools Library Group (EBSLG)	85 A	Business General
	260 K 240 B	Logistics Information Systems Management
	250 B	Fuzzy theory
Gemeenschappelijke Onderwerpsontsluiting (GOO)		
Classification GOO	85.00	Bedrijfskunde, Organisatiekunde: algemeen
	85.34 85.20	Logistiek management Bestuurlijke informatie, informatieverzorging
	31.80	Toepassingen van de Wiskunde
Keywords GOO	Bedrijfskunde / Bedrijfseconomie	
	Bedrijfsprocessen, logistiek, management informatiesystemen	
	Algoritmen, Fuzzy theorie, Leerprocessen	
Free keywords	exception learning, fuzzy pattern recognition, competitive learning	
Other information		

COMPETITIVE EXCEPTION LEARNING USING FUZZY FREQUENCY DISTRIBUTIONS

W-M. van den Bergh, J. van den Berg *
Faculty of Economics, Erasmus University of Rotterdam
Email : vandenbergh@few.eur.nl, jvandenberg@few.eur.nl

Abstract: A competitive exception learning algorithm for finding a non-linear mapping is proposed which puts the emphasis on the discovery of the important exceptions rather than the main rules. To do so, we first cluster the output space using a competitive fuzzy clustering algorithm and derive a fuzzy frequency distribution describing the general, average system's output behavior. Next, we look for a fuzzy partitioning of the input space in such a way that the corresponding fuzzy output frequency distributions 'deviate at most' from the average one as found in the first step. In this way, the most important 'exceptional regions' in the input-output relation are determined. Using the joint input-output fuzzy frequency distributions, the complete input-output function as extracted from the data, can be expressed mathematically. In addition, the exceptions encountered can be collected and described as a set of fuzzy if-then-else-rules. Besides presenting a theoretical description of the new exception learning algorithm, we report on the outcomes of certain practical simulations.

1. Introduction

The quotes (1) "The ability to handle exceptions and uncertain rules is extremely important, but is orthogonal to the task of understanding the general ontology." [5], (2) "These monitors, however, generally cannot predict when the robot *will* get into trouble." [7], and (3) "This structure can be viewed either as a generalization of rules allowing exceptions (...)." [2], show that at least at certain places in the literature of Artificial Intelligence some attention is given to the concept of 'exceptions'. However, a search for more references to exceptions in this wide literature, yields the ironical result that paying attention to exceptions seems to be itself an exception because practically all emphasis is put on the discovery (learning, mining) of the main or general rules.

Our inspiration for developing the *competitive exception learning algorithm* (CELA) stems from the wish to analyze certain time series based on economical data. Here, one often discovers various 'regimes' of behavior: the volatility may

*We thank Maarten van Enschot for being our inspiring sparring partner during the kick-off of this research project.

change over time (slowly or rapidly), a certain trend may suddenly be interrupted, or the period of a cycle may strongly differ during various, possibly short, time periods. In more extensive terms, our motivation may expressed as follows: The efficient market hypothesis (EMH) states that all relevant information is instantaneously and fully assimilated in the asset prices. So, in a truly efficient (and frictionless) market, an agent cannot gain utility by entering a (buy or sell) transaction. However, if one admits the fact that "agents are constantly learning" [6] - i.e., if one admits a less rigid view of efficiency - the possibility to add utility (i.e., to employ profit opportunities) may exist *temporally* for fast learning agents. In other words, if an agent is better able to perceive the true risk involved, taking a position may yield him risk-adjusted returns. In addition, since profit opportunities often (seem to) appear randomly and infrequently, the agent should concentrate on detecting the unusual, abnormal states (which we call exceptions) rather than the average normal states[1]. Within this economic framework, an agent (or, in our case, a software method) capable of identifying exceptions has to interpret accurately the available information, that is the collection of time series signals. His (its) primary concern is to detect this kind of exploitable departures from a random walk, that is, to identify deterministic parts in relevant time series. His (its) task is to predict the future market state $y(t+1)$ given m historical states $y(t), y(t-1), y(t-2), \ldots, y(t-m+1)$ where the predicting power of the historical data changes over time. In most periods, the time series behaves like a random walk and historical data do not possess any forecasting value. But infrequently at certain sudden moments, the actual data contain some exceptional pattern having predicting information of interest: it is conditional for the subsequent behavior of the market.

We think that our algorithms can be useful in many other fields of pattern recognition (outside the area of time series), namely in all cases where it is required to pay especial attention to exceptions. Therefore, we formulated our algorithm in very broad terms. The general purpose underlying our new algorithm CELA, which, basically, is a composition of two *unsupervised* learning algorithms, can be stated as follows: Given a set S of, often noisy, training data, its goal is to find a (non-linear) mapping

$$f : X \to Y. \tag{1}$$

More particularly however, we are interested in finding *exceptions*, that is, regions in the input space X where the corresponding function values $y = f(x)$ *notably deviate* from the average ones. This is, of course, a quite vague criterion that should be formalized mathematically.

Before doing so, we think that some words on the methodology we used are appropriate. The research leading to this paper started using a mainly empirical approach. Only later on, when simulation results turned out to be quite successful, we started to look for a more mathematical description. We then decided to formulate our findings within the framework of machine learning: both fuzzy systems and competitive learning, with in the background the paradigm of genetic algorithms, appeared to be appropriate fields in this context. The term 'exception learning', we introduced ourselves.

[1] In this sense, the goal of an economic agent deviates from the goal of a descriptive statistician.

The structure of this paper is follows. In the next section, we introduce our CELA using a mathematical description and illuminating the various (sub)steps using a chaotic time series example where quite a lot of noise is added to the data. Altogether this explanation takes 3 subsections. In the final section, we take a look at what we have done so far, evaluate the results and propose some future directions of research.

2. Competitive exception learning

We now present in detail the new competitive exception learning algorithm. As mentioned above, we seek to find a mapping (1) from an (M-dimensional) input space X into an (N-dimensional) output space Y, given a representative data set S. The set S contains P data pairs $(x_p; y_p) = (x_{p,1}, x_{p,2} \ldots x_{p,M}; y_{p,1}, y_{p,2} \ldots y_{p,N})$, where $p = 1, \ldots, P$. The CELA consists of three steps:

2.1 Finding the unconditional output cluster membership distribution

Irrespective of the x_p-values of the data points of S, we (unconditionally) cluster the given y_p-values using the following fuzzy competitive learning approach:

- Consider the case of using C_y cluster centroids $\bar{y}_c = (y_{c,1}, y_{c,2}, \ldots, y_{c,N})$, $(c = 1, \ldots, C_y)$ somewhere in the output space Y. We define the degree $\mu^Y_{p,c}$ to which data point y_p belongs to cluster centroid $\bar{y}_c$ as

$$\mu^Y_{p,c} = \frac{d^{-q}_{p,c}}{\sum_{k=1}^{C_y} d^{-q}_{p,k}}, \tag{2}$$

  where the power q usually equals 2 and where $d_{p,c}$ represents a certain natural distance measure between the points y_p and $\bar{y}_c$ (e.g., $d_{p,c} = \|y_p - \bar{y}_c\|$, with $\|\cdot\|$ representing the Euclidean norm). The above-given definition implies that, irrespective the concrete cluster centroid locations $\bar{y}_c$, we have

$$\forall y_p : \sum_{k=1}^{C_y} \mu^Y_{p,k} = 1. \tag{3}$$

  Or, in words, every data point y_p belongs to all clusters $\bar{y}_c$ in some degree as defined by the *fuzzy membership function* $\mu^Y_{p,c}$. These membership values are normalized: for any data point y_p, they sum up till precisely 1, independent of the actual positions $\bar{y}_c$ of the cluster centroids.

  For any data point y_p, the index c^* is used for the index of the *winning* cluster, i.e., the cluster with the highest membership value with respect to y_p. So c^* has the property that - given p -

$$\forall c : \mu^Y_{p,c^*} \geq \mu^Y_{p,c}. \tag{4}$$

- We next introduce our first competitive learning algorithm in order to discover an 'optimal' partitioning of the output space. In order to measure the quality

of a partition found, we use the so-called *partition fitness* function $PF()$ being the average of the winner membership values μ^Y_{p,c^*}:

$$PF(\bar{y}_1, \bar{y}_2, \ldots, \bar{y}_{C_y}) = \frac{1}{P} \sum_{p=1}^{P} \mu^Y_{p,c^*}. \tag{5}$$

Note that

$$\forall p : 0 \leq \mu^Y_{p,c^*} \leq 1 \implies 0 \leq PF() \leq 1. \tag{6}$$

By changing the cluster centroid positions $\bar{y}_1, \bar{y}_2, \ldots, \bar{y}_{C_y}$, we try to maximize $PF()$. The motivation for this is as follows: If every data point y_p have been able to 'attract' (at least) one cluster centroid $\bar{y}_c$ such that $y_p = \bar{y}_c$, the fitness function will have its maximum value, i.e., $PF() = 1$. In the more general case that the number of cluster centroids is smaller than the number of data points, the highest value of $PF()$ is less than 1 and corresponds to a constellation where each cluster centroid is situated in the center of one of the 'data clouds'.

Instead of applying some gradient-based competitive learning [3] strategy, we experimented several other search heuristics. The simplest one is a *random search* heuristic: applying this approach, we continuously generate at random a set of C_y cluster centroid positions and calculate (5). If this value (PF^{curr}) is lower than the previous best one found, we replace this previous best one by PF^{curr} and generate a new set of randomly chosen cluster centroid positions. Otherwise, we directly generate such a new set[2].

- The result of the above-given sub-step is a fuzzy clustering where the output space Y has been fuzzily partitioned [1]: the fuzzy centroid positions should be met near the centers of the various data crowds. This partitioning can also be used to construct a *fuzzy frequency distribution* of y. To do so, the fuzzy subsets as resulting from the partitioning are chosen as the (fuzzy) classes. To find the fuzzy frequency distribution, the membership values of all data points y_p with respect to all fuzzy classes are summed up[3] and, for normalization reasons, divided by P. This yields

$$\bar{\mu}^Y = (\bar{\mu}^Y_1, \bar{\mu}^Y_2, \ldots, \bar{\mu}^Y_{C_y}), \quad \text{where} \quad \bar{\mu}^Y_c = \frac{1}{P} \sum_{p=1}^{P} \mu^Y_{p,c}. \tag{7}$$

So, the vector $\bar{\mu}^Y = (\bar{\mu}^Y_1, \bar{\mu}^Y_2, \ldots, \bar{\mu}^Y_{C_y})$ describes the distribution of the expected (average) membership values $\bar{\mu}^Y_c$ $(c = 1, 2, \ldots, C_y)$. Since our analysis so far only takes y-values into account and ignores the corresponding x-values, this frequency distribution is called the *unconditional output cluster membership distribution* (UOD).

Note that based on the membership and the cluster centroid values of the UOD, we can easily calculate an (unconditional) output estimate $\hat{y}$ of the

[2] In addition, a local search step can be applied in order to improve this heuristic.

[3] Note the difference of this approach to the classical *crisp* one where any data point belongs to precisely one class.

overall output mean $\bar{y}$ conform

$$\hat{y} = \sum_{c=1}^{C_y} \bar{y}_c \times \bar{\mu}_c^Y. \tag{8}$$

In addition, we can also calculate an unconditional output estimate $\hat{y}_p$ for any data point y_p conform

$$\hat{y}_p = \sum_{c=1}^{C_y} \bar{y}_c \times \mu_{p,c}^Y. \tag{9}$$

An illustrative example

A simple example may further clarify the first step. We will investigate a chaotic one-dimensional time series known as a *logistic map*. Above the deterministic (but seemingly random) process of the logistic map, we add uniformly distributed noise e_p, drawn randomly from U(0;1). The logistic map has the form

$$\acute{y_p} = 1 - 4y_{p-1} + 4y_{p-1}^2 \tag{10}$$

and the resulting series looks like

$$y_p = 0.5\acute{y_p} + 0.5e_p. \tag{11}$$

Note that the deterministic part and the randomly drawn part each weight for 50 percent. Since the average value of the noise is .5, the unconditional expectation of the (p)-th element in this series is

$$\bar{y}_p = 0.75 - 2y_{p-1} + 2y_{p-1}^2. \tag{12}$$

In figure 1 we plot the time series for 100 subsequent values resulting from this process and in figure 2 the scatter plot from state $p-1$ to state p is shown. In this figure we also inserted the deterministic part of the map following (12). To conclude we added the linear regression line between y_p and y_{p-1}. The downward slope

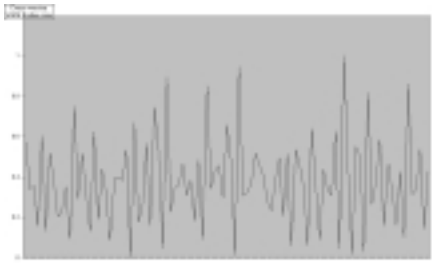

Fig. 1. Noisy chaotic time series used.

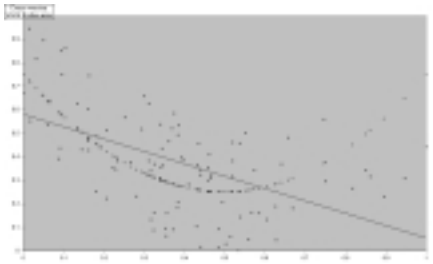

Fig. 2. Scatter plot and 2 fitting curves.

of this regression is merely coincidence: the randomly drawn noise added to the (infrequent) higher values of the series proves to be lower than average (no central limit helping us here), thus introducing some downward bias. Preferably a pattern

recognition system should not be led away by this bias. The only deterministic (i.e. predictable) part is the pattern described by (12) and that is precisely the structure we are trying to infer from the noisy data given. We applied the above-given step 1 of the algorithm to the noisy logistic series using 2, 3, and 4 Y-clusters respectively. The UOD conform equation (7) and the corresponding clusters centroids $\bar{y}_c$ for each are shown in the following tables:

$C_y = 2$	$c = 1$	$c = 2$
$\bar{\mu}_c^Y$	0.8728	0.1272
$\bar{y}_c$	0.4047	1.0000

$C_y = 3$	$c = 1$	$c = 2$	$c = 3$
$\bar{\mu}_c^Y$	0.0962	0.5938	0.3100
$\bar{y}_c$	0.9959	0.4077	0.1023

$C_y = 4$	$c = 1$	$c = 2$	$c = 3$	$c = 4$
$\bar{\mu}_c^Y$	0.2686	0.3804	0.2668	0.0842
$\bar{y}_c$	0.1339	0.3546	0.5418	0.8758

Based on these values and using (2), we can determine the cluster membership value for all points y. The next figures at the left-hand side show the resulting cluster membership distribution for $C_y = 2$, $C_y = 3$, and $C_y = 4$ respectively.

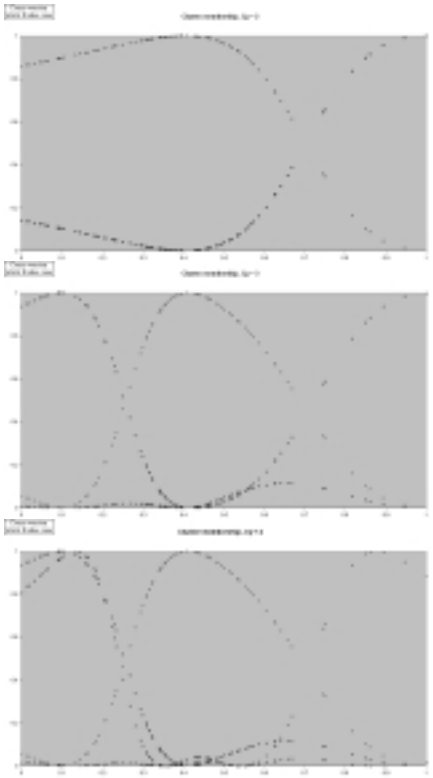

Fig. 3. Cluster membership distributions.

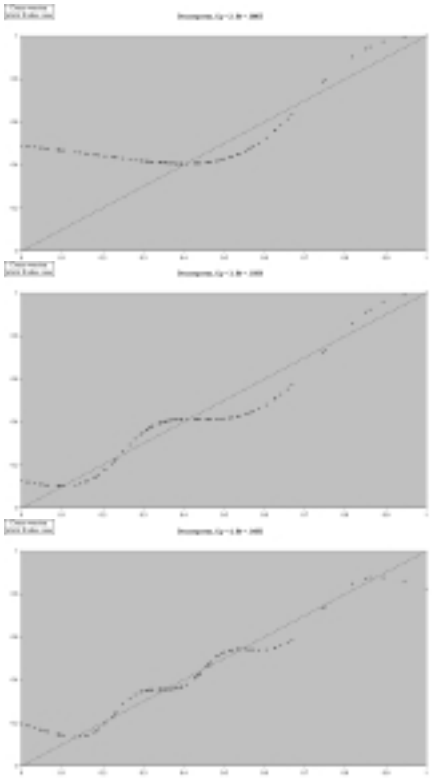

Fig. 4. Reconstruction of the Y-space.

The case $C_y = 4$ deserves some special attention. At first sight the number of clusters seems redundant, especially the first two clusters. However, a more careful

examination reveals that the regions which are more difficult to classify - i.e., around $y_p = 0.25$ and $y_p = 0.75$ - are better captured using the extra degrees of freedom. Following (8), we can also calculate the unconditional estimate $\hat{y}$. For $C_y = 2$, $C_y = 3$ and $C_y = 4$ respectively, we get 0.4804, 0.3696 and 0.3892. These values approximate the mean y-value which appears to be $\bar{y} = 0.3806$. Note that the more cluster are used, the approximation is better[4].

Another interesting result of step 1 is that by application of (9), we can reconstruct (actually 'decompress') the original Y-space. In the figures 4, we have plotted the resulting y-values against the original ones. A diagonal line has been included as a reference. As fitness criterion we used the mean squared error criterion

$$\text{fit } = 1 - \sqrt{\frac{1}{P}\sum_{p=1}^{P}(\hat{y}_p - y_p)^2}, \tag{13}$$

which appears to be .8065, .9359, and .9456 respectively. An again, the fit becomes better when more clusters are used.

2.2 Finding the conditional output cluster membership distribution

In the second step of the CELA, we consider the complete data set of pairs (x_p, y_p) and suppose that some (non-)linear stochastic dependency of y on x exists. We want to identify fuzzy regions in the domain of x where the corresponding cluster membership values of y significantly deviate from the expected unconditional ones $\bar{\mu}_c^Y$ as found in step 1. In short, we want to identify values of x that relate to exceptional values of y, or, in still other words, we want to identify exceptional behavior as compared to expected behavior. There are 3 sub steps:

- We start calculating the 'output exception' OE for every output value y_p conform

$$OE(p) = \sqrt{\sum_{c=1}^{C_y}(\mu_{p,c}^Y - \bar{\mu}_c^Y)^2}. \tag{14}$$

  So, $OE(p)$ tells how much the cluster membership values of y_p deviate unconditionally from expected cluster membership values. These output exceptions can be normalized to so-termed Exception Signals (ES) defined as

$$ES(p) = \frac{OE(p)}{\sum_{p=1}^{P} OE(p)} \tag{15}$$

- Now, we take the x-values into account supposing that they affect the dependent variable y. More particularly, we define fuzzy clusters in the input space and try to discover fuzzy input clusters (by again using a competitive learning strategy) that explain at best exceptional y-values. This is the trickiest part of the algorithm: For any set of cluster centroids $\bar{x}_b = (x_{b,1}, x_{b,2}, \ldots, x_{b,M})$, $(b = 1, \ldots, B_x$, B_x being a user-defined number) in the domain of X, the

[4] If we use P clusters the fit would be 1, i.e., an exact representation, but at the cost of loosing generality.

input membership values $\mu^X_{p,b}$ can be calculated using the approach as given by equation (2):

$$\mu^X_{p,b} = \frac{d^{-q}_{p,b}}{\sum_{k=1}^{B_x} d^{-q}_{p,k}}. \tag{16}$$

This time, $d_{p,b}$ represents the distance between x_p and $\bar{x}_b$. Next, we calculate the Exception Contributions (ECs) conform

$$EC(p,b) = \mu^X_{p,b}\, ES(p). \tag{17}$$

Roughly spoken, $EC(p,b)$ expresses how much the input cluster b contributes to the output exception $ES(p)$. For every x_p, we can also calculate the winning input cluster b^* indicating which of the input clusters contributes at most to the output exception[5] $ES(p)$. So b^* has the property that - given p -

$$\forall b : EC^X_{p,b^*} \geq EC^X_{p,b}. \tag{18}$$

Finally, the *exception fitness* $EF()$ defined as

$$EF(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_{B_x}) = \sum_{p=1}^{P} EC^X_{p,b^*}, \tag{19}$$

can be used as yield function for the second competitive learning algorithm. Applying a simple heuristic, we again use the random search approach from step 1 by (a) repeatedly generating new input cluster centroid positions and (b) calculating the corresponding fitness $EF()$. The set of input clusters resulting into the highest exception fitness is the one we are looking for.[6]

- Having determined both the input and the output cluster centroids, we can calculate the so-called *conditional output cluster membership distribution* (COD): For any input cluster b

$$\bar{\mu}^Y|\bar{\mu}^X_b = (\bar{\mu}^Y_1|\bar{\mu}^X_b, \bar{\mu}^Y_2|\bar{\mu}^X_b, \ldots, \bar{\mu}^Y_{C_y}|\bar{\mu}^X_b), \tag{20}$$

where

$$\bar{\mu}^Y_c|\bar{\mu}^X_b = \frac{\sum_p \mu^X_{p,b}\,\mu^Y_{p,c}}{\sum_p \mu^X_{p,b}\,\mu^Y_{p,1} + \ldots + \sum_p \mu^X_{p,b}\,\mu^Y_{p,C_y}} = \frac{\sum_p \mu^X_{p,b}\,\mu^Y_{p,c}}{\sum_p \mu^X_{p,b}}. \tag{21}$$

Thus, the COD is composed of a set of C_x fuzzy membership distributions $\bar{\mu}^Y|\bar{\mu}^X_b$ of y. Each of these distributions consists of a series of normalized frequency values expressing in which percentages the output training values on average belong to the various fuzzy output classes, *given* the average membership value of x in respect of input cluster b.

[5] Here we have the crux: we look for regions in the input space which correspond at most to exceptional output values.

[6] The nett effect of maximizing the exceptional fitness function $EF()$ can also be expressed as follows: *It is tried to position the cluster centroids* $\bar{x}_b$ near those points x_p which correspond to exceptional y_p-values or, more generally, *near those clouds of* x_p*-points which correspond to clouds of exceptional* y_p*-values.*

Continuation of the illustrative example

For our previous example we have estimated a mapping from 2, 3, 4, and 5 X-clusters respectively to 3 Y-clusters. The resulting conditional output cluster membership mapping is shown in the following table:

		$C_y = 3$	$c = 1$	$c = 2$	$c = 3$	
		$\bar{y}_c$	0.9959	0.4077	0.1023	
	$\bar{x}_b$					$\hat{y}_b$
$b = 1$	0.4112	$\bar{\mu}_c^Y \vert \bar{\mu}_1^X$	0.0971	0.3147	0.5881	0.2852
$b = 2$	1.0000	$\bar{\mu}_c^Y \vert \bar{\mu}_2^X$	0.0894	0.2774	0.6332	0.2669
$b = 1$	0.0979	$\bar{\mu}_c^Y \vert \bar{\mu}_1^X$	0.2561	0.6082	0.1357	0.5169
$b = 2$	0.4140	$\bar{\mu}_c^Y \vert \bar{\mu}_2^X$	0.0242	0.5763	0.3995	0.2999
$b = 3$	1.0000	$\bar{\mu}_c^Y \vert \bar{\mu}_3^X$	0.0214	0.6563	0.3224	0.3219
$b = 1$	0.9014	$\bar{\mu}_c^Y \vert \bar{\mu}_1^X$	0.0271	0.6886	0.2843	0.3368
$b = 2$	0.5017	$\bar{\mu}_c^Y \vert \bar{\mu}_2^X$	0.0181	0.5126	0.4693	0.2750
$b = 3$	0.1411	$\bar{\mu}_c^Y \vert \bar{\mu}_3^X$	0.2736	0.6229	0.1034	0.5370
$b = 4$	0.3404	$\bar{\mu}_c^Y \vert \bar{\mu}_4^X$	0.0467	0.6186	0.3347	0.3330
$b = 1$	0.1776	$\bar{\mu}_c^Y \vert \bar{\mu}_1^X$	0.1041	0.7460	0.1498	0.4231
$b = 2$	0.5015	$\bar{\mu}_c^Y \vert \bar{\mu}_2^X$	0.0119	0.5120	0.4761	0.2693
$b = 3$	0.3376	$\bar{\mu}_c^Y \vert \bar{\mu}_3^X$	0.0354	0.6149	0.3497	0.3217
$b = 4$	0.0605	$\bar{\mu}_c^Y \vert \bar{\mu}_4^X$	0.4546	0.4726	0.0728	0.6529
$b = 5$	0.8947	$\bar{\mu}_c^Y \vert \bar{\mu}_5^X$	0.0210	0.6904	0.2887	0.3319

The second row and the second column show the cluster centroid locations of the output y and the input x respectively. Most cells are used to show the conditional membership values $\bar{\mu}_c^Y \vert \bar{x}_b^X$. They sum up to 1 for each row. In the last column we calculated the estimated y-centroid value $\hat{y}_b$ for every input cluster b, according to (23). This will be further illuminated in the next step.

2.3 The final map and its interpretation

In this last step of the CELA, we derive a formula describing the structure underlying the given data set S. We also show how the mapping can be interpreted.

- Using the above-given analysis, we can formulate the mapping (1). For any input point x_p, even outside the training set S, the corresponding y_p can be estimated conform

$$\hat{y}_p = F(x_p) = \sum_{b=1}^{C_x} \mu_{p,b}^X \, (\sum_{c=1}^{C_y} \bar{y}_c \times \bar{\mu}_c^Y \vert \bar{\mu}_b^X). \tag{22}$$

  Here the $\bar{y}_c$-values come from the step 1, the $\bar{\mu}_c^Y \vert \bar{\mu}_b^X$ equal (21), and the $\mu_{p,b}^X$ equal (16). So the desired value of y is calculated from an (input membership) weighted sum of (conditional output membership) weighted sum of cluster centroid locations.

  The following observations may be illustrative to understand formula (22):

(1) If x_p coincides with cluster centroid $\bar{x}_b$, then (22) reduces to

$$\hat{y}_b = \sum_{c=1}^{C_y} \bar{y}_c \times \bar{\mu}_c^Y | \bar{\mu}_b^X, \tag{23}$$

i.e., to the sum of output cluster centroids $\bar{y}_c$, weighted by the 'local' membership values $\bar{\mu}_c^Y | \bar{\mu}_b^X$.
(2) If x_p does not coincide with one of the input cluster centroids, we simply extend the previous procedure by summing over the weighted sums of output cluster centroids also taking the input membership values $\mu_{p,b}^X$ ($b = 1, 2, \ldots, C_x$) of x_p into account.

- If desired, a *fuzzy rule base* describing the exceptions in linguistic terms, can also be derived. To illuminate: we know that around input centroids $\bar{x}_p$, the y-values are exceptional. Inspecting both the input centroids and the exceptional output values and providing them with 'linguistic values' [4], it is possible to derive fuzzy rules of type

  if $x_{-,1}$ is small, if $x_{-,2}$ is large,..., and if $x_{-,M}$ is medium,
  then $y_{-,1}$ is large, ..., and $y_{-,N}$ is medium.

  Note that the consequence of this fuzzy rule is multidimensional.

Continuation of the illustrative example

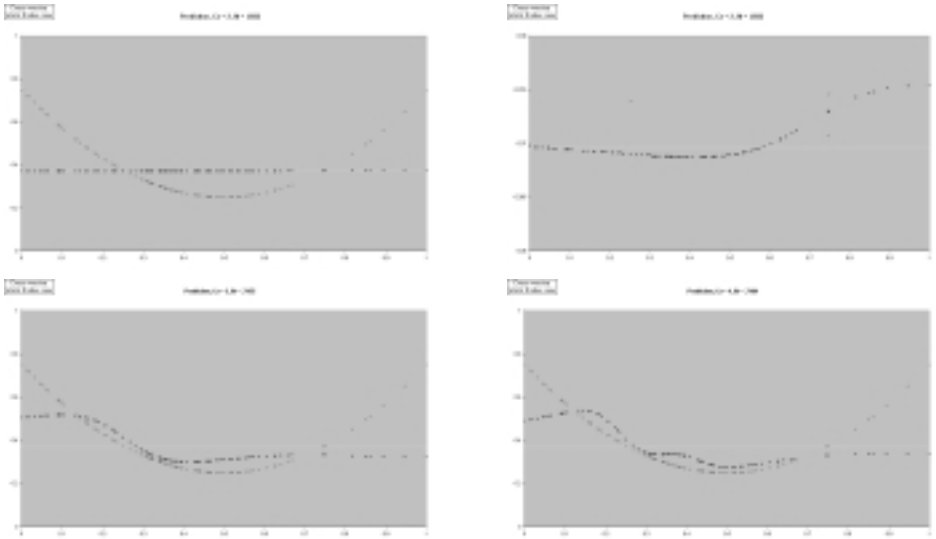

Fig. 5. Various estimates of the structure underlying the noisy data.

To get more insight in the way the CELA is capable of recognizing the underlying structure, we generated y-estimates $\hat{y}_p$ for all x-values conform (22) and scattered

them in the above-given plots where we also included the deterministic (parabolic) pattern (12). The two figures in the first row represent the estimates $\hat{y}_p$ in case $C_x = 2$, where the figure at the right-hand side is a magnification of the figure at the left-hand side. The magnification clearly shows that even with 2 input clusters the algorithm *primitively* recognizes the U- shape. The two figures in the second row represent the y-estimates in case $C_x = 3$, $C_x = 5$ respectively. The fit values (13) now compare the estimated $\hat{y}_P$-values conform (22) to the noiseless y-values conform (12) and equal, from left to right and from top to bottom, 0.6922, 0.6922, 0.7455, and 0.7697. As is to be expected, the more clusters, the better the fit. It becomes also clear that the performance for high input values is worse than for low and average values. The reason is, as noted earlier, the incidental bias in the noise for the relatively few high input values. But the CELA seems to be less vulnerable for this than the linear regression (see fig. 2).

The calculated $\hat{y}_p$-values as given in the table at the end of step 2 of the CELA, reveal the hidden *fuzzy rules* underlying the mapping. If we look, for instance, to the 3x3 mapping, we see that low x-values yield relatively high y- estimates, average x-values yield average y-estimates and high x-values yield higher than average y-estimates. Indeed, the basic properties of the U-shaped logistic map have been (roughly) identified.

3. Discussion and Future Research

This paper has introduced a new algorithm termed the Competitive Exception Learning Algorithm (CELA) which tries to discover the functional relationship between an input and an output space, given a large set of possibly very noisy data. Emphasis is put on the discovery of (clouds of) exceptions in this relationship. In order to reach this goal, the output space is fuzzily clustered: the corresponding fuzzy frequency distribution describes in a condensed way the structure of the output space. Next, the input space is partitioned in those fuzzy input clusters that correspond at best to output clusters with exceptional y-values. In the end, *a mapping is found based on both the input cluster centroids, the output cluster centroids and the corresponding (conditional) membership functions.* We think, this can be considered - like various other techniques in machine learning - as a way of *data compression.* In our approach, the underlying structure is found while putting much emphasis on the discovery of exceptions and this is precisely what makes it so interesting. The initial simulations performed show that the CELA is able to analyze various time series. In this paper, we presented the outcomes of a simulation concerning a very noisy chaotic time series. Still the algorithm approximated the underlying structure quite well. We also want to emphasize here that the CELA can deal with mapping problems having both a multi-dimensional input space *and* a multi-dimensional output space. This is big advantage over most traditional methods since they usually consider an one-dimensional output space only.

In our view, this initial research step is only the beginning. Much of the properties of the CELA, we still not understand. For example, simulation results suggest that the CELA is indeed capable of discovering the basis structures underlying a set of quite noisy training data. But how precisely does this relate to its sensi-

tiveness to exceptions when the number of cluster centroids is gradually increased? Does overfitting springs up here rapidly? If so, can we find the minimal numbers of cluster centroids (i.e., minimal model complexity) that is able to represent the real, deterministic structure? Can cross-validation be used here in order to validate the model complexity to use? We doubt it: one can argue that the number of clusters needed depend on the goal of the CELA-user: if (s)he is interested in finding several (small) exceptions rather than average behavior (compare the decision problem of the intelligent agent trying to employ profit opportunities in a financial market, see the Introduction), larger cluster numbers may be needed. In other words, if one is interested in the discovery of various (possibly detailed) patterns, the complexity of the model used should be increased appropriately (here, by increasing the number of clusters in the input or output space). This is of course generally a very hard problem to tackle since in practice, we often do not know much about the detailed properties of the underlying structure.

To finish, mathematicians might be interested in the universal approximation capabilities of the CELA while experimentalists might prefer to try new application domains. We ourselves shall continue our research in many of the directions suggested in the near future.

References

[1] R. Babuska. *Fuzzy Modeling and Identification*, PhD-thesis, Technical University Delft, 1996.

[2] B.R. Gaines, Transforming Rules and Trees, In: *Advances in Knowledge Discovery and Data Mining*, eds. U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, MIT Press, 1996.

[3] J. Hertz, A. Krogh, and R.G. Palmer. *Introduction to the Theory of Neural Computation.* Addison-Wesley, 1991.

[4] G.J. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic, Theory and Applications*, Prentice Hall, 1995.

[5] S. Russell and P. Norvig, *Artificial Intelligence, A Modern Approach*, Prentice Hall, 1995.

[6] J.A. Scheinkman and B. LeBaron, Nonlinear Dynamics and Stock Returns, *Journal of Business* 62, no. 3, pp. 311-337, 1989.

[7] R. Simmons, *Layering Learning on top of the Task Control Architecture*, http://www.cs.cmu.edu/people/xavier/talks/simmons_030895.txt.

ERASMUS **R**ESEARCH **I**NSTITUTE OF **M**ANAGEMENT

REPORT SERIES
RESEARCH IN MANAGEMENT

Other Publications in the Report Series Research* in Management

Impact of the Employee Communication and Perceived External Prestige on Organizational Identification
Ale Smidts, Cees B.M. van Riel & Ad Th.H. Pruyn
ERS-2000-01-MKT

Critical Complexities, from marginal paradigms to learning networks
Slawomir Magala
ERS-2000-02-ORG

Forecasting Market Shares from Models for Sales
Dennis Fok & Philip Hans Franses
ERS-2000-03-MKT

A Greedy Heuristic for a Three-Level Multi-Period Single-Sourcing Problem
H. Edwin Romeijn & Dolores Romero Morales
ERS-2000-04-LIS

Integer Constraints for Train Series Connections
Rob A. Zuidwijk & Leo G. Kroon
ERS-2000-05-LIS

---

* ERIM Research Programs:
LIS Business Processes, Logistics and Information Systems
ORG Managing Relationships for Performance
MKT Decision Making in Marketing Management
F&A Financial Decision Making and Accounting
STR Strategic Renewal and the Dynamics of Firms, Networks and Industries