Internet Search Engines based on Artificial Neural Systems implemented in Hardware would enable a Powerful and Flexible Content Based Research of Professional and Scientific Documents

Luca Marchese[1]

[1] Syn@ptics, Genova, 16151, Italy, luca.marchese@synaptics.org, www.synaptics.org

Abstract This document is a feasibility study on the development of a new generation of Internet Search Engines that should enable the professional user to search documents by complex contexts instead of single words linked by logical conditions. The method is well known in the literature and is, often, correlated to the WEB-SOM neural technology. This paper approaches the problem with a new perspective that uses a hierarchical structure of the contexts and the latest neuro-morphic VLSI chip technology. Furthermore, the proposed method is based on a user-friendly X3D (successor of the Virtual Reality Modelling Language) GUI (Graphical User Interface). The user can search documents giving as input a reference document that is used by the neural search engine in order to find similar documents.

Keywords Self Organizing Map, WEB-SOM, Internet Search Engine, Radial Basis Function, Neural VLSI, X3D, Context-Based Data Base Navigation

1. Introduction

In this paper, we describe a feasibility study of the Internet Search Engine based on context instead of a collection of single words. The main difference is that the importance of any single word builds the context. If the word is repeated many times in a document, such a word assumes a higher importance. The vector composed by the number of repetitions of meaningful words constitutes the "fingerprint" of the document. The standard way to search a document is by typing some words that the desired document must contain. The context-search modality is by writing a small prototype of the document we are searching. In such a way, the input could be a document and the output is a list of documents strongly correlated with it.
There have been many studies on this argument, and they were prevalently based on the SOM (Self Organizing Map) developed by Teuvo Kohonen [15]. These studies are, commonly, known as WEB-SOM.
The architecture of the proposed search engine is based on the RBF (Radial Basis Function) neural network [16]. More precisely, on its implementation in hardware on the commercial neural chip CM1K [7].

1.1. The WEB-SOM Model

WEB-SOM means WEB Self-Organizing Maps and is an algorithm that orders an information space [12][13][14]. The map places similar documents in closed spaces. The order helps in finding related documents once any interesting document is found. Fig.1 shows the basic principles of the method and Fig.2 shows a typical map obtained with such algorithm. The experiments made around this algorithm, typically, produced bi-dimensional maps containing "agglomerates" of similar documents. Moving the cursor on such agglomerates the most used words are displayed to the user in order to inform about the context of these documents. There are many different implementations of the WEB-SOM methodology that produce various types of graphical outputs and are, sometimes, organized in hierarchical structures. The main characteristic of the WEB-SOM methodology is the self-organization that characterizes the SOM neural network, often cited as Kohonen Map.

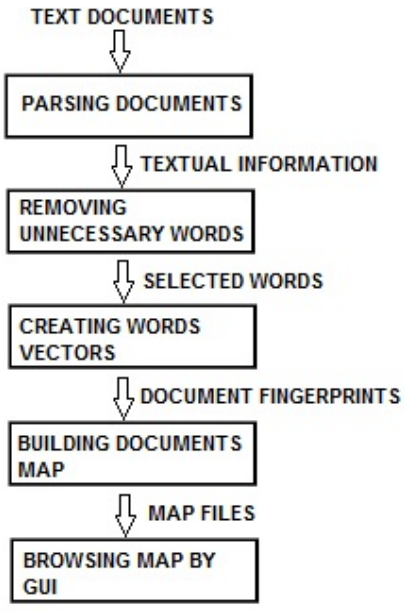

Figure 1. The WEB-SOM methodology.

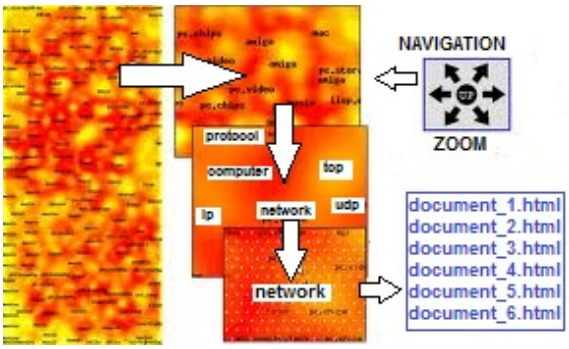

Figure 2. The typical output map of the WEB-SOM. Some words identify the concentration of documents. A dark color indicates a more populated area. Moving the cursor on the words, a list of documents can be displayed in some other window.

2. The New Proposed Methodology

The methodology proposed in this paper is different from the WEB-SOM approach in many aspects. The first is that it uses a supervised neural network model (Radial Basis Function). The second difference is that the hierarchical structure of clusters organizes different levels of sensitivity to the fingerprint of the documents. Another important difference is the use of the latest commercially available neuro-morphic VLSI technology for pattern recognition. Such a technology should enable the servers of the Internet Search Engine to manage multiple concurrent queries of document's fingerprint fuzzy comparisons with the database. Furthermore, we propose an X3D (successor of VRML) browser that enables the user to navigate in a 3D world filled by a hierarchical structure of clusters of documents. The experience is that of moving in a bookstore with nested rooms. As we add new words, we contribute to specialize the desired context and we, automatically, move in a room or we enter in a more specialized sub-room. Direct moving commands could help the navigation. If the user has a reference document, in any format, he can put directly such a document in the browser. The complex fingerprint extracted shall be sent to the Internet Search Engine that shall compare it with the entire database responding directly with a list of similar documents.

This technology has been already presented, in 2000, by L.Marchese at fourth "International Conference on Cognitive and Neural Systems" [1], and at and at "First International Zero Instruction Sensory Computing Conference" [2]. The concept of "Pattern Recognition Neural Server" was already presented by L.Marchese in 1999 at third "International Conference on Cognitive and Neural Systems" [3] and ANNIE 2000 [4][5]. In the year 2000 the demand and requirements of the internet users were not sophisticated as today, and this proposal arrived, probably, to early. Furthermore, the commercial silicon neuro-morphic technology was, at that time, limited to 36 neurons in a chip [6] while the current technology has 1000 neurons in a chip [7].

2.1. The Radial Basis Function Model and CM1K Neuro-Morphic Chip

A radial basis function network (Fig.3) is a neural network that uses radial basis functions as activation functions [16]. It is capable of representing complex nonlinear mappings and is widely used for function approximation, time series prediction, and control.

The CM1K chip is a hardware implementation of an RBF neural network. The neurons are capable of ranking similarities (L1 or LSUP distance as described in Fig.4) between input vectors and the reference patterns they hold in memory. The neurons report conflicting responses or cases of uncertainty, as well as unknown responses or cases of anomaly or novelty. The time necessary to obtain a response is independent of the number of committed neurons in the network.

The model generator built-in the CM1K chip makes it possible to learn examples in real-time when they drift from the knowledge residing in the current neurons. The "novelty" examples can be stored in neurons assigned to a different context to allow a supervised verification and to learn at a later time.

The ability to assign the neurons to different contexts or sub-networks allows building hierarchical or parallel decision trees between sub-networks. This behavior leads to advanced machine learning with uncertainty management and hypothesis generation. Fig.5 and Fig.6 show the learning process respectively, and the recognition process flows in the CM1K chip. Fig.7 shows the board NeuroStack that contains four CM1K chips for a total of 4096 neurons and a stack of multiple boards. The user can stack these boards together, and they self connect in daisy-chain: two boards constitute an RBF neural network of 8192 neurons and so on.

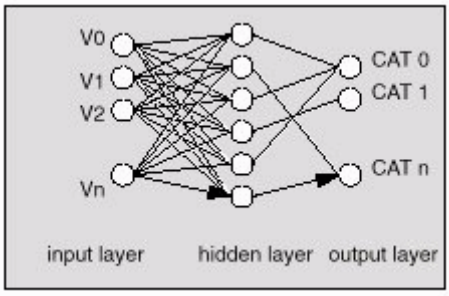

Figure.3 Radial Basis Function architecture.

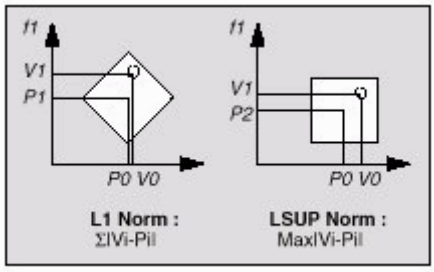

Figure 4. L1 and LSUP distance calculation.

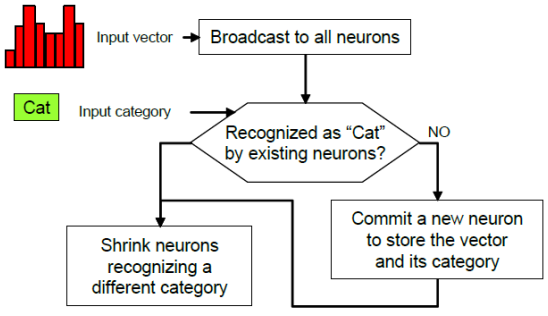

Figure 5. Learning process flow chart in CM1K.

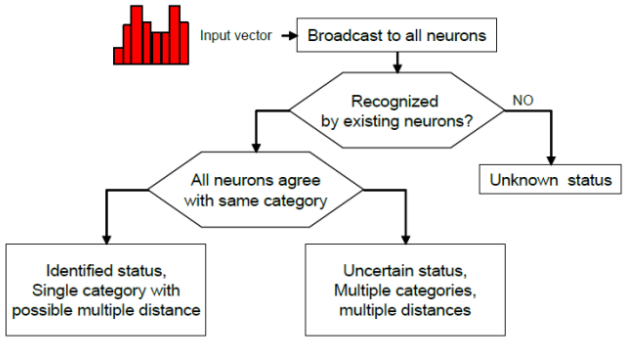

Figure 6. Recognition process flow chart in CM1K.

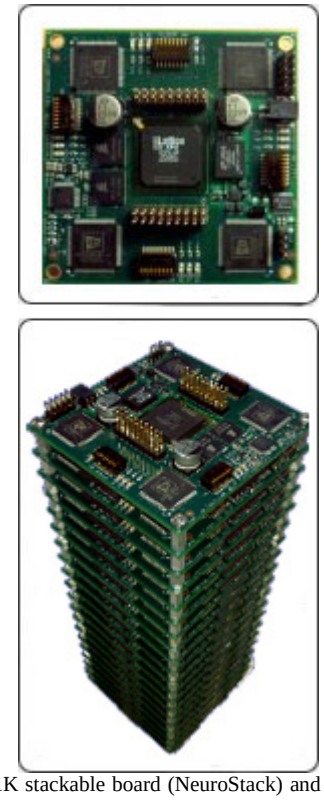

Figure 7. A CM1K stackable board (NeuroStack) and a stack of multiple boards in daisy-chain connection.A stack of multiple boards works like a large neural network.

2.2. Dictionary building process

Many WEB-SOM papers describe this process that has the purpose to build a dictionary of meaningful words by analyzing a huge collection of documents. The meaningful words are those that can be considered useful to discriminate between contexts. In this paper, we briefly analyze this process introducing some new algorithmic proposals. We must note that the scope of the mentioned studies on the WEB-SOM was quite limited. We think that other methodologies could be more efficiently applied in order to build a dictionary of meaningful words in a comprehensive context like that of the entire Internet contents. In this study, we use the CM1K in order to create the dictionary containing all the meaningful words for arguments discrimination. However, we want suggest that a hash algorithm could efficiently perform this procedure in a conventional computer. The CM1K can compare the current word with all the stored words in parallel mode without the use of hash algorithms and then increase the counter associated with the firing neuron category. In order to perform this operation, we must set MIF=0 and MAF=0 (MInimum influence Field and MAximum influence Field). Infact we need a perfect match between the pattern and the prototype: words must be equal and not similar. Therefore, the L1 or LSUP distance must be null. Two counters are associated to any neuron. The first counts the number of times the word is present in all documents (TCNT). The second counts the number of documents in which the word has been found (DCNT). Finally, elaborating this information, we can decide which words are meaningful in order to discriminate between arguments. A conventional computer program implements these counters that are external to the CM1K. Fig.8 shows the procedure flow of this process. When we have stored the words on the CM1K(s) memory, we need to optimize the dictionary, removing words not meaningful for arguments discrimination. We propose two rules in order to perform this operation:

Meaningful_Flag [Word [Class]] = True
if both the following conditions are true:

1. (Min_1 < TCNT[Class] < Max_1) = True
2. (Min_2 < DCNT[Class] < Max_2) = True

If (TCNT < Min_1) then the word is used too few times to be associated with a context. If (TCNT > Max_1) then word is probabilistically evaluated as a common use word and thus, it has not context discrimination value. Min_1 and Max_1 should be empirically or statistically evaluated. Min_2 = (tot_no_of_analyzed_docs / G) * (W/100). The context granularity of the database is forced to G (the maximum number of contexts/categories) . If the number of documents containing the specified word is lower than W% of the mean dimension of one context/category, then the word is not valid for the context/category discrimination. Max_2 = (tot_no_of_analyzed_docs * C) / G. If the number of documents having at least one element of the specified word is matching more than C contexts/categories, the word is considered not valid for the context/category discrimination. TCNT is the total counter of words, DCNT is the counter of documents containing the word and Class is the class associated with a word.

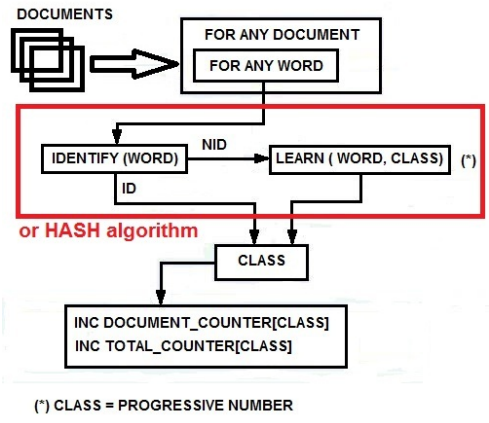

Figure 8. The dictionary building process

When the Boolean vector of meaningful words is complete, it is possible to optimize the memory of the CM1K(s) removing the not utilized prototypes (those prototypes associated with a false flag). We can make it knowing that CLASS = NEURON_INDEX because of CLASS is assigned to a new committed neuron, during learning, as the sequential index of the word. We need to build an algorithm that shift down the memory of CM1K in any position is needed (from a position to top) up of the required size correcting the associated classes. We have to access in this case the CM1K in SR mode (Save/Restore) in order to disengage useless prototypes.
Now we can save the synaptic memory of the CM1K on a file and reload it to recognize words from documents. The number of neurons utilized (equal to the number of the valid words) is the dimension of the "fingerprint" of documents.
Documents clustered by source and clusters logically ordered will result in such a degree of context linearity in the dictionary. This property can be an important improvement of the behavior conditions of the document vector compression.

2.3. Documents fingerprint extraction and compression

We build a histogram of any dictionary's word to extract the document's fingerprint. The document's fingerprint is a vector with the dimension of the dictionary. This process will be quickly performed reading any word of the document and identifying its byte-vector with an "Identify" operation on the CM1K loaded with the synaptic memory representing the dictionary. Fig.9 shows this procedure. Alternatively we could use a Hash algorithm because the identification of the word is, again a "perfect match" comparison.

	IN 0	IN 1	IN 2	IN 3	IN 4	IN 5	IN 6	IN 7	IN 8	…	IN M
OUT 0	0	1	0	0	0	1	0	0	1	…	0
OUT 1	1	0	0	0	0	1	1	0	0	…	0
OUT 2	0	0	1	0	1	0	0	1	0	…	0
OUT 3	1	0	0	0	1	0	0	0	1	…	0
…	…	…	…	…	…	…	…	…	…	…	…
OUT N	0	1	0	1	0	0	0	0	0	…	1

Table 1. Random Bits Matrix for compression M to N. The component <n> is computed as the sum of the components of the original vector corresponding to bits=1. The number of bits=1 in each row is fixed.

	IN 0	IN 1	IN 2	IN 3	IN 4	IN 5	IN 6	IN 7	IN 8	…	IN M
VECT 0	0	1	0	0	0	1	0	0	1	…	0
VECT 1	1	0	0	0	0	1	1	0	0	…	0
VECT 2	0	0	1	0	1	0	0	1	0	…	0
VECT 3	1	0	0	0	1	0	0	0	1	…	0
…	…	…	…	…	…	…	…	…	…	…	…
VECT N	0	1	0	1	0	0	0	0	0	…	1

Table 2. Random Bits Matrix for compression M to 256. The components of the original vector of size M are selected if the corresponding bit value is 1. There are 256 bits=1 in each row. The table builds N vectors of 256 components.

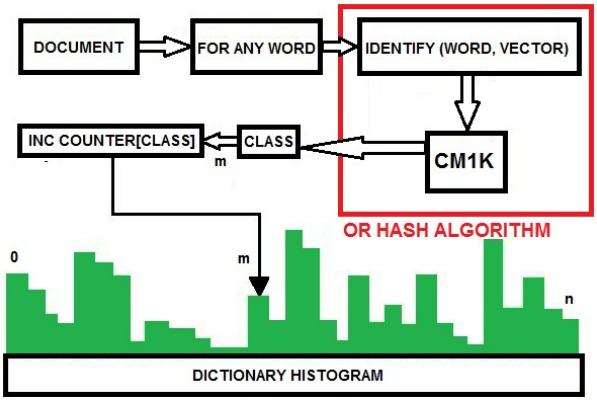

Figure 9. The document fingerprint extraction process

We have an N-dimensional vector for any document of the collection. N is the dimension of the dictionary and is too large for any successive identification (N > 10000). We require a compression of the vector that does not affect similitude between couples of vectors. More precisely, the compression must not affect the Euclidean distances between any possible couple of vectors (in our case the Euclidean distance is replaced by the L1 or LSUP). In the WEB-SOM work (Kohonen), a compression based on vertically normally distributed random bits matrix was used in order to compress the dictionary vector (Table.1). The formula used to compute the new reduced vector is:

$$V'[n] = \sum_0^M V[m] \times Bit[n][m] \qquad (1)$$

We want use the CM1K to perform the compression task. We have 10000 components vectors and need to have equivalent vectors of 256 components. The equivalence is referred to the normalized distance between couples of vectors: we do not use Euclidean distance but the L1 or LSUP distance in order to have compatibility with the CM1K. The final vector should be recognizable by the CM1K (256 components byte wide comparison) in one single recognition operation. The compression is performed using a matrix with 256 randomly distributed bits "1" in each row that builds a 256 components vector as a result of the AND operation with the original vector (Table.2). The CM1K(s), previously trained with fixed random patterns recognizes the vector and the matching class is one component of the new compressed vector. The training set is composed of 256 patterns (256 components sized) randomly generated. These patterns are sorted on the basis of the L1 distance with a K-Nearest-Neighborhood procedure and assigned to an incremental category number. Furthermore, the reference patterns must have a minimal distance (a defined threshold) from the nearest patterns assigned to the lower and higher category number. We repeat this step with different Boolean vectors for any component of the new compressed vector (Fig.10). This compression is more reliable than the simple Bits Matrix compression: one of the reasons can be intuitive as explained in Fig.11 and Fig.12. The new vectors conserve the similitude property better than the random bits matrix processed vectors. This property is true if the documents contain a quite high number of words of the dictionary. The system behaves better working on the database of large documents that little ones, as well as any other text-analysis algorithm. The test to verify the similitude-safe property is following explained. We build a database of pseudo-random

vectors composed of 10000 components and chose a reference vector. The reference vector is compared using a normal program of L1 distance computation with all the other vectors. We insert in the list all the vectors having normalized distance < D_MAX (2). This process is very long and could be performed quickly using an appropriate algorithm on the CM1K. However, we use a simple program on a conventional computer, because it is used only one time for a test.

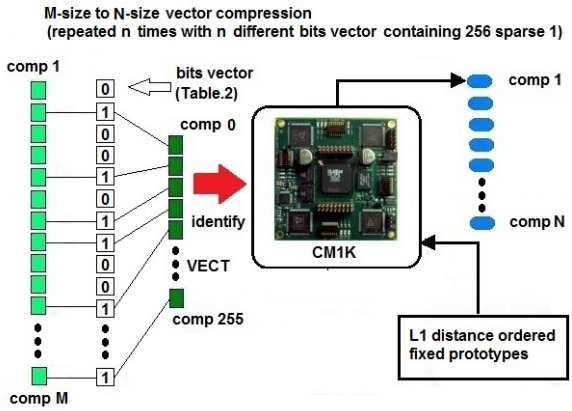

Figure 10a. Random-bits-matrix compression with L1 or LSUP distance recognition based on random reference prototypes . The reference prototypes are ordered by L1 distance (K-nearest-Neighborhood) and assigned to an incremental category number (Fig.10b).

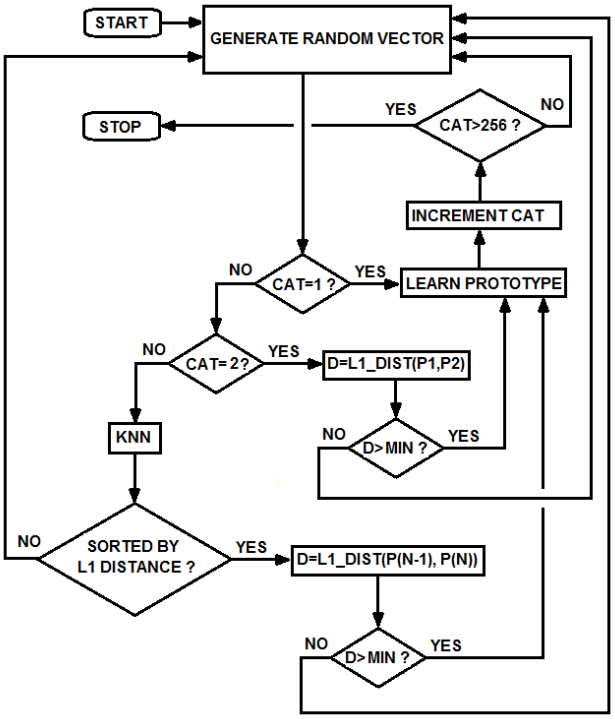

Figure 10b. The flow-chart of the procedure used to sort the reference prototypes by L1 distance. Any randomly generated vector is compared with all the previous vectors and they are sorted by L1 distance with the KNN algorithm (K-Nearest-Neighborhood).

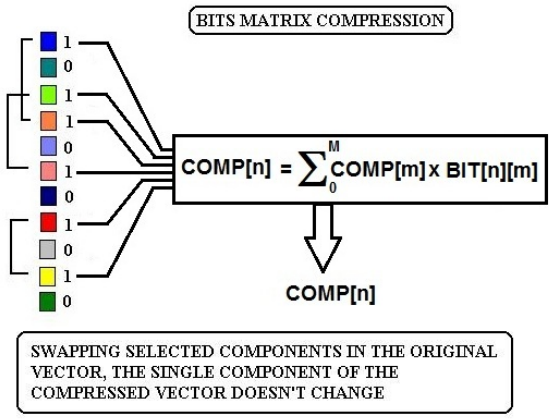

Figure 11. The standard random-bit-matrix compression methodology: swapping couples of selected components in the original vector, the single component of the compressed vector does not change.

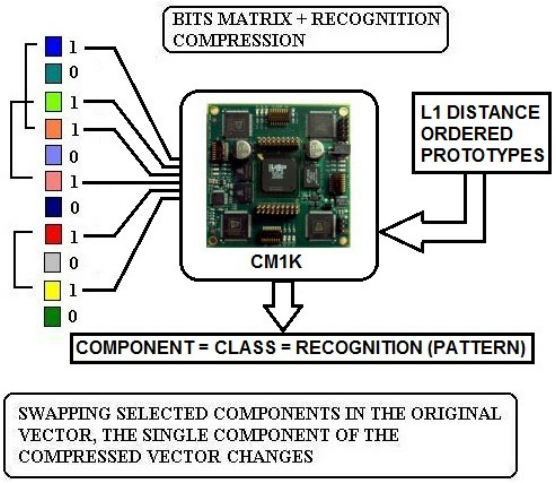

Figure 12. In the proposed random-bits-matrix compression followed by L1or LSUP distance recognition swapping couples of selected components in the original vector, the single component of the compressed vector changes as required.

We repeat the process with the equivalent 256 components compressed vectors. Then we insert in the second list the vectors having normalized distance < D_MAX. The equivalence of the two lists should mean a 100% similitude property safe: a value over 70% is acceptable. We repeat the test with many target vectors. The standard deviation (3) of the results obtained should be very limited in order to consider this test reliable. At the end of this process, the average of the results should be considered.

$$
\begin{aligned}
D &= L1\_d / N \\
D &= normalized\_L1\_dist \\
L1\_d &= L1\_dist \\
N &= vector\_size
\end{aligned}
\qquad (2)
$$

$$\sigma = \sqrt{\frac{\sum_{n=1}^{N}(x-\mu)}{N}}$$
$$N = number\_of\_samples$$
$$x = sample\_result \quad (3)$$
$$\mu = mean\_result$$
$$\sigma = stnd\_dev$$

2.3.1 Examining the output of this process

The output of this process is a list of records composed of [URL][VECTOR][MUW] related to the examined documents. The URL is Uniform Resource Locator for the specified document while VECTOR and MUW are respectively the fingerprints and the most used word of the document (Table.3).

URL	VECTOR	MUW
http://domainX.dir1.docZ.html	[230][234][200]...[010]	transistor
http://domainY.dir1.docW.html	[240][134][002]...[030]	optical
http://domainZ.dir1.docY.html	[130][034][005]...[050]	car
http://domainZ.dir2.docD.html	[030][114][135]...[120]	network
.....	.....	.....

Table 3. Association of URL, fingerprint and Most Used Word

2.4. Hierarchical clustering and 3D mapping

As we have explained above, the system is based on a hierarchical structure of clusters or "containers" that we can see as "nested" rooms of a bookstore. The level <0> is the most detailed level or, using the bookstore's simile, is the room where you can find, finally, the books you are searching. The hierarchy replicates upper levels a certain number of times (the granularity of the hierarchy). These levels are here, generically, indicated as level <n>.

2.4.1. Level <0> clustering

When we have a table with records <URL><VECTOR><MUW> we must put them into clusters basing this operation on the distances between their vectors that we have called "document fingerprint." We want perform this process as fast as possible using the recognition of the Radial Basis Function hardware implementation. The choice of CM1K neural chip is related to the easy to understand behavior and programming interface and mainly for its "unlimited" expandability at no cost of performance.
The process of clustering (Fig.13) is a hybrid situation where some aspects of supervised and unsupervised learning behave together. We must perform the learning process without knowing a-priori classes associated with patterns, but, at the same time, we need to associate a class to any cluster. The choice is to associate an incremental number linked with the commitment of a new prototype neuron in the Radial Basis Function neural network. We select MAF (Maximum Influence Field)[7] following the relation: MAF = f (DB_SIZE, NN_MAX_SIZE); DB_SIZE is the size of the database and NN_MAX_SIZE is the maximum number of neurons available.When the input pattern does not match within the influence field of any other existing prototype, the learning process commits a new neuron. This new neuron has MAF as influence field. The new prototype connects with a new cluster. The learning process never reduces the influence field of a prototype: the category-mismatch condition cannot happen [7] because any new category is a sequential number. When the CM1K learns a pattern, we memorize the URL (Uniform Resource Locator) of the document and its most used word (MUW) in a database. The key of such a database is simply the number of the cluster. The relation is "one to many" because any cluster contains many URLs with the associated MUW (Table.4 and Table.5).

CLUSTER_0	VECTOR	URL	MUW
23403	[120] [030] ... [240]	http://domainX.dirX.docX.html	neural
		...	...
		http://domainX.dirX.docX.html	Kohonen

Table 4. Example of records for the key CLUSTER = 23403 at the level<0>

CLUSTER_0	VECTOR	URL	MUW
…	…	…	…
23403	[120] [030] ... [240]	http://domainX/dirX/docX.html	neural
		...	...
		http://domainX.dirX.docX.html	Kohonen
…	…	…	…
30000	[220] [130] ... [140]	http://domainX.dirX.docX.html	RBF
		…	…
		http://domainX.dirX.docX.html	KNN
…	…	…	…

Table 5. The table with more clusters at the level<0>. A cluster contains many URLs and the associated MUWs.

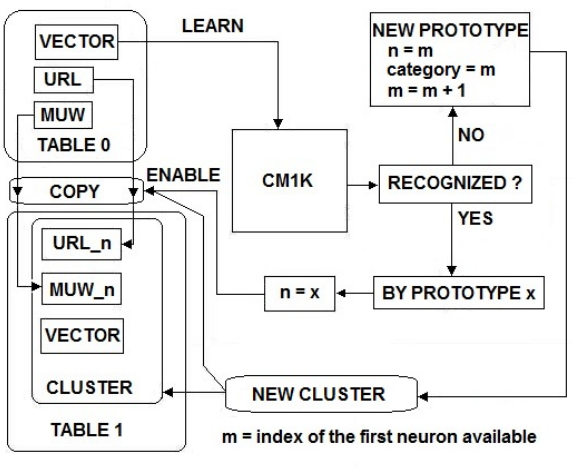

Figure 13. Clustering process from documents to level<0> clusters

2.4.2. Level <n> clustering

The upper clustering levels supply a hierarchical structure for the navigation of the database. The number of clustering levels is a function of the database dimension and the degree of roughness that best fits a meaningful navigation. Starting from n=1, any cluster of Level<n> does not contain URLs but the Level<(n-1)> clusters. An MUWs list connects with any Level<(n-1)> cluster, and a number identifies any cluster (Table.6 and Table.7). In order to enable a 3D navigation, we need to add at this record some information that represents the x-y-z position of the cluster in 3D space. We perform this operation with a recognition operation on the vector divided into three elements, using CM1K(s) trained with predefined pseudo-random patterns (Fig.14 and Fig.15):

X = CLASS(k = 0 - m STEP 3) { V[k] }
Y = CLASS(k = 1 - m STEP 3) { V[k] }
Z = CLASS(k = 2 - m STEP 3) { V[k] }

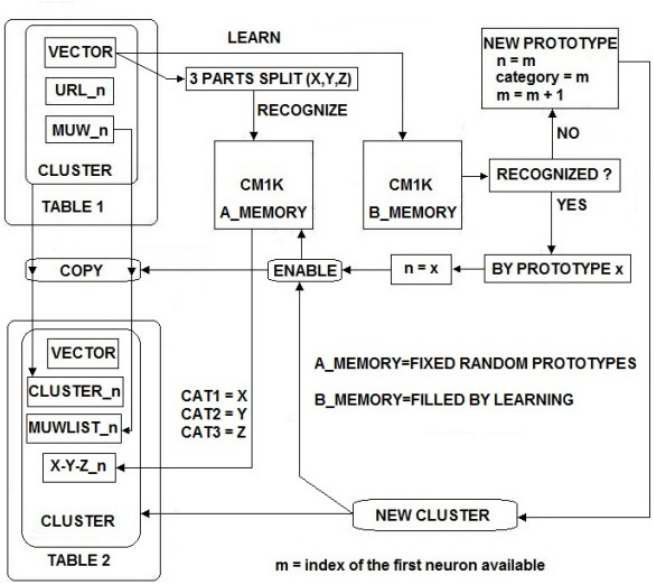

Figure 14. Clustering process from level <n-1> to level <n> clusters

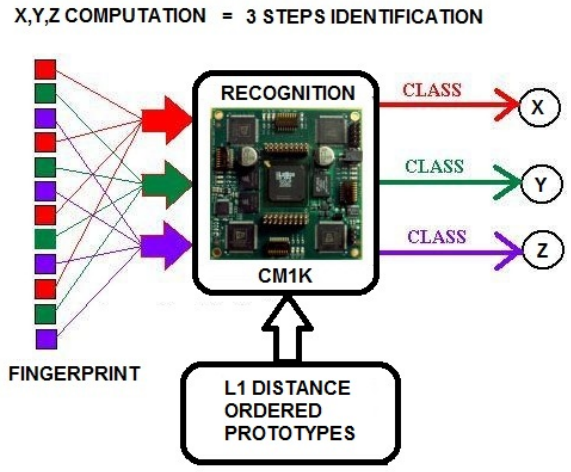

Figure 15. The computation process of coordinates. We split the fingerprint into three parts, and the CM1K recognizes these parts separately. We use a database of fixed random patterns as neural memory of the CM1K. We use the class (range 0-16000) as a coordinate.

2.5. From clusters to X3D

The process described here is required in order to obtain a full X3D hierarchical description of the database space. X3D (successor of VRML) is featured to describe tri-dimensional spaces and environments containing three-dimensional objects. X3D files are files that can be interpreted by X3D browsers. An X3D browser enables the navigation of the spaces described by X3D files. Our objects are very simple geometrical shapes because they must represent the clusters as “containers” or “rooms”. Fig.16 shows how the data contained in cluster tables are used to build X3D files.
Fig.17 shows the process that creates a hierarchical structure of X3D files starting from cluster tables.

CLUSTER n	VECTOR	CLUSTER n-1	MUW-LIST	X	Y	Z
2366	[120] [030] … [240]	23403	WORD1, ...n	2404	1230	240
		...	WORD1, ...n	...	...	...
		32240	WORD1, ...n	2400	1040	200

Table 6. Example of records for the key CLUSTER = 2366 at the level<n>

CLUSTER n	VECTOR	CLUSTER n-1	MUW-LIST	X	Y	Z
…	…	…	…	…	…	…
2366	[120] [030] … [240]	1	WORD1, ...n	2404	1230	240
		...	WORD1, ...n	...	...	...
		32240	WORD1, ...n	2400	1040	200
…	…	…	…	…	…	…
3000	[230] [200] … [100]	1	WORD1, ...n	1600	200	500
		…	WORD1, ...n	…	…	…
		2000	WORD1, ...n	340	2340	1300
…	…	…	…	…	…	…

Table 7. The table with more clusters at level<n>. The sub-clusters of level<n-1> have their MUW-LISTs and coordinates associated.

2.6. X3D Data Base Navigation

The first access to the database is the highest level X3D file in a hierarchical structure. Moving in this space, we can see many objects representing lower level clusters positioned dependently of their associated vector. Thus, we can see zones more populated of objects than others.
The user can walk, fly, translate, tilt, yaw, pitch and roll using keys on the browser or a specialized joystick. When the cursor is on an object, the browser shows, in another frame, the list of words associated with the cluster that is represented by this object. The X3D file contains this list of words. Looking at these words, we can understand the contents of the cluster and decide if it is interesting for our research. In this case we can click on the object obtaining the download and visualization of the X3D file associated. This type of navigation looks more enjoyable than useful and must be enhanced enabling the user to move in this world inputting complex context information.

2.7. Context-based navigation

We need to enable the user to move in the 3D space not only making direct X3D movements, but also putting context information. There are two possible scenarios of this behavior. In the first, the user has a "reference document" and wants find similar documents. In this case, the clustering processes explained in Figs.13, and 14 can manage the navigation. The user can upload the "reference document" into the browser, and its fingerprint is sent to the Internet Search Engine. The Internet Search Engine compares the fingerprint with the database using a Pattern Recognition Neural Server based on CM1K boards (Fig.19). The user is then automatically transported in the 3D world in the sub-space addressed by the fingerprint of the "reference document". The user could, optionally, select if he wants confirm manually any sub-cluster passage. If the fingerprint of the "reference document" is enough detailed, the user could be directly transported to the deepest bookstore room containing the "books". If the fingerprint is not enough detailed or it does not fall in the "influence field" of any deepest "room", the user could be stopped at any clustering level in any x-y-z position. In such a case, the user should navigate from that point with direct 3D movements helped by the MUW-lists associated to clusters that are visible in the window of the browser when left-clicking on the cluster.

In the second scenario, the user does not have a reference document and needs to build incrementally a context by typing words in the window. In this second case, a real fingerprint cannot be constructed reliably because the user should be asked to input too many words and indicate the "weights" of them for the context. In order to manage a navigation driven by small reference context, we have proposed the clustering process of Fig.17. Comparing such a process with that of Fig.14, the reader can note that we have added the management of a new table "word-to-coordinates". Fig.18 explains the navigation process. Any time the user inputs a new word the server calculates the contribution of such a word to build a new position and sends these coordinates to the client browser. The server and the client share the computation of the new position. The server sends the coordinates associated to that word in a specific cluster. The server finds the coordinates in the corresponding table "word-to-coordinates". The client (the browser) averages these coordinates with the current coordinates in order to build the final position. The clustering process of Fig.17 builds a word-to-coordinates table for any cluster at any clustering level. This navigation process does not require the pattern recognition capability of the Neural Server, but it is not exactly the most important target of this feasibility study. We believe that the real revolution in the research of scientific and professional documents is that one performed uploading the "reference document" as shown in Fig.19. The browser should accept any document format (Txt, RTF, Pdf, Word, PostScript), and the server should extract the fingerprint. Then, the server asks for a pattern recognition service to the CM1K server and forwards the results (coordinates or cluster number) to the client (browser). In the next chapter, we propose an alternative simplified Internet Search Engine based on multiple dictionaries associated with specific disciplines.

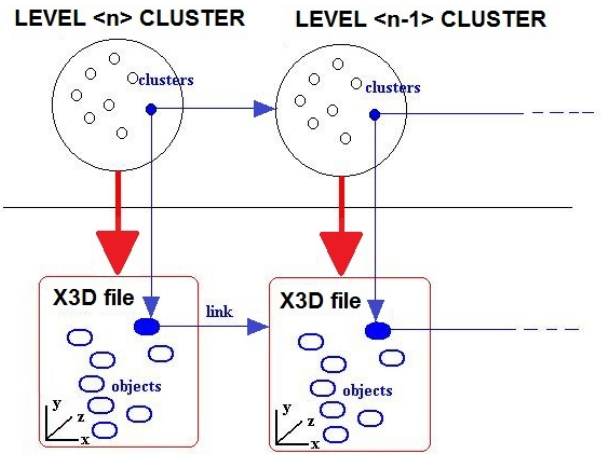

Figure 16. CLUSTER_n associates with an X3D file and contains the CLUSTER_(n-1) - object inside X3D file. The server dynamically builds the X3D file from the table. The x, y, z are the coordinates of the object. The MUW-LIST associated to the object is displayed by the browser when the cursor is on the object.

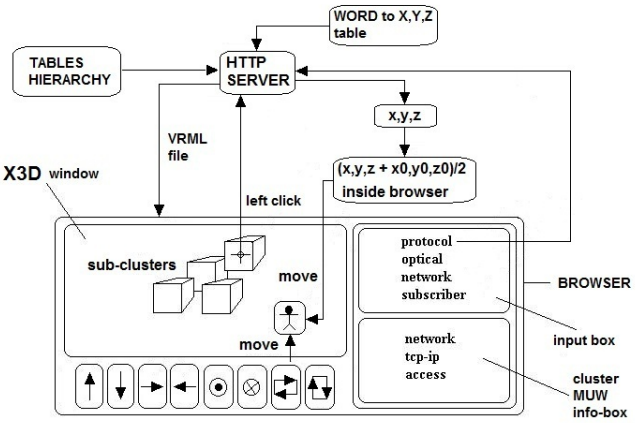

Figure 18. The Internet X3D browser queries the Internet Search Engine with some words that build a small context system.

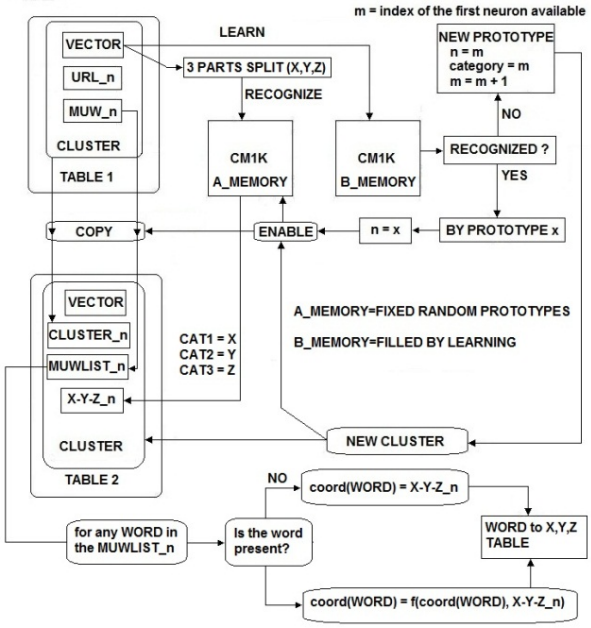

Figure 17. Clustering process from level n-1 to level n clusters with the management of queries containing a small number of words (small context) instead of a real reference document.

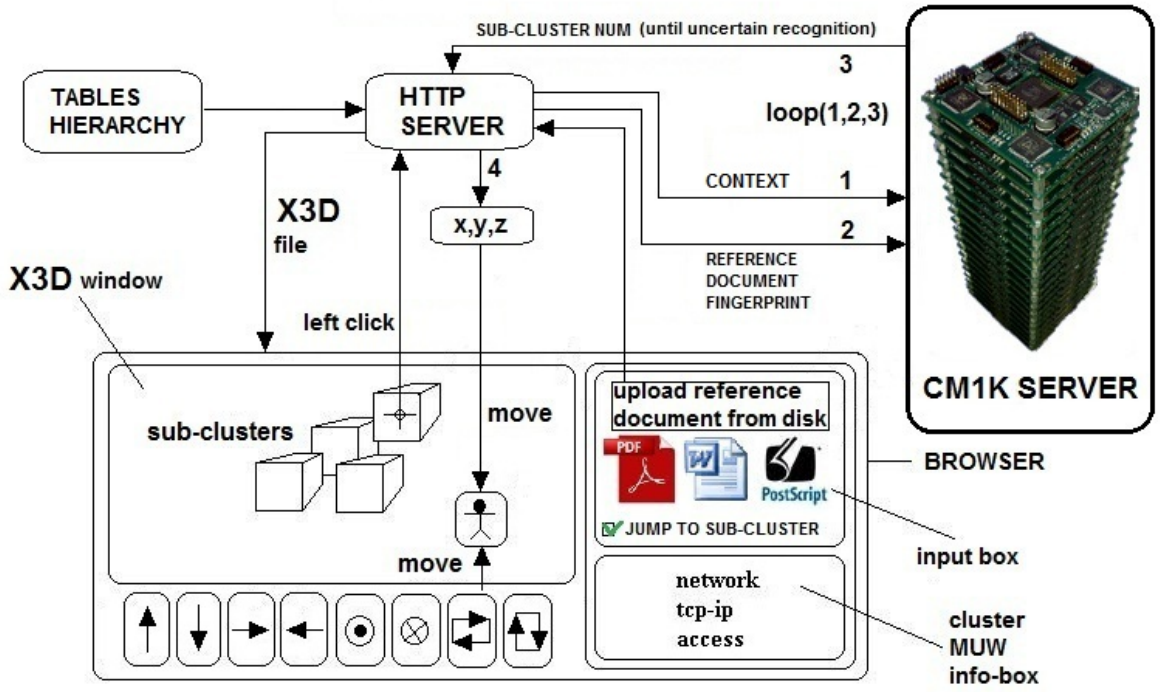

Figure 19. The Internet X3D browser queries the Internet Search Engine with reference documents. The "CONTEXT" sent by the HTTP server to the CM1K server is not referred to the document but is an internal parameter of the CM1K that selects the appropriate synaptic memory for the hierarchy level and the current cluster.

3. Fingerprint-based on Specialized Dictionaries

There is an alternative simpler way to build context based Internet Search Engines. Instead of considering context in a continuous space we could decide to discretize this space considering a large number of specialized dictionaries dedicated to specific disciplines from economy to science and technology. A hierarchy with different levels of specialization organizes the system. Experts in any specific discipline could build specialized dictionaries. A "one to many" database relation should link any specific word to dictionaries at different levels of the hierarchy. The fingerprint of a document is the number of words linked to any specific dictionary. The fingerprint's size is the number of dictionaries for the current level of the hierarchy. The server uses the database of relations word-dictionary in order to build the fingerprint of the document. The Internet Search Engine, as usually, scans the net contents and builds the fingerprints of the documents updating the database based on Table 3, Table 4 and Table 6. In our proposal, we fix the number of dictionaries for any level of the hierarchy to 256 in order to be compatible with the maximum dimension of the CM1K vector. The fingerprint of a document is a 256 elements vector that can be computed counting the document's words belonging to the 256 specialized vocabularies. From the model described in the first part of this document, only the fingerprint extraction modality is different, and we could consider valid any other process described. The advantage of this approach is the higher reliability of the fingerprint. The drawback is the need to fix, a priori, a comprehensive hierarchical structure of the global knowledge available on the Internet. Indeed the origin of a new discipline would not be tracked by the system. On the contrary, this new discipline would be tracked by the system based on the global dictionary's fingerprint, provided that this discipline does not introduce too many neologisms. Fig.21 shows the fingerprint extraction process based on specialized dictionaries. In order to perform this process with the CM1K (MIF=MAF=0), a previous learning process is required (Fig.20).

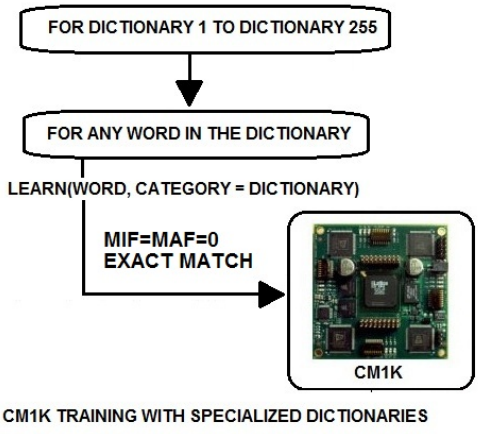

Figure 20. The process related to the association of words to multiple dictionaries

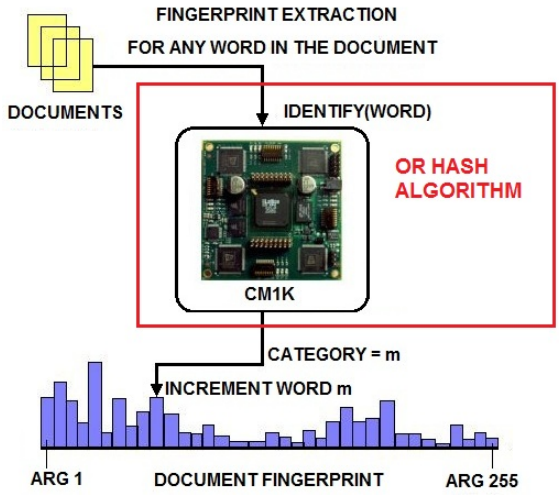

Figure 21. The fingerprint extraction process from a document in a multiple dictionaries framework.

4. Conclusions

In this paper, we have presented a feasibility study for the realization of a new generation of Internet Search Engines. These engines are more suitable for professional customers that need to search documents by complex contextual information instead of simple collections of words linked by logical conditions. In this study, we have used the latest neuro-morphic VLSI pattern recognition technology and the most sophisticated Internet navigation languages. The use of pattern recognition specialized chips enables the real-time management of multiple clients queries on a database of document fingerprints. The server must analyze a huge quantity of fingerprints in a fuzzy way that is not suitable for conventional processors.
We believe that this is the right time for a new generation of Internet Search Engines that could search images and documents following a fuzzy "brain-like" philosophy instead of an Aristotelian or Boolean method.

ACKNOWLEDGEMENT

We want to thank Anne Menendez and Guy Paillet of General-Vision for the technical support on the CM1K chip technology and the suggestions and feedbacks on this feasibility study.

REFERENCES

[1] L.Marchese, "Distributed Database content based navigation and Internet search engines powered by neural VLSI". - "Fourth International Conference on Cognitive and Neural Systems"- (27/05/2000) - Department of Cognitive and Neural Systems – Boston University S

[2] L.Marchese, "Neural VLSI for Internet and telecommunications technology" - First International Zero Instruction Sensory Computing Conference (LeCorum Montpellier) (27/07/2000)

[3] L.Marchese, "Neuromorphic VLSI server" - "Third International Conference on Cognitive and Neural Systems" (27/05/1999) Department of Cognitive and Neural Systems - Boston University

[4] L.Marchese "A neural network chips based server" - ANNIE 2000: Smart Engineering System Design (05/08/2000) (University of Missouri-Rolla)

[5] L.Marchese "Intelligent engineering systems through Artificial Neural Networks: a neural network chips based server" - ASME PRESS - Editors: Cihan Dagli, Anna L.Buczak, Joydeep Ghosh, Mark J.Embrechts, Ocan Ersoy, Stephen Kercel

[6] IBM, Silicon Recognition, "ZISC036 Operational Manual"

[7] General Vision Inc, "CM1K Technical Manual", www.general-vision.com

[8] General Vision Inc, "CM1K, the simplest API", www.general-vision.com

[9] General Vision Inc, "CM1K The fastest KNN chip", www.general-vision.com

[10] General Vision Inc, "NeuroMem Decision Space Mapping Manual", www.general-vision.com

[11] General Vision Inc, "NeuroMem Technology Reference Guide", www.general-vision.com

[12] Kohonen, T. (1982). "Self-organized formation of topologically correct feature maps." Biological Cybernetics, 43:59-69.

[13] Honkela, T., Kaski, S., Lagus, K., and Kohonen, T. (1996). "Exploration of full-text databases with self-organizing maps." Submitted to ICNN-96, Washington D.C.

[14] Kaski, S., Honkela, T., Lagus, K., and Kohonen, T. (1996). "Creating an order in digital libraries with self-organizing maps" Submitted to WCNN-96, San Diego, California.

[15] Kohonen, T. (1995). "Self-Organizing Maps" Springer, Berlin, Heidelberg.

[16] Buhmann, Martin D. (2003), "Radial Basis Functions: Theory and Implementations", Cambridge University Press, ISBN 978-0-521-63338-3.