

CogniMem Reference Guide

Parallel neural network for pattern recognition

Version 1.3.1
(Revision May, 2009)

General Vision Inc.
1150 Industrial Avenue, #A, Petaluma, CA 94951, USA
+1-707-765-6150 / www.general-vision.com

CogniMem Technology Reference Guide

© 2007 General Vision Inc.

All rights reserved. No parts of this work may be reproduced in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems - without the written permission of the publisher.

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.

While every precaution has been taken in the preparation of this document, the publisher and the author assume no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document. In no event shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

CogniMem is a trademark of NorliTech LLC.

Table of Contents

CogniMem Technology Reference Guide	2
Table of Contents	3
Introduction	5
Key features	5
Building a decision space.....	6
Representation of a decision space	6
A Decision space Is a Feature space.....	7
Construction of a decision space	7
Using the decision space for recognition.....	8
Classification status	9
Exact-match classification.....	9
Best-match classification	9
Detailed classification	9
Shaping the decision space with a Norm	10
Norm L1 (Manhattan distance)	10
Norm L Sup.....	10
Which Norm to use?	11
Refining the decision space by adapting the neurons' Influence Fields	11
Enforce conservatism with counter examples	12
Enforce conservatism with the Maximum Influence Field.....	12
Modulate uncertainty areas with the Minimum Influence Field.....	13
Benefit of uncertainty zones	13
Iterative Learning.....	14
Incremental Decision Space mapping	14
Convergence of the decision space	16
Learning curve and feature relevancy	16
Impact of sample variations	16
Conclusion	17
Recognition model.....	18
KNN (K-Nearest Neighbor)	18
RBF (Radial Basis Function)	18
Neural network as Knowledge base.....	19
CogniMem Decision Space Summary.....	19
Network architecture	20
Neuron parallel bus	20
Segmentation per context	21
Multiple contexts for multiple parallel decision criteria.....	21
Multiple contexts for hierarchical decision	21
Network operations	23
Vector broadcasting.....	23
Vector Recognition	24
RBF Classification.....	24
KNN Classification	25
Type of recognition per applications	25
Vector Learning	26
Types of neuron interactions	26
Global settings prior to learning.....	26
Waiving dependency from the teaching sequence.....	27
Save and Restore mode (SR)	28
Activation of the SR mode	28
Sequential access to the neurons.....	28
Reset chain	28

Direct access to a neuron	28
De-activation of the SR mode	28
Example of Save and Restore sequences	28
The neuron entity	29
Neuron Status	29
Neuron Components	29
Neuron memory	29
Distance evaluation unit	30
Associative logic	30
Learning logic	30
Neuron bus	30
Static registers	30
Neuron Context Register (NCR)	31
Category (CAT)	31
Identifier (NID)	31
Dynamic registers	31
Active Influence Field	31
Distance	31
Degenerated flag	31
CogniMem Register Summary	32
Neuron registers	32
Global registers	33
Knowledge File Management	34
Bibliography	35
Index	36

Introduction

Neural networks have been extensively discussed in the late eighty's and DARPA's recommendation in a survey published in 1987 was to "go silicon". As a neural network is inherently a group of elements with the same behavior, it is a candidate for a parallel architecture. While it has been claimed by many publications that neural networks could benefit from a parallel architecture, most current implementations are running on standard computers, which suffer from the following limitations: (1) They execute one instruction at a time (sometime four with quad cores); (2) A good portion of the data bandwidth is dedicated to fetching and decoding instructions before to executing them. Furthermore, the emerging concept of parallel architectures based on multiple processors indicates that synchronization between these processors can become a serious hurdle. As a result, a neural network implemented by software and running on computer(s) cannot be defined as truly parallel.

The CogniMem chip is a fully parallel silicon neural network: it is a chain of identical elements (i.e. neurons) addressed in parallel and which have their own "genetic" material to learn and recall patterns without running a single line of code and without synchronizing to any supervising unit. A resulting achievement of this architecture is a constant learning and recognition time regardless of the number of connected neurons. A second achievement is the ability to expand the size of the network by cascading chips. CogniMem is not the first of its kind. It is actually the successor of the ZISC (Zero Instruction Set Computing) chip invented by Guy Paillet and jointly developed and patented with IBM. IBM released the first ZISC chip with 36 neurons in 1993, followed by the ZISC78 with 78 neurons in 1999. The first CogniMem chip has 1024 neurons.

KEY FEATURES

Knowledge representation

- Radial Coulomb Energy (RCE) neural network
- Choice of two non-linear classifier: [Radial Basis Function \(RBF\)](#) or [K-Nearest Neighbor \(KNN\)](#)
- Automatic adaptive [model generator](#)
- [Save and Restore of the knowledge](#) of the neurons

Recognition

- [Classification status](#) can be identified, uncertain or unknown
- Response of all the firing neurons is accessible leading to uncertainty management and hypothesis generation

Parallel [network architecture](#) on silicon

- Thousands of [neurons](#) (cognitive memories) working in parallel to recognize a same input pattern
- Unlimited network size by cascading chips through a parallel bus
- Learn in 18 clock cycles
- Recognize in 36 clock cycle

Building a decision space

Pattern recognition starts with the definition of a decision space suitable to discriminate different categories of objects. A decision space can be represented by a graph with N dimensions where N is the number of attributes or measurements considered to represent the objects. The N attributes compose a feature vector or signature which can be plotted in the graph. After a significant number of samples has been entered, the decision space starts taking shape and hopefully revealing clusters of objects corresponding to different categories. If this is not the case, the decision space might not turn very useful, and other features should be considered to build a space in which categories are noticeable.

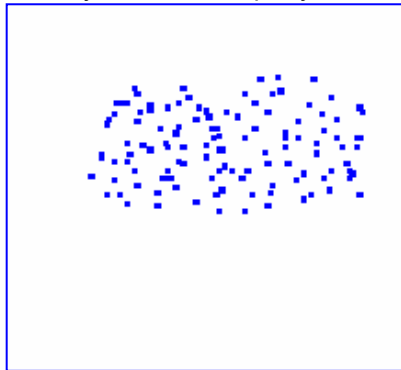
REPRESENTATION OF A DECISION SPACE

If a population of objects is characterized by two measurements (X,Y) , their distribution can be plotted and visualized in a two dimensional graph as seen below. The characterization of population with N measurements instead of 2 just becomes an extension of the illustrations below.

Given a set of (X,Y) collected over many samples, the spatial distribution of these X and Y data can outline significant clusters of points. These clusters can be associated to categories or classes of objects.

Case study 1:

X = Weight
 H = Days of vacation per year

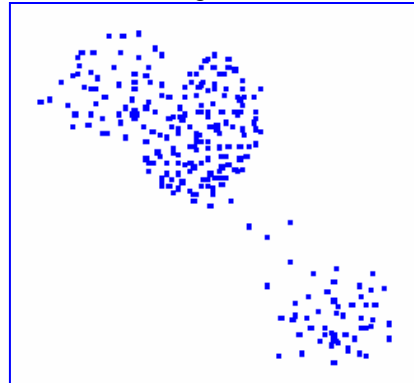


Observations:

No specific cluster can be outlined. The measurements X and Y are either irrelevant or insufficient to reveal categories of persons within the surveyed population.

Case study 2:

U = Shoe size
 V = Annual budget for cosmetics



Observations:

Two obvious and disjoint clusters appear: (1) persons with shoe size above 10 and spending less than \$800/yr in cosmetics; (2) persons with shoe size under 10 and spending over \$1000/yr in cosmetics. Among the persons with smaller feet, two adjacent categories can be outlined.

"The data depicted in the graphs above is fictitious. Any similarity to any person is merely coincidental."

Conclusion:

If the intend of the survey is to discriminate males and females, the measurements (U,V) will do a better job than the measurements (X,Y) . Other measurements might perform even better.

A DECISION SPACE IS A FEATURE SPACE

The selection and availability of the measurements used to model the decision space are critical since they can render a classification obvious, uncertain or undetermined.

In neural network technology, the set of measurements (X,Y) is called the feature vector or signature vector. It can have a dimension n if it is composed of n values (X_1, X_2, \dots, X_n) . The X_i measurements can be unrelated, or they can be linked and represent a progression or variation in space, time, color or another dimension.

Example 1: Vector of unrelated measurements with different dimensions

For the monitoring of prospective customers, a neural network can be trained to classify people as high, medium, low and unlikely buyers. The decision space can be modeled by assembling the following values into a feature vector:

X1= age

X2= gender

X3= ethnicity

X4= average income

X5= single or married

X6-X8= number of children

etc

Example 2: Vector representing a spatial arrangement

For the recognition of fingerprint, the vector is an assembly of minutia points related in the spatial domain.

Example 3: Vector representing a time progression

For the monitoring of an EKG, the vector is a repetitive time series of a heart beat amplitude.

CONSTRUCTION OF A DECISION SPACE

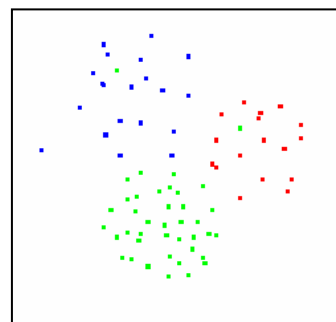
Neural networks are trainable and good at modeling decision spaces with some fuzzy logic. They can generalize what they are taught and therefore recognize patterns which they have never seen before.

The decision space is mapped by teaching a series of examples and labeling them with a category.

Going back to the illustration with a 2D graph, the example vectors (X,Y) are plotted and their color indicates their category C .

Each time a set (X,Y, C) is presented to the network, the neurons first verify if one of them does not already recognize (X,Y) as belonging to category C . If this is the case, the network will not create a new neuron to store (X,Y,C) .

If, on the contrary, (X,Y,C) is not recognized by any neuron, a new one is created to store (X,Y) as a reference pattern and C as the category. In addition the new neuron inherits an influence field which defines its area of attraction or similarity domain.

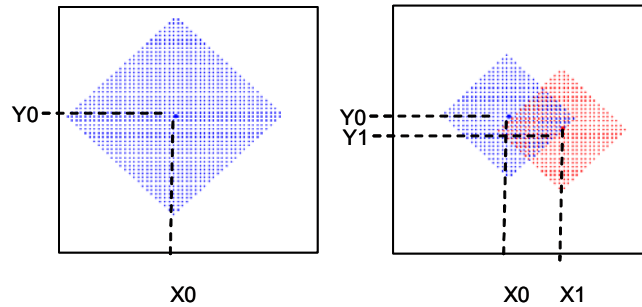


Graph 1

A neuron is represented in the decision space by the point (X,Y) surrounded by a dotted area with a color associated to C . The influence field determines the radius of the area.

If a next example $(X1, Y1)$ falls in the area of the neuron $(X0, Y0)$, it is considered as belonging to category $C0$.

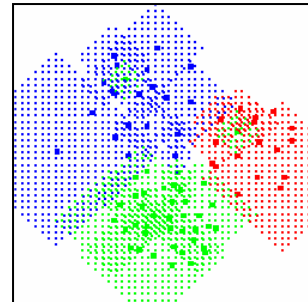
If a teacher instructs the neural network that $(X1, Y1)$ belongs to a category $C1$, the neuron storing $(X0, Y0, C0)$ learns its lesson and shrinks its influence field such that it no longer recognizes $(X1, Y1)$ as $C0$.



Graph 2

As examples are taught, the decision space gets modeled by more smaller neurons. It may feature neurons with portions of their influence fields overlapping with one another. Their intersection represents a zone of uncertainty. The [Norm](#) in use determines the shape of the influence fields and therefore to shape of the decision space.

The 77 examples plotted in Graph1 define a decision space which can be modeled by 21 neurons represented in Graph 3.



Graph 3

USING THE DECISION SPACE FOR RECOGNITION

The classification of a vector consists of evaluating if it lies within the influence field of one or more neurons modeling the decision space.

When a vector is broadcasted to the neural network, all the neurons calculate their distance between the input vector and the prototype stored in their memory. If the distance of a neuron is greater than its influence field, the neuron excludes itself from the list of neurons recognizing the vector. Otherwise it “fires” to indicate that it recognizes “somewhat” the vector. The similarity range is expressed with the distance value. Its dimension is function of the type of data stored in the vector and the norm in use to calculate the distance.

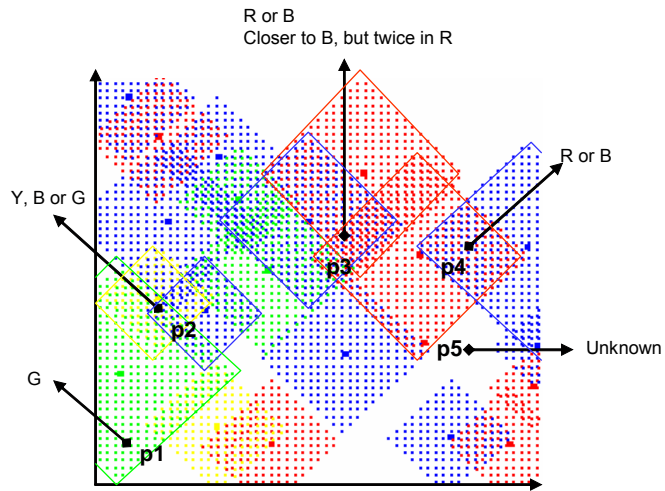
Several neurons can recognize the input vector. The one with the smallest distance value has a prototype in memory which is the closest to the input vector. Also, more than one neuron can fire with the same smallest distance. If they have identical categories, it reinforces the confidence level of the recognition. If they do not have the same category, they point a level of uncertainty in the recognition and potential ambiguities between certain categories. This uncertainty can be further considered by reading the categories recognized by the next firing neurons, that is with the next smallest distance value, etc.

The higher the distance, the less similarity between the prototype and the input vector.

If a neuron has a distance equal to 0, it means that the input vector matches exactly its prototype.

In the example to the right, a 2D decision space is divided into four categories labeled with the colors R, G, B, and Y. It has been modeled with 19 neurons: 6 associated to the R category, 3 to the G category, 8 to the B category and 2 to the Y category. The influence fields of the models have variable size and some are overlapping one another.

The classification of the points P1 to P5 is interpreted depending on their positions inside neurons. P1 is identified without uncertainty as G. P2, P3 and P4 are identified with uncertainty. P5 is not recognized.



Classification status

The recognition of a vector can have three outcomes or classification status:

Unknown: no neuron recognizes the input vector (ex p5)

Identified: one or several neurons recognize the vector and they agree with its category value (ex: p1)

Uncertain: several neurons recognize the vector and they are not all in agreement with its category value. (ex: p2, p3, p4)

Exact-match classification

If a neuron fires with a distance 0, it means that the vector matches exactly the prototype stored in the neuron. The classification is an exact match.

Best-match classification

The response of the firing neuron with the smallest distance value is equivalent to a best match. It can still be uncertain and in this case a detailed classification can be read out.

Detailed classification

Examining the distance and category of all the firing neurons can be of interest to reinforce the accuracy of a decision, especially in the cases of uncertainty. Rules have to be established on a “per application” basis depending on the cost of a mistake, the requirements for a minimum throughput, minimum false negative, etc.

Let’s take the example of a recognition where a vector is recognized by the firing neurons:

Distance	5	12	13	15	38	39
Category	8	7	7	7	3	5

The best match is a reference pattern of category 8 recognized with a distance 5. However, category 7 is recognized next within a close distance by three firing neurons.

If the cost of an inaccurate recognition is low, the response of the 1st neuron with category 8 is the simplest to retrieve (and very fast). On the contrary, if the application cannot afford a false-positive, it might be wiser to involve some statistics and assume that category 7 is the dominant category and should be the one selected for a final decision. More sophisticated rules can be deployed including the analysis of the histogram of the categories, and more. Some applications might even consider the generation of a “response” vector composed of all the “firing” categories (i.e. 8,7,7,7,3,5) and to be classified by another

set of neurons taught to classify the “response” vectors. CogniMem can handle up to 127 subsets of neurons trained for different purposes. These subsets are called [contexts](#).

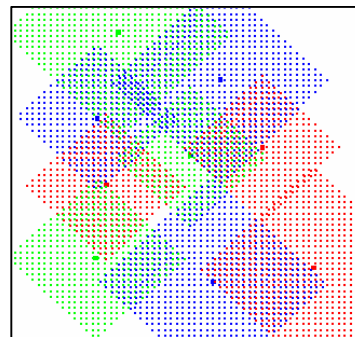
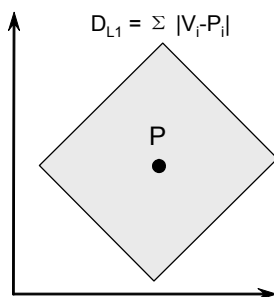
SHAPING THE DECISION SPACE WITH A NORM

The Norm determines how to calculate the distance between the reference pattern or prototype stored in a neuron (P) and an input vector (V). If the calculated distance is less than the influence field of the neuron, the vector V is considered as similar to the prototype P.

Norm L1 (Manhattan distance)

Distance= Sum $|V_i - P_i|$

It emphasizes the drift of the sum of the all components between V and P

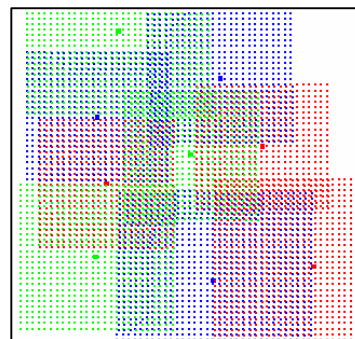
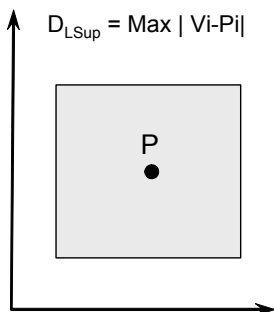


In a two dimensional space, the points V such that their distance D_{L1} to a reference point P is less than or equal to C describe the area of a diamond shape centered in P and with a side equal to $\sqrt{2} \times C$.

Norm L Sup

Distance = Max. $|V_i - P_i|$

It emphasizes the largest drift of the same component between V and P.



In a two dimensional space, the points V such that their distance D_{Lsup} to a reference point P is less than or equal to a constant C describe the area of a square centered in P and with a side equal to $2 \times C$.

The Norm shapes the shape of the neurons’ influence field and therefore the shape of the decision space.

Remark about Norm L2 (Euclidian distance)

The L2 Norm calculating Distance= $\sqrt{\text{Sum } (V_i - P_i)^2}$ would give circular shapes to the neurons’ influence field. However, the little precision improvement of the L2 distance compared to the L1 distance did not justify adding this processing-intensive calculation in the ASIC which would have significantly impact its size.

Which Norm to use?

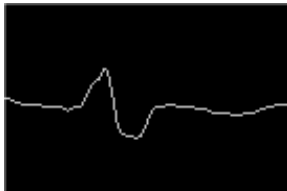
The selection of a Norm depends on the application and in particular the type of patterns to classify, their possible variations between categories and the intend of the recognition such as identification, classification, anomaly detection.

L1 for vector data with unrelated component values

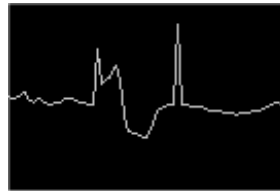
Let's take the example of a data mining application where the profile of customers is categorized based on attributes such as age, sex, weight, skin color, date of graduation, income bracket, etc . These attributes are measured with different units, and some of them are actually codes rather than measurements. Still they can be assembled in a pattern vector to help classify people. In this case, the distance between an input vector and a stored prototype is not representative of any unit, but the L1 Distance gives an idea of the overall variations between them. On the other hand, the Lsup Distance is meaningless since depending on the index of the component with the highest difference, the unit can be years, dollars, codes, etc.

Lsup for noise filtering

A noisy vector features random peaks added to significant signal profile. In the graph below, the vector V_n shown to the right is a noisy version of vector V shown to the left. The L1 distance between V and V_n is 4900 when the Lsup distance is equal to 50. indeed the L1 distance increases dramatically when there is noise in a pattern. Neurons trained with an L1 distance will not easily associate V_n to V unless the engine is very moderate. Neurons trained with the Lsup distance will make this association more easily.



Normal signal



Noisy signal

Neurons can be used to act as a noise filter if they hold prototypes corresponding to non-noisy patterns and their Norm is set to Lsup. If a noisy signal is recognized by one of these neurons, it could be replaced by the reference signal stored in the neuron.

REFINING THE DECISION SPACE BY ADAPTING THE NEURONS' INFLUENCE FIELDS

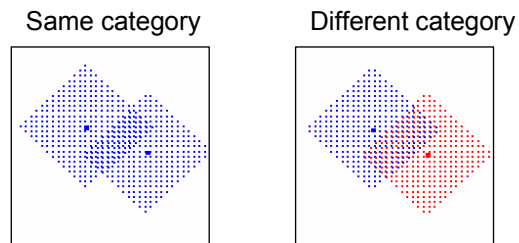
One of the strength of the CogniMem neurons is that they know when and how to adapt their influence field in order to comply with the teacher.

If a neuron containing a prototype of category A recognizes a vector V and V has to be learned as category B, the neuron automatically reduces its influence field to exclude V . All neurons firing with a category different from B do the same and a new neuron is committed to store V as a new prototype with a category B. The influence field of this new neuron is set to the smallest distance of all the firing neurons.

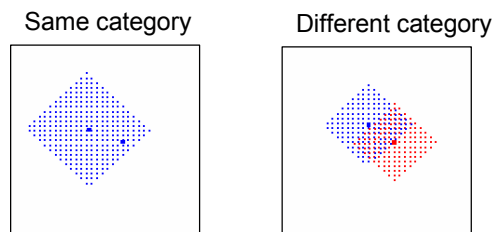
Learning new categories of vectors reduces the influence field of existing neurons and create new smaller neurons. The category 0 has a special functionality in the sense that it reduces existing neurons, but does not create new ones. Learning a category 0 is equivalent to showing a counter-example, or a background example.

The following illustrations show how neurons adjust their influence field as new examples are taught.

If two examples are significantly different, two neurons are committed to store their (X,Y) vector and C category. Their influence fields are set to the default maximum value. They overlap slightly, thus defining a zone of redundancy for the category C_{blue} , or a zone of uncertainty between the categories C_{red} and C_{blue} ,



If two examples are close and with the same category, one neuron is sufficient to ensure their recognition. If they have different categories, two neurons are created, but their influence fields becomes equal to their distance minus 1, so the neurons mutually exclude the prototype of the other one.

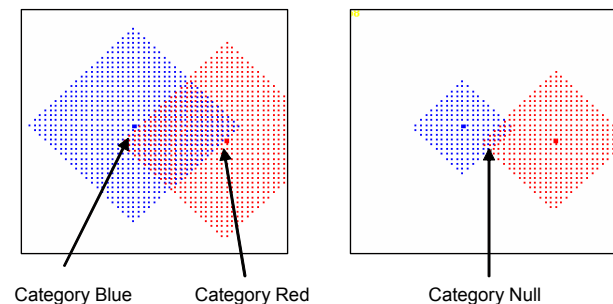


The tendency to create neurons with small or large influence fields can be controlled by teaching counter examples or by changing the Maximum Influence Field global register.

Enforce conservatism with counter examples

The definition of conservatism is to prevent the neurons from over generalizing what they are taught. The easiest way to avoid this behavior is to teach limits or examples which the neurons should not associate to their model. This is the purpose of the category 0 or Null category.

Teaching a category 0 is only intended to force some neurons to reduce their influence field. It does not create any new neuron.



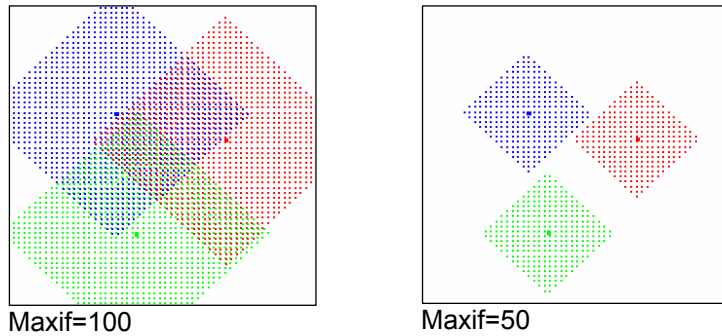
Enforce conservatism with the Maximum Influence Field

The Maximum Influence Field is the default value of a newly committed neuron when no other neuron recognizes the vector to learn

The higher the Maximum Influence Field, the larger the similarity domains of the neurons and the more liberal the recognition engine. The neurons with large influence fields have the tendency to over-generalize and recognize patterns with a high throughput. Liberalism is practical when the cost of an error is not critical and the number of neurons available is limited. If, on the other hand, an application has a very low tolerance to errors, it is best to set a small Maximum Influence Field or teaching many counter-

examples. A conservative engine will require more teaching than a liberal engine because it has to understand the diversity and subtle differences between many examples.

The graphs to the right illustrate how the Maximum Influence Field can help fill more or less the decision space using a same set of examples.



Warning:

Changing the Maximum Influence Field once neurons are already committed can affect the consistency of the knowledge under construction.

Modulate uncertainty areas with the Minimum Influence Field

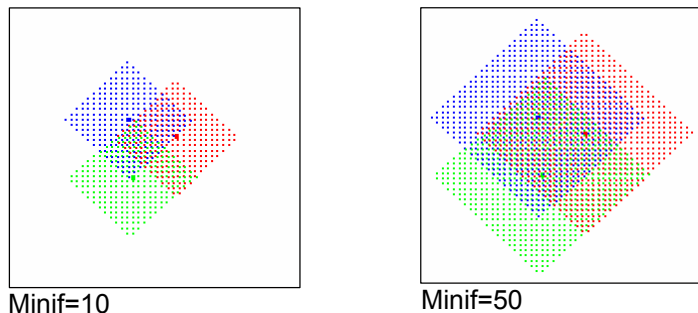
The Minimum Influence Field is the value below which the active influence field of the neurons cannot shrink.

If the influence field of a neuron becomes limited to this minimum value, it means that the prototype stored in the neuron lies close to the boundary of another category, and may be overlapped by another neuron of a different category. The neuron is tagged as degenerated. The higher the Minif, the bigger the extend of “Uncertainty” zones in the decision space.

The graphs to the right show how the Minimum Influence Field can enforce zones of uncertainty.

The three neurons created when Minif=50 are all degenerated. They are not excluded from the influence fields of the other two neurons.

A Minimum Influence Field of 1 minimizes the zones of uncertainty.



Benefit of uncertainty zones

Entertaining an uncertainty domain between critical categories can be a benefit for the implementation of certain applications. For example, a recognition engine can recognize 90% of a population with a high accuracy and a minimal training, but perform poorly on the remaining 10%. Obviously the engine can be tuned to recognize the 10% but at the risk of creating one neuron per example. An another alternative might be the use of a second engine (or context) trained specifically to discriminate objects falling in this 10% of the population (using a different feature, different norm, set of Minif and Maxif values, or else).

The higher the Minimum Influence Field, the bigger the extend of “Uncertainty” zones in the decision space.

Warning:

Changing the Minimum Influence Field of the network once neurons are already committed can affect the consistency of the knowledge under construction.

ITERATIVE LEARNING

The decision space is modeled as examples are taught and its shape depends on their sequence. This dependency is not desirable to build an accurate knowledge and it is recommended whenever possible to learn the examples repeatedly until the decision space is stable. This condition is established when no new neuron is committed between two iterations. Obviously the ability to execute an iterative learning requires that the examples be archived which is technically possible in most applications.

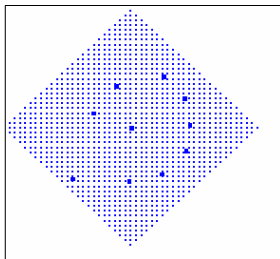
Incremental Decision Space mapping

If an example A is taught as category #1, the CogniMem neural network first identifies if it recognizes the example. If only one category is recognized and it is category #1, the network does not take any action and simply discard the example A. If several categories are recognized, including the category #1, the network does not commit a new neuron to store the example, but instructs the neurons responding with a category other than #1 to shrink their influence fields.

With the learning of additional examples, the decision space will continue to change and it is possible that the example A might no longer fall inside the influence field of a neuron recognizing the category #1. If this is the case, teaching this example again will commit a new neuron with the category #1. This illustrates the fact that the order in which the examples are taught has a direct impact on the modeling of the decision space.

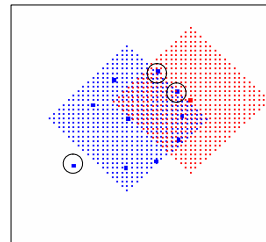
Sequence 1:

A single neuron recognizes the 10 initial examples of the “blue” category.



Sequence 2:

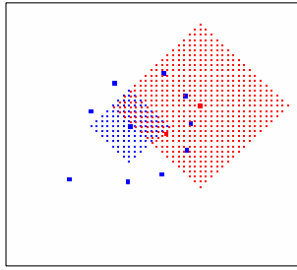
The first example of the “red” category causes the single neuron to shrink. Three of the former blue examples are no longer recognized.



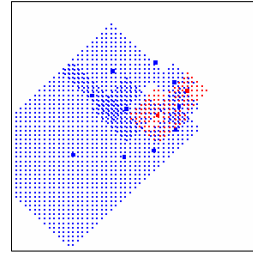
Sequence 3:

The second example of the “red” category causes the first neuron to shrink even more. The 9 original examples are no longer recognized.

Iterative learning:



Re-learning the 9 “excluded examples will this time create 6 “blue” neurons with smaller influence fields.



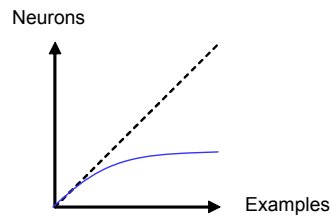
Whenever possible, it is recommended to record the examples so they can be learned in a repetitive loop until the knowledge gets steady and no more neurons are created or shrunken. The decision space is then the most accurate possible since all examples have been taken into consideration.

CONVERGENCE OF THE DECISION SPACE

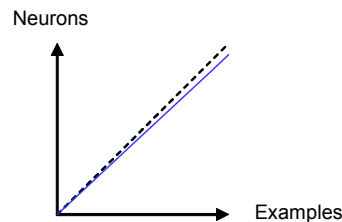
Learning curve and feature relevancy

The learning curve can be represented by the number of neurons created as new examples are taught.

Ideally, if the feature vectors are relevant for the intended classification, the neurons should generalize quickly and properly giving the following profile to the learning curve:



A linear learning curve with a slope close to 1 as shown to the right is a hint that the selected feature vector does not perform well since the neurons do not generalize.

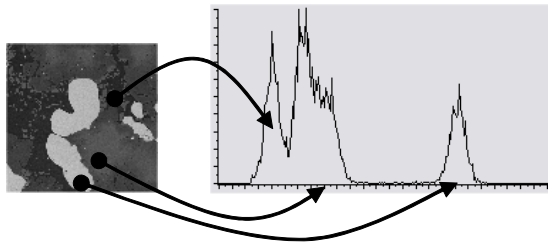


If a feature is not performing as expected, studying the learning curves per categories of examples might be helpful to identify the categories which will be easy or hard to discriminate with the selected feature.

Impact of sample variations

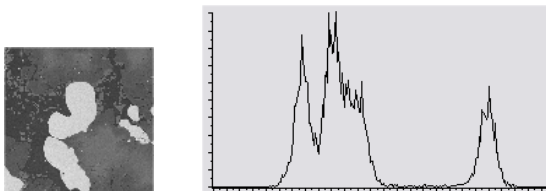
Let's take the example of a classification based on the color of objects and a "histogram" feature vector which represents the number of pixels found in the object for each intensity value.

Sample #1



The histogram #1 is extracted from a material composed of three different components. The bright material is iron and produces the rightmost peak in the histogram. The dark material is copper and produces the leftmost peak. The in-between material is an undefined substrate and produces the second peak.

Sample #2



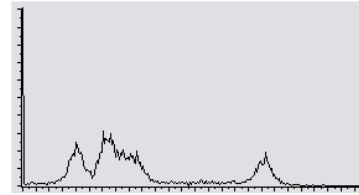
Histogram #2 is extracted from the same sample but displayed under brighter lighting conditions. Its profile is similar to Histogram#1 but slightly shifted to the right towards the higher intensities.

If we assume that Sample#1 was taught as a reference, the neuron holding this prototype might not recognize the Sample#2 because the L1 distance between the two shifted histograms quickly reaches high values. The LSup distance on the other hand is a better norm to classify the two images as similar.

Sample #3

Histogram #3 is extracted from the same sample but acquired with a noisy sensor.

The three peaks seen in Histogram#1 are still present but with a much smaller amplitude. Also a fourth peak to the right appears and corresponds to all the white noisy pixels.



Conclusion

The key to building a robust and accurate knowledge:

- Choose a feature which is efficient to discriminate your objects
- Learn examples which are representative of all the conditions
- Apply iterative learning whenever possible
- Combine the use of multiple feature to reinforce the recognition

RECOGNITION MODEL

The CogniMem neural network can use two different models in recognition:

- Radial Basis Function (RBF)
- K-Nearest Neighbor (KNN)

KNN (K-Nearest Neighbor)

In KNN mode the notion of influence field is discarded and the network always returns a response which is the first closest match.

- Always give a response: Closest match (note that the shortest distance value can still be high)
- The zones of uncertainty are inexistent

Warning: KNN should be turned off during a teaching phase or the knowledge will be limited to one neuron per category.

RBF (Radial Basis Function)

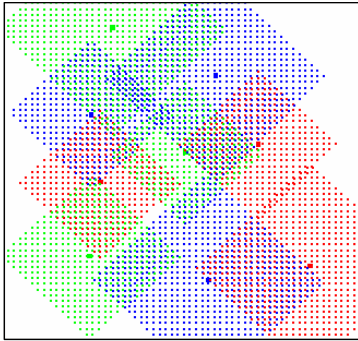
In RBF mode, which is presented in the above graphs, the response can be more refined. Also this classifier is especially suited for anomaly or novelty detection where the “unknown” classification is the one of importance.

- Three possible classification status: Identified, Uncertain, Unknown.
- The mapping of the decision space can vary from conservative to moderate by using different values of the [Maximum Influence Field](#). or by teaching counter-examples.
- The decision space can be tuned by learning both examples and their counter examples
- The zones of uncertainty can be controlled by using different values of the [Minimum Influence Field](#).

The following plots illustrate how a set of identical neurons can model different decision spaces whether they are used in RBF or KNN recognition mode.

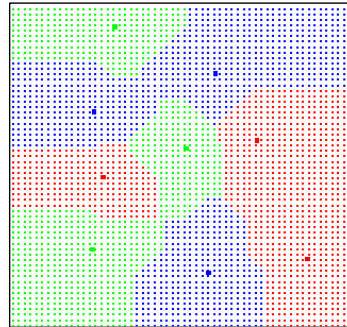
RBF

The space is mapped partially with certain zones being unclassified (i.e. black color). The zones with multiple colors are zones of uncertainty.



KNN

The entire space is mapped and with a single possible category (i.e. color code) per position.



NEURAL NETWORK AS KNOWLEDGE BASE

Once a decision space has been modeled and validated by recognizing many samples, the contents of the neurons can represent a valuable knowledge base and intellectual property.

The contents of each neurons takes 264 bytes, so the knowledge file is limited to 264-bytes times the number of committed neurons. Depending on the application, it might be necessary to also record the settings of the network such as the Minimum and Maximum Influence fields in case additional training is envision to complete or improve the knowledge. Also the knowledge file must be associated to the description of the features used to train the neurons, per context if several context are in use.

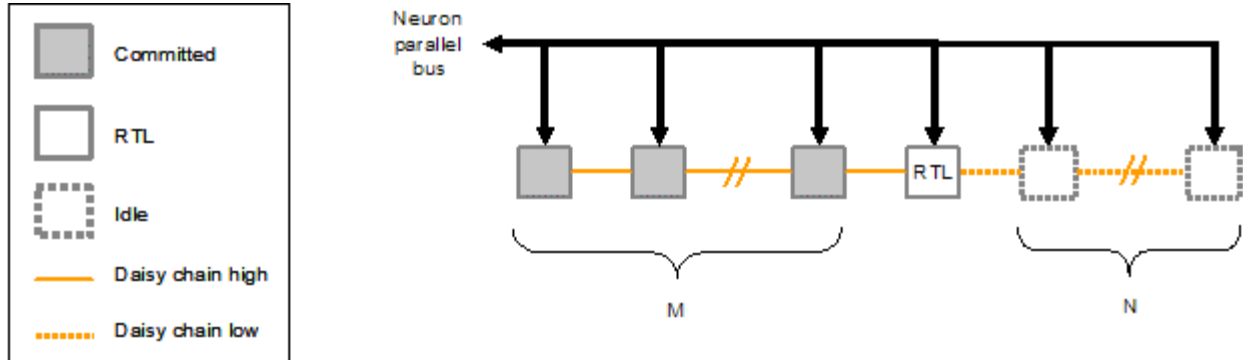
A knowledge file can be built in advance using a system interfaced with a CogniMem silicon network or a simulation of the CogniMem network. The 264 bytes of information necessary per neuron can be loaded at a later time. This transfer should be preceded by a clear of the neurons whenever possible. Otherwise, it becomes a merger between two knowledge bases (the one residing in the neurons and the one to load) and it must be considered cautiously since their neurons could contradict each another. If the two knowledge bases use different contexts then merging them is always safe.

COGNIMEM DECISION SPACE SUMMARY

- RBF or KNN classifier
- L1 or Lsup norm
- Minimum influence field
- Maximum influence field
- Learn examples and counter examples
- Unlimited number of examples can be taught
- Unlimited number of neurons can be committed

Network architecture

The CogniMem neural network is a chain of neurons arranged in parallel. At initialization, the neurons are empty meaning that they do not have any knowledge. Their status is "Idle" except for the first one which is "Ready-To-Learn". As examples are learned by the network, neurons are progressively used to store and build knowledge. They become "Committed".



At any time, the neural network has the following sequential arrangement of neurons per status:

- M committed neurons containing a knowledge (i.e. a context value, a prototype vector, a category value and an influence field value).

- 1 ready-to-learn (RTL) neuron waiting to become committed

- N idle neurons doing nothing, except waiting to become the RTL neuron

$M+1+N$ is the total number of neurons available in the network

M is the size of the knowledge

When a learning operation is executed, the RTL neuron stores the vector and category to learn and waits for the M committed neurons to confirm that the vector and its proposed category bring novelty to the existing knowledge. If this confirmation is not received, nothing occurs. If confirmation is received, the RTL neuron changes status from RTL to Committed and the next Idle neuron in the chain becomes the RTL neuron. The RTL neuron moves to the right as the knowledge increases.

All committed neurons attempt to recognize any incoming vector. Once committed, a neuron can only shrink its Influence Field and set its Degenerated flag. Its status can return to Idle using the Forget and Init commands which clear the category register of all the committed neurons at once.

The Idle neurons execute only the commands which update the global registers: Context, Minimum Influence Field and Maximum Influence Field.

NEURON PARALLEL BUS

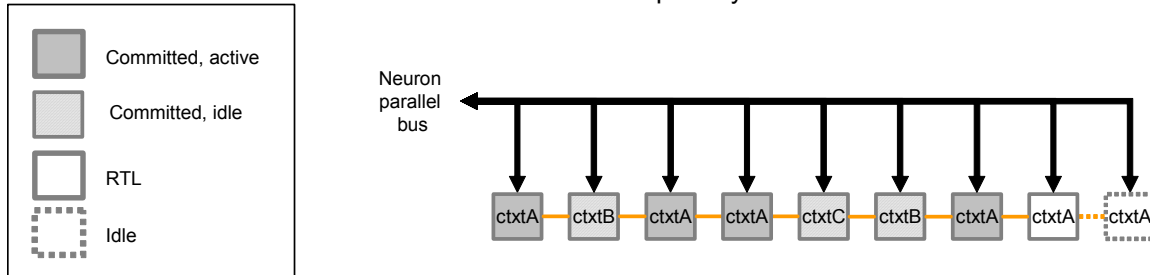
The fully parallel architecture of CogniMem is made possible because all the neurons are identical and do not require any controller or supervisor to interact with one another. They receive the same instructions and data over the neuron parallel bus and execute them at the same time. Note that the execution of certain instructions require that the RTL and Committed neurons consult the response of one another over the neuron parallel bus. This interaction is necessary to build a consistent and adaptive knowledge. Refer to the [introduction to Decision Space](#) for more information.

SEGMENTATION PER CONTEXT

The neurons can be associated to different contexts and their use can be enable/disable per context value.

When a neuron gets committed, its Neuron Context Register (NCR) is set to the value of the Global Context Register (GCR). Whenever the GCR is changed, all the neurons with a different context value will not attempt to recognize any input vector, nor react to the learning of a new vector. They remain idle until the GCR is changed to a value matching their context value. A GCR equal to 0 activates all the neurons regardless of their context value.

The illustration below shows how neurons can turn temporarily idle when the GCR is set to the value A:



The neurons belonging to a given context make a recognition engine. If the network has neurons belonging to N different contexts, it means that it can switch between N different recognition engines prior to learning or recognizing an input pattern. Finally if the Global Context is set to the value zero, all engines will be used in conjunction to recognize the input pattern.

Multiple contexts for multiple parallel decision criteria

Example 1: multiple zones of inspection in a part

If the good quality of a part can be decided based on the proper size, shape and position of N different components in a part, the neural network can be trained to recognize each component separately using one context per component. The final classification of the part is then based on the classification of each component and rules which can be more or less conservative or moderate. For example, a glass bottle can be shipped if its cap is properly twisted, the cap seal is in place, no contaminant is found at the bottom, and the filling level is above the minimum. It can then be diverted to a container for "Quality Assurance A" versus "Quality Assurance B" depending on its filling level.

Example 2: multiple features per object

If a material can be characterized by its colors and texture, two sets of vectors can be extracted from each sample: one describing its color distribution, and one describing its graininess. The classification of the material then relies on two contexts, or two decision spaces.

Multiple contexts for hierarchical decision

Example 3:

A practical approach to recognizing moving targets such as vehicles in an outdoor scene consists of learning relevant subsets of these targets (such as the silhouette of a car, a pick-up, a truck, the front view of a hood, the side view of a hood, a pair of headlights, wheels, etc). These subsets can appear at

different scales and sometimes partially hidden, Therefore their recognition can be made by neurons belonging to N “liberal” and “entry-level” contexts. A 2D map of the subsets recognized by the N contexts can be plotted to produce a “Transform” image which is much simpler (almost binary) and can be recognized as a 2D pattern by a higher level context.

Example 4:

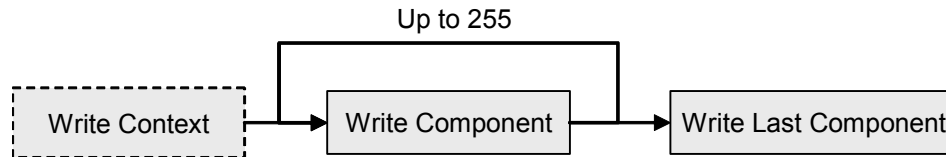
In predictive maintenance, the classification of anomalies can be processed hierarchically starting with the detection of any novelty by a top “conservative” engine (neurons of context#1) to make sure that nothing is discarded. The samples recognized as novelty trigger the use of a second engine trained to classify the data with a greater level of details. Based on its classification results, this engine can trigger other engines and so on until the novelty becomes a classified anomaly.

Network operations

VECTOR BROADCASTING

The memory of the neurons is 256 bytes long so the vectors to learn or recognize can be composed of up to 256 components of 8-bit value.

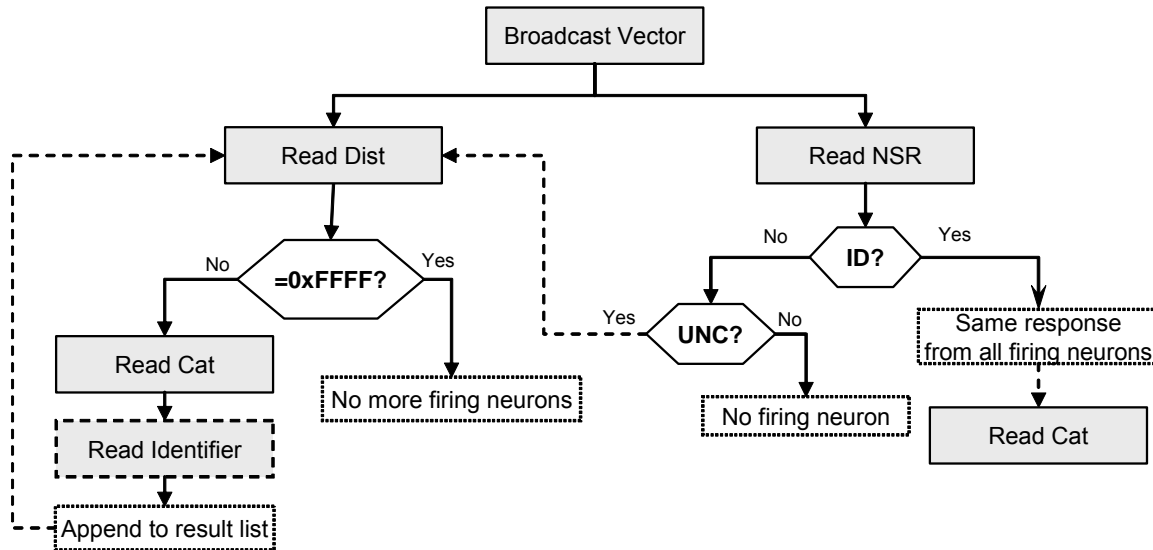
The broadcast of a vector to the neurons is made with the following sequence of operations:



- 1 Write Context
 - o Optional
 - o Needed if the new vector must be associated to a [context](#) different than the current value of the Global Context
 - o Needed if the [distance norm](#) coded in bit 7 of the context must be changed
- Up to 255 Write Component
 - o Write all the components of the input vector but the last one
 - o For all the neurons with a context equal to the Global Context, their distance register is updated after each Write Component according to the Norm in use.
- 1 Write Last Component
 - o For all neurons with a context value equal to the Global Context, their distance register is updated and represents the distance between the input vector and the prototype stored in their memory.
 - o Furthermore, if their distance falls in their influence field the neuron “fires” and outputs its category to the neuron bus.
 - o The status of the network global response is immediately known and updated in the Network Status Register. It reports if the input vector is recognized or not, and in the former case if it is recognized with or without uncertainty. This limited response can be sufficient for certain applications, such as anomaly detection, or for some entry-level recognition engine (i.e. context) used to trigger more sophisticated engines.

VECTOR RECOGNITION

A vector can be recognized by one or more neurons. As soon as its last component has been broadcasted to the neurons, the ID (identified) and UNC (uncertain) flags are updated and written in the Network Status register. The response of all the firing neurons, if any, can be read as presented in the following diagram.



The CogniMem network can exercise two types of classifiers: [Radial Basis Function \(RBF\)](#) or [K Nearest Neighbor \(KNN\)](#). The KNN classifier always returns an identification, when the RBF classifier can also report uncertainties and unknown.

RBF Classification

RBF, Unknown classification

If the vector is not recognized by any neuron, ID and UNC are both set to 0. The Read Distance and Read Category commands both return 0xFFFF since no neuron has fired.

RBF, Identified classification

If the vector is recognized by one or more neurons which have the same category, ID is set to 1 and UNC is set to 0. The first Read Category command returns the one and only recognized category. Note that if it is of interest to review the distances of the firing neurons, the execution of consecutive Read Distance commands will retrieve them in increasing order. The first distance quantifies the difference between the vector and the neuron with the closest pattern. The second distance quantifies the difference between the vector and the neuron with the second closest pattern, and so on.

RBF, Uncertain classification

If the vector is recognized by several neurons which have different categories, UNC is set to 1 and ID is set to 0. The distance and category of the firing neurons can be read in sequence per increasing order of distance. The first distance and category quantifies the difference between the vector and the neuron with the closest pattern. The category of this neuron is the classification with the highest confidence level. The second distance and category quantifies the difference between the vector and the neuron with the second closest pattern. The category of this neuron is the classification with the next highest confidence level, and so on.

REMARK1: After a Read Category command, the identifier of the neuron can also be read using the Read Identifier command. This feature can be useful to review the content of the neuron(s) which recognize the vector.

REMARK2: A Write Category command can be executed immediately after a Read Distance + Read Category sequence without having to re-enter the vector. This can be useful for applications where you first want to read the result of a recognition prior to learning such as in predictive maintenance or target tracking.

KNN Classification

KNN, always best match classification

The vector is always recognized since the KNN model discards the relationship between the distance and influence field of a neuron. As a result, all the neurons fire. Their distance and category can be read in sequence per increasing order of distance. The first distance quantifies the difference between the vector and the neuron with the closest pattern. The category of this neuron is the category with the highest confidence level. The second distance quantifies the difference between the vector and the neuron with the second closest pattern. The category of this neuron is the category with the second highest confidence level, and so on.

Type of recognition per applications

Predictive maintenance

In predictive maintenance, it is important to detect when a pattern is not recognized. It can represent an anomaly or an event never seen before which should be added to the knowledge of “normal” event. The Read NSR command is sufficient to deliver this information immediately after the broadcast of a vector in one clock cycle.

Target tracking

In target tracking, the category is most likely the same for all neurons and equal to “Target”. It is more important to monitor the ID flag and the variations of the distances. Their drift away from 0 indicate that the target is changing and should be re-learned quickly before the neurons stop firing. The Read NSR and Read Distance of the top firing neuron might be sufficient to track the target if it belongs to a single category. If it is considered best to verify that multiple neurons recognize the target and within a reasonable distance range, then a Read Category is required in between each Read Distance.

Recognition with low cost of the mistake

In machine vision, the speed and throughput of the recognition can be of greater importance than the accuracy and identification of possible uncertainties in the classification. Reading the distance and category of the neuron with the best match can be retrieved in 35 clock cycles with a sequence of Read Distance + Read Category

Recognition with high cost of mistake

In medical imaging, the response of the neuron with the best match might be too simplistic. The identification of possible uncertainties between neurons must be monitored and rules applied to deal with these uncertainties depending on the categories causing confusion and their respective confidence levels or number of occurrences. Note that all these considerations are raised at the recognition time, but that the engine can already be very conservative due to the way it was trained. The response of the top N neurons in the firing list is retrieved in N times 35 clock cycles with a series of N Read Distance + Read category.

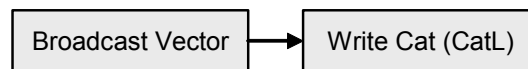
VECTOR LEARNING

Since all the neurons have their internal learning logic, teaching a vector is as simple as writing its category value (CatL).

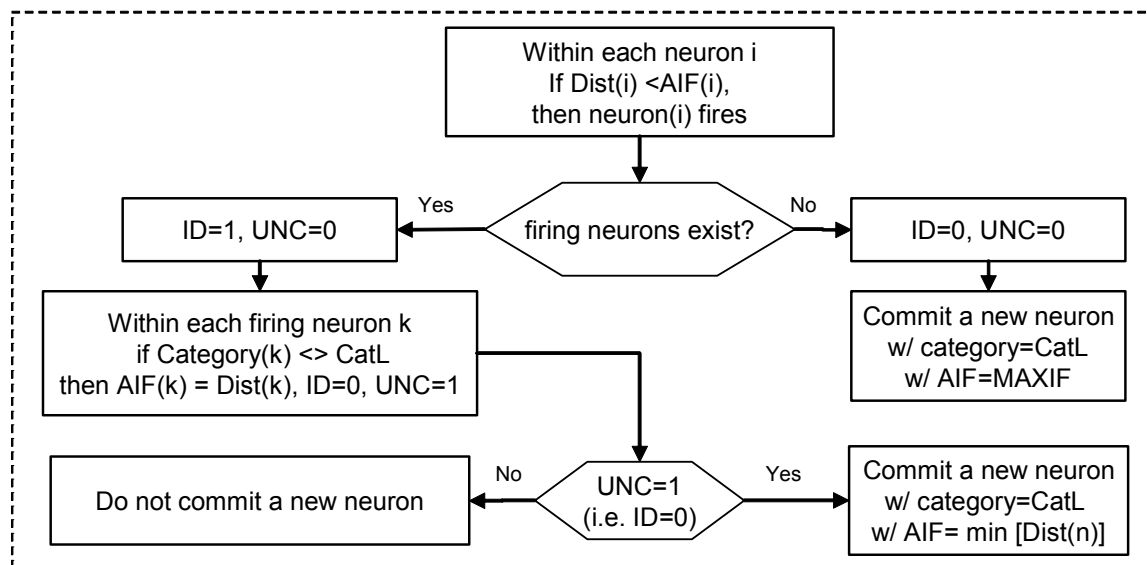
Types of neuron interactions

The committed neurons automatically evaluate if the vector is not already recognized as CatL. If so, they consider that the learning operation does not bring sufficient knowledge to commit a new neuron. On the other hand, the neurons which recognize the vector with a category other than CatL reduce their influence field to exclude this recognition the next time the same vector is broadcasted.

When no committed neuron recognizes the vector as CatL, the RTL neuron writes CatL in its category register and by definition becomes committed. The next neuron in the chain turns from idle to RTL (ready-to-learn). The initial influence field of the new neuron is set to the distance of the firing neuron with the closest pattern or the [Minimum Influence Field](#) whichever is greater. If they are no firing neurons at all, the initial influence field is set to the current value of the [Maximum influence field](#).



Within the automatic model generator



Remark: If the network is full, a learning operation will have no effect. You can detect that all the neurons of the network are already committed by executing the Read NCOUNT command which will then return the value 0xFFFF.

Global settings prior to learning

At the beginning of a learning session, the following parameters can be written:

Write [Context](#) (to segment the network)

Write [Maxif](#) (to adjust conservatism)

Write [Minif](#) (to control uncertainty domain)

Waiving dependency from the teaching sequence

The knowledge depends on the order in which the examples are learned. To circumvent this dependency [iterative learning](#) is recommended whenever possible.

SAVE AND RESTORE MODE (SR)

The Save and Restore mode allows to save the knowledge residing in the committed neurons and reload it at a later time. It disables the use of the neurons to learn and recognize vectors, but enables the read/write of the internal neuron registers while moving sequentially through the chain of neurons.

Activation of the SR mode

The SR mode is activated by setting the bit 4 of the Network Status Register to 1. The RTL neuron becomes the “indexed” neuron of the chain

Sequential access to the neurons

The registers of the “indexed” neuron can be read and written. The Category register must be the last register to read or write since either command will shift the “index” to the next neuron of the chain.

Reset chain

The Reset Chain command allows to point directly to the first neuron in the chain.

Direct access to a neuron

To access a neuron #i in the chain, a Reset Chain must be executed followed by (i-1) commands Read category.

De-activation of the SR mode

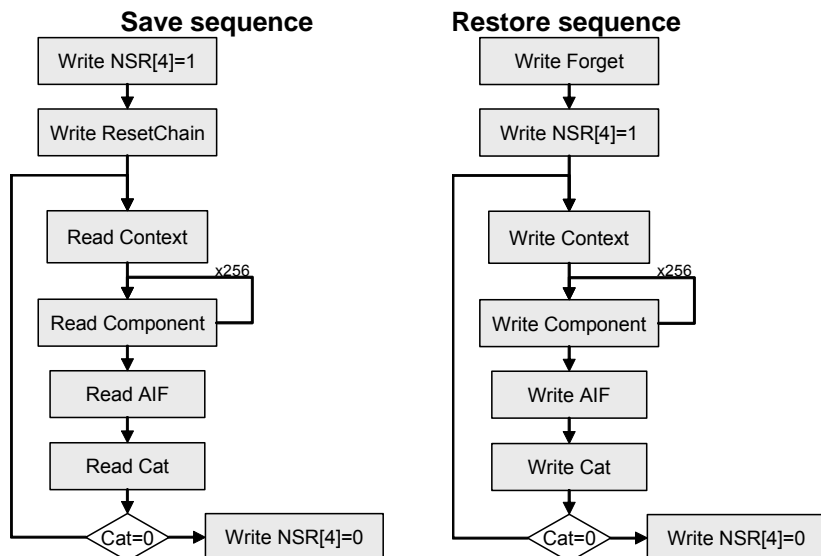
When the SR mode is cancelled by setting the bit 4 of the Network Status register to 0. The committed neurons with a category different from 0 are ready to learn and recognize again. The first neuron in the chain with a null category value automatically becomes the RTL (Ready-To-Learn) neuron.

Example of Save and Restore sequences

The diagrams to the right show typical sequences to save or restore the contents of the committed neurons.

The Write Forget in the Restore sequence is optional, but please note that appending neurons to an existing knowledge in SR mode can generate an ambiguous decision space since the neurons do not interact and adjust their influence fields.

The Category register must be the last register to be read or written since either of these two actions increments the neuron pointer. The order in which the other neuron registers are read is not important.



The neuron entity

A neuron is a cognitive and reactive memory which can autonomously evaluate the distance between an incoming pattern and a reference pattern stored in its memory. If this distance falls within the influence field of the neuron, the latter returns a positive classification which consists of the distance value it calculated and the category of its reference pattern. Although a neuron has its own processing unit to react to a pattern, it is the collective response of all the neurons which produces an interesting diagnostics. When attempting to recognize a pattern, each neuron has the capability to spy the response of its counter-parts and to withdraw itself from the race if another neuron reports a smaller distance value.

NEURON STATUS

A neuron consists essentially of a memory storing the learned pattern prototype (or kernel), a hardwired distance evaluation unit, a learning logic and an associative logic. Depending upon its content the neuron has [three different states](#):

- Idle (does not participate)
- Ready to learn (next in line to learn a pattern)
- Committed (holds a pattern and its associated category. has an adjustable influence field)

NEURON COMPONENTS

Neuron memory

The neuron memory has a capacity of 256 bytes. This means that it can store a pattern or prototype of 256 8-bit values. All the neurons point to the same memory cell index at any time.

The access to the memory cells is controlled by the neural network itself. The index is automatically incremented each time a new component is broadcasted to the neurons. It is reset to 0 after writing the last vector component of the vector.

The following operations reset the index pointing to the neuron memory cells:

(1) In Learning and Recognition mode:

- Write Last Component
- Write IndexComponent with value 0
- Write Network Status Register (NSR)

(2) In Save and Restore mode

- Read or Write to the category register
- Reset the chain of neurons to point on the first neuron

Distance evaluation unit

The distance evaluation unit computes the L1 or Lsup distance between the incoming vector (up to 256 components) and the stored prototype. This occurs in parallel for all the committed neurons, at each Write Component and Write Last Component commands.

The Distance value is automatically reset to 0 by writing to the Network Status Register (to change the operation mode or the classifier in use) or by writing the first component of a vector.

Associative logic

The associative logic is activated by the Write Last Component command. If the Distance is less than the Influence Field, the neuron fires and output its Category register to the neuron data bus. This output is multiplexed for all the firing neurons and the classification status is immediately known as ID, UNC or Unknown.

Learning logic

The learning logic succeeds to the associative logic. The neuron behaves differently whether it is committed or ready-to-learn. If the neuron is committed and has fired, the execution of the Write Category command can cause the following changes: (1) the Influence Field is reduced to Distance if Category is different from the category to learn. (2) if Distance is less than the Minimum Influence Field, the neuron Influence Field is clipped to this minimum and bit 15 of the category register is set to 1 to mark the neuron has degenerated. This is self-adaptation.

The learning logic of the Ready-To_Learn neuron is slightly different: it adjusts its influence field to the distance to the closest neighbor and changes status to committed. As a consequence the learned vector and category are stored for good in the newly committed neuron and cannot be changed except with a Restore operation.

All these operations occur inside the neuron and are not controlled by any external logic.

NEURON BUS

While each neuron has its own learning and recognition logic, its behavior is influenced by the behavior of the other neurons of the chain. A patented "search and sort" mechanism allows to retain the smallest value transmitted by all neurons at the same time on the neuron data bus. Each neuron can then use this feedback to determine if it can stay in the race of the best match in term of distance value or category value.

STATIC REGISTERS

The neuron static registers are written when the neuron gets committed and cannot be changed afterwards, unless they are edited under the SR mode.

Neuron Context Register (NCR)

The neuron context is set to the value of the Global Context register at the time the neuron gets committed. It is coded on bit[0:6]. Bit 7 is the Norm used by the neuron to update its Distance register.

Category (CAT)

Category is the label or class assigned to the vector stored in the neuron memory. It is coded on bit [0:14]. Bit 15 is reserved for the “Degenerated” flag set automatically by the neuron learning logic. The category of a firing neuron can be read after its Distance value.

Identifier (NID)

The neuron identifier is the index of the neuron in the chain. It is automatically updated when the RTL neuron becomes committed, or when the SR mode is turned from ON to OFF. The identifier of a firing neuron can be read after its Category value. This value is coded on 24-bit and will not increase after the neuron number 16,777,216 or the last neuron in a chain of 16,384 CogniMem chips.

The neuron identifier can be read in SR mode. If you have more than 65535 neurons in a chain and the identifier is coded on more than 16-bit, Read NID retrieves bit [15:0] and Read NCR retrieves bit [23:16] in the unused upper byte of the neuron context register.

DYNAMIC REGISTERS

The dynamic registers are updated internally by the associative logic and learning logic of the neurons. They can be overwritten in SR mode.

Active Influence Field

The [Actual Influence Field](#) of a neuron defines the amplitude of its similarity domain. If an input vector is similar to the prototype stored in the neuron and their distance is less than the neuron’s Actual Influence Field, the neuron will fire to acknowledge recognition of the input vector. The AIF of a neuron is automatically reduced when a learning operation requires that the neuron shrinks its similarity domain to leave some decision space for a new neuron.

Distance

The neuron [distance](#) is calculated by the neuron when an input vector is broadcasted to the network. It represents the distance between the input vector and the prototype stored in the neuron. A Distance of 0 means that they match exactly. If the Distance is smaller than the neuron’s AIF, the neuron will fire to acknowledge that the input vector falls within its influence field. If the Distance is greater than the neuron’s AIF, the neuron does not fire and output a distance 0xFFFF to the neuron bus to notify that it is not in the race.

The Distance register is actually updated each time a component is written with Write Comp or Write Last Comp commands. It is reset to 0 when a new first component is written or when the network is set to SR mode.

Degenerated flag

A committed neuron is declared [‘degenerated’](#) when its influence field becomes equal to the Minimum Influence Field. This means that its prototype lies close to the boundary of its category space, and may be overlapped by another category space.

CogniMem Register Summary

This table is a summary of the neuron registers, their data type and their read/write access depending if the network is in LR (Learning/Recognition) or SR (Save/Restore) mode.

All neurons receive the Read/Write Register commands in parallel and execute them depending on their status as idle, ready-to-learn or committed. The timing of each command are supplied in the CogniMem chip datasheet.

NEURON REGISTERS

	Description	Type	LR mode	SR mode	Addr	Default
COMP	Component Writes to the neuron memory at the current index, updates the distance register and increments the index.	8-bit	W	RW	0x01	0x00
LCOMP	Last Component Writes to the neuron memory at the current index, updates the distance register and launch the neuron associative logic. The fire, ID, UNC flags and NSR registers are updated.	8-bit	W	n/a	0x02	0x00
INDEXCOMP	Component index Set the neuron memory index to an input value which can range between 0 and 255.	8-bit	W	W	0x03	0x00
DIST	Distance register	16-bit	R	R	0x03	0xFFFF
CAT	Category register This value can range from 0 to 32768 (15 bit) Bit 15 is reserved to flag if the neuron is degenerated.	16-bit	RW	RW	0x04	0xFFFF
AIF	Active Influence Field	16-bit		RW	0x05	0x4000
NCR	Bit [6:0]= Neuron Context Register Bit[7]= neuron Norm , 0 for L1, 1 for Lsup Bit [15:8]= neuron identifier[23:16]	16-bit		RW	0x00	0x0000
NID	Neuron Identifier	16-bit	R	R	0x0A	0x000000
TESTCOMP	Write the pointed component of all neurons with the input value	8-bit	NA	W	0x08	0x00

GLOBAL REGISTERS

	Description	Format	LR mode	SR mode	Addr	Default
NSR	Network Status Register This register is updated after each Write Last Comp command	16-Bit	R/W	W	0x0D	0x00
Bit2	Uncertain		R			0
Bit3	Identified		R			0
Bit4	Mode, 0=LR, 1=SR		W			0
Bit5	Model , 0=RBF, 1=KNN		W			0
GCR	Bit [6:0]= Global Context Register Bit[7]= Norm , 0 for L1, 1 for Lsup	16-bit	RW*		0x0B	0x0001
MINIF	Minimum Influence Field	16-bit	RW*	R	0x06	0x0002
MAXIF	Maximum Influence Field	16-bit	RW*		0x07	0x4000
FORGET	Clear the neuron category (status become idle)		W	R	0x0F	
NCOUNT	Number of committed neurons		R*	R	0x0F	
RESET CHAIN	Points to the first neuron of the chain			W	0x0C	

*Register is equal to 0xFFFF if the network is Full, that is all neurons are committed.

Knowledge File Management

Knowledge expandability

Consistent vector length

Bibliography

Darpa Neural Network Study October 1987-February 1988
by DARPA Neural Network Study (U.S.) (Author)

How We Learn; How We Remember: Toward an Understanding of Brain and Neural Systems : Selected
Papers of Leon N. Cooper (World Scientific Series in 20th Century Physics, Vol 10) by Leon N. Cooper

Practical Approach to Pattern Classification by Bruce Batchelor (Nov 1, 1974)

Index

Best-match	See Classification	Maximum.....	11
Category		Minimum.....	12
Background	10	Learning	
Best match.....	8	Influence fields	12
Counter-example	11	Iterative.....	13
Dominant	8	Mechanism	5
Null.....	10, 11	Liberalism.....	11
Classification		Maximum Influence Field.....	11, 17
Best match.....	8	Minimum Influence Field.....	12, 17
Detailed.....	8	Mode	
Exact match.....	8	Learn and Recognition (LR).....	25, 31
Status.....	8	Save and Restore (SR).....	27, 31
Classifier		Network	
K-Nearest Neighbor.....	17	Full.....	25
Radial Basis Function.....	17	Network Full.....	32
Confidence level.....	7	Neuron	
Counter-example.....	11	Committed	19
Decision space		Identifier.....	30
Conservatism.....	11	Idle.....	19
Distance Norm.....	9	Ready-To-Learn	19
Learning mechanism	5	RTL.....	19
Recognition mechanism	7	Norm	7, 9
Uncertainty areas	12	L Sup	9
Degenerated	12	L1	9
Distance	7	Recognition	
Exact-match	See Classification	Classification status	See Classification
Feature vector	5	Mechanism	7
Firing neuron	7	Signature vector.....	5
Influence Field		Uncertainty.....	7
Active	10		