NeuroMem Technology Reference Guide

Version 5.2

Revised 02/04/2019

NeuroMem is a technology of General Vision, Inc. (www.general-vision.com)

1 TABLE OF CONTENTS

2 Introduction 6

2.1 NeuroMem Key Features 6

3 What is a neuron? 8

3.1 Neuron part 1: A memory holding a pattern 9

3.2 Neuron part 2: A distance evaluation unit 9

3.3 Neuron Part 3: An associative recognition logic 10

3.3.1 Firing stage 10

3.3.2 RBF or KNN behavior 11

3.3.3 Identified or Uncertain recognition 11

3.3.4 Unknown recognition 11

3.3.5 Winner-Takes-All 11

3.4 Neuron Part 4: A learning logic 12

3.4.1 Commitment of a new neuron 12

3.4.2 Reduction of the influence field of firing neurons 12

3.4.3 Learning the “0” or null category 12

3.4.4 Degeneration of a firing neuron 13

3.4.5 What happens when the network is full? 13

3.5 Neuron Part 5: An element in an infinite chain 14

4 Network architecture 15

4.1 A chain of identical neurons 15

4.2 Save and Restore of the neurons 17

4.3 Network full 18

5 Managing multiple networks 19

5.1 Use of the context register 20

5.2 Multiple networks for multiple experts 21

5.2.1 Assigning a user-defined context 21

5.2.2 Using Context 0 22

5.2.3 Building inter-experts robust decision 22

6 Operations in Normal mode 24
6.1 Vector broadcasting 24
6.2 Vector Learning 25
6.2.1 Global settings prior to learning 25
6.2.2 Reading the number of committed neurons 26
6.2.3 Building a knowledge independent of the training sequence 26
6.2.4 Controlling the generalization capabilities of the network 27
6.3 Vector Recognition in RBF mode 27
6.3.1 Response Type 1: Conformity detection 28
6.3.2 Response type 2: Best-match 28
6.3.3 Response type 3: Detailed matches 29
6.4 Vector recognition in KNN mode 30
6.5 Practice 32
6.5.1 Example 1 32
6.5.2 Example 2 34
7 Operations in Save_Restore Mode 35
7.1 Save and Restore of the neurons' content 35
7.1.1 Important Remarks 36
7.1.2 Warning about merging knowledge 37
7.2 Reading the contents of a specific neuron 37
8 Knowledge base 39
8.1 Formatting Knowledge files 39
8.2 Merging knowledge Bases 39
8.2.1 Case #1: independent networks 39
8.2.2 Case #2: Related networks 40
9 NeuroMem registers 40
9.1 Neuron behavior per status per instruction 42
9.2 Commands changing the RTL neuron in chain 43

2 INTRODUCTION

The NeuroMem technology is a crucial enabler for cognitive computing and artificial intelligence. It is an architecture of cognitive memories which react to input patterns and can be compared to the brain because of its low power requirements, scalability, and instantaneous internal communications.

A NeuroMem chip is a fully parallel silicon neural network: it is a chain of identical elements (i.e. neurons) which can store and process information simultaneously. They are addressed in parallel and have their own “genetic” material to learn and recall patterns without running a single line of code and without reporting to any supervising unit. In addition, the neurons fully collaborate with each other through a bi-directional and parallel neuron bus which is the key to accuracy, flexibility, and speed performance. Indeed, each neuron incorporates information from all the other neurons into its own learning logic and into its response logic. This mechanism prevents the learning of any redundant information, but also the immediate detection of novelty or potential conflicts. Another resulting achievement of the parallel architecture of NeuroMem is its constant learning and recognition time regardless of the number of connected neurons, as well as the ability to expand the size of the neural network by cascading chips.

2.1 NEUROMEM KEY FEATURES

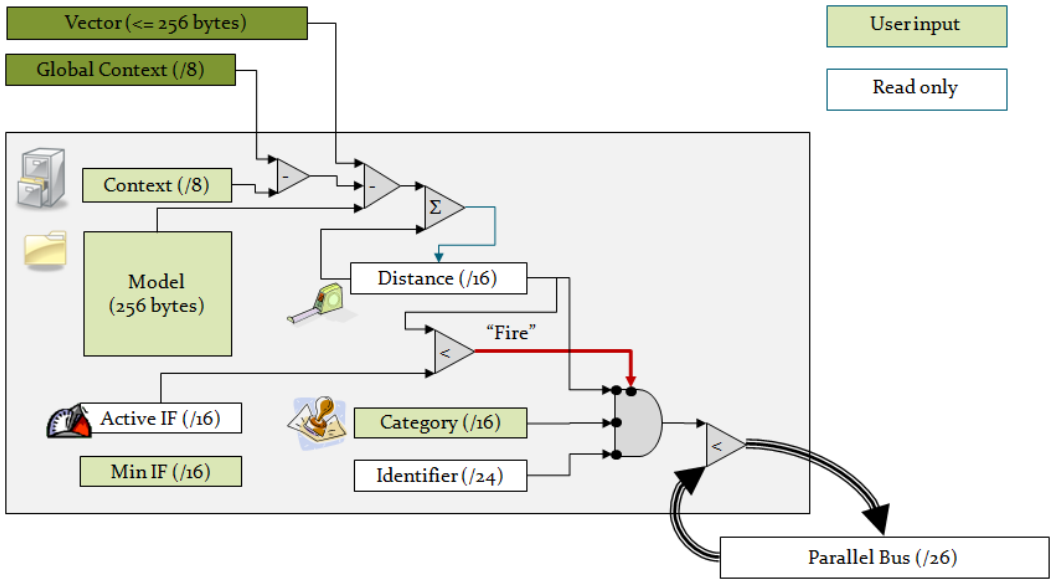
- Parallel broadcast mode
    - All the neurons update their distance value simultaneously as the components of an input vector are broadcasted on their parallel bus. Upon receipt of the last component of the input vector, all neurons have calculated its distance to the reference pattern they hold in memory. If an input vector is broadcasted to a chain of 10, 100 or 1000 NeuroMem chips, their distance values are calculated and ready to be read as soon as the last component of a vector his broadcasted.
- Autonomous model generator
    - The model generator built-in the NeuroMem 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 learning later.
    - The knowledge built by the neurons is cloneable since the content of the neurons can be saved and restored.
- Reactive recognition with Winner-Takes-All

- The neurons are capable of ranking similarities between input vectors and the reference patterns they hold in memory, but also reporting conflicting responses or cases of uncertainty, reporting unknown responses or cases of anomaly or novelty.
- The neurons order their response autonomously per increasing distance value as the host processor sends K successive read commands of the Distance register. Again, this unique feature pertains to the parallel architecture of the neurons and a patented Search and Sort process which allows them to know if other neurons have a smaller distance value without the need for a supervisor or controller.

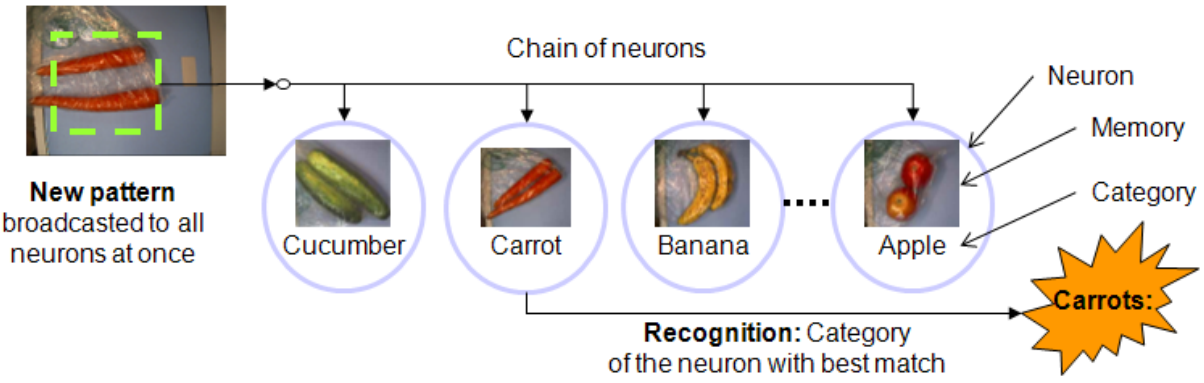
- Fixed latency
  - The time necessary to obtain a response is independent from the number of committed neurons in the network and from their type of response. At each read command, only the neuron with the smallest distance outputs its value to the parallel bus after 19 clock cycles. If an application requires the use of a KNN with K equal to 50 for example, the distance values of the 50^{th} closest neurons are read in 50 * 19 clock cycles.
- Multiple contexts or network dynamic segmentation
  - The ability to assign the neurons to different contexts or sub-network allows building hierarchical or parallel decision trees between sub-networks. This leads to advanced machine learning with uncertainty management and hypothesis generation.
- Multiple type of classifier
  - The neurons can behave as a KNN or RCE (class of RBF)
  - A Restricted Coulomb Energy (RCE) classifier uses Radial Basis Function as activation function. It can represent complex nonlinear mappings and widely used for function approximation, time series prediction, and control.
  - A K-Nearest Neighbor algorithm (*K*NN) is a method for classifying objects based on closest training examples in the feature space. The parallel architecture of the NeuroMem chip makes it the fastest candidate to retrieve the K closest neighbors of a vector among ANY number.

3 WHAT IS A NEURON?

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 a range called the influence field, the neuron returns a positive classification which consists of the distance value 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 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.

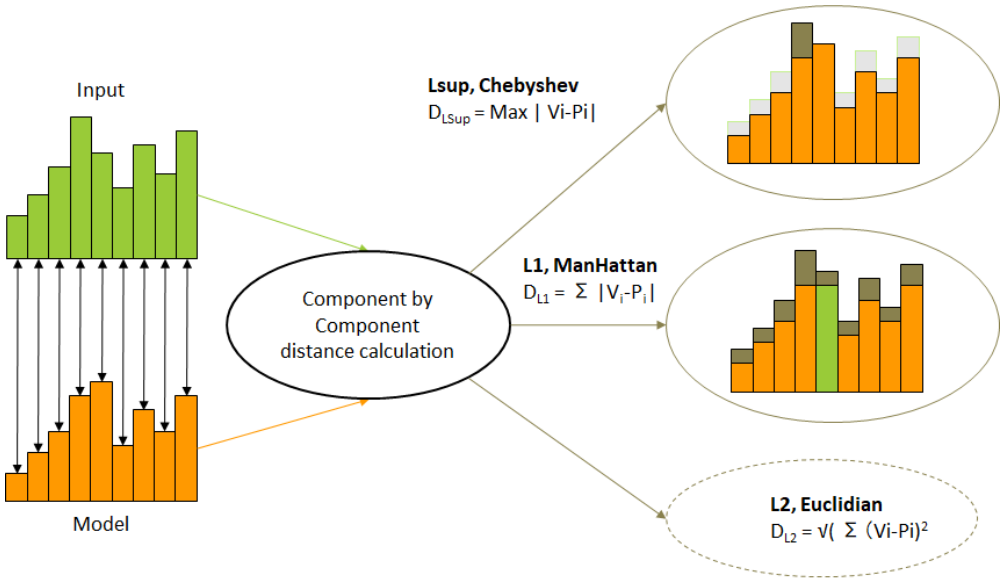


3.1 NEURON PART 1: A MEMORY HOLDING A PATTERN

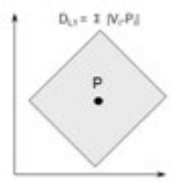
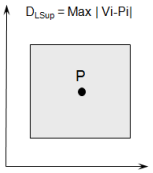
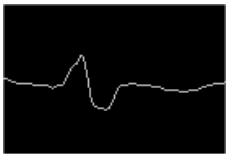
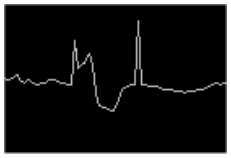
The neuron has a memory with a given capacity (256 bytes in the NeuroMem CM1K chip for example). The access to the memory cells is controlled by the neurons. All the neurons point to the same memory cell index at any time. The cell index is automatically incremented each time a new component is broadcasted to the neurons. It is reset to 0 when the last component is entered. Other subsidiary operations can change or reset the index as described under the chapter “Functional diagrams”.

3.2 NEURON PART 2: A DISTANCE EVALUATION UNIT

The distance evaluation unit computes the distance between the incoming vector and the pattern stored in the neuron memory. This occurs in parallel for all the committed neurons each time a vector component is broadcasted to the neuron bus. The Distance value is automatically reset to 0 by writing the first component of a vector or by writing to the Network Status Register (to change the operation mode or the type of classifier).



The distance can be calculated using two norms: L1 (default) or Manhattan distance, and Lsup. The selection of a Norm depends on the application and the type of patterns to classify, their possible variations between categories and the final intent of the recognition (identification, classification, anomaly detection).

Norm L1 (Manhattan distance)	Norm L Sup
 The L1 distance emphasizes the drift of the sum of the all components between V and P.	 The Lsup distance emphasizes the largest drift of the same component between V and P.
Use model 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 expressed in different units, and some of them are 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.	Use model Neurons can be used as a noise filter if they hold prototypes of non-noisy patterns and their Norm is set to Lsup.   In the above example, V_{noisy} shown to the right is a noisy version of vector V shown to the left. The Lsup distance between V and V_{noisy} is 50 when the L1 distance is 4900. Indeed, the L1 distance increases dramatically when noisy peaks are superimposed onto the signal. Neurons trained with an L1 distance will not easily associate V_{noisy} to V. Neurons trained with the Lsup distance will make this association more easily.

3.3 NEURON PART 3: AN ASSOCIATIVE RECOGNITION LOGIC

3.3.1 FIRING STAGE

The associative logic of the neuron is activated when the last component of the input pattern is received. The calculation of the distance between the input pattern and the pattern stored in the neuron is complete and if its value is less than the neuron's Influence Field, the neuron enters in a "firing" mode and is ready to respond to a query for its distance, category and identifier.

3.3.2 RBF OR KNN BEHAVIOR

The default recognition logic of a neuron is a Radial Basis Function (RBF) as described in the above paragraph "Firing stage". RBF allows to learn examples and build a highly non-linear decision space. RBF also introduces the powerful notion of "unknown" and "uncertainty".

The recognition logic can also be set to a K-Nearest Neighbor (KNN) which is a subset of RBF where the value of the Influence Field is discarded, and the neuron always fires. Under KNN mode, a pattern is always recognized, but the closest firing neuron may hold a distant pattern and report a high distance value.

3.3.3 IDENTIFIED OR UNCERTAIN RECOGNITION

When a neuron fires, it immediately knows if other firing neurons have a different category. If yes, the classification is labeled as Uncertain. If no, the classification is labeled as Identified.

This flagging is executed in a single clock cycle regardless of the number of the firing neurons. This is made possible thanks to the patented parallel architecture of the network interconnecting all the neurons of a NeuroMem network (both intra and inter chips).

3.3.4 UNKNOWN RECOGNITION

If no neuron fires, the input pattern is considered as unknown. This feature is essential for two purposes: (1) to learn more and fine tune the knowledge built by the neurons; (2) to detect novelty and anomaly.

Note that an Unknown recognition cannot occur if the network is set in KNN mode.

3.3.5 WINNER-TAKES-ALL

When a neuron fires, it is ready to respond to queries for its distance, category or identifier but it will first comply to a Winner-Takes-All race, such that it outputs its response on the NeuroMem bus only if it holds the smallest value among all the remaining firing neurons. Smallest value in the case of the distance register means the closest match.

The Winner-Takes-All is executed in a fixed number of clock cycles regardless of the number of the firing neurons. This is made possible thanks to a patented “Search and Sort” mechanism interconnecting all the neurons of a NeuroMem network (both intra and inter chips).

3.4 NEURON PART 4: A LEARNING LOGIC

The learning logic is activated after the associative logic when a category value is assigned to the last input pattern.

3.4.1 COMMITMENT OF A NEW NEURON

Once a new pattern has been broadcasted to the network, only the “firing” neurons or the “ready-to-learn” neuron will react to a learning operation.

If no neuron fires, a new neuron gets automatically committed to hold the input pattern and its associated category. Its influence field is set to the current value of the Maximum Influence Field.

If neurons fire, a new neuron gets committed only if none of the firing neurons identifies the input pattern as belonging to the category to learn. Its influence field is set to the distance of the closest firing neuron.

3.4.2 REDUCTION OF THE INFLUENCE FIELD OF FIRING NEURONS

If the category of a firing neuron is different from the category to learn, it will automatically reduce its Active Influence Field (AIF) to the distance value between its stored pattern and the input pattern. This distance is calculated by the neuron during the broadcast of the input pattern. The reduction of the AIF is a corrective action which will prevent the neuron from firing if the same input pattern is broadcasted to the network again.

3.4.3 LEARNING THE “0” OR NULL CATEGORY

Learning a category equal to the value 0 is a special teaching instruction which cannot commit any new neuron but can force neurons which are firing erroneously to reduce their influence fields and not repeat this misfiring the next time a same pattern is broadcasted to the neurons.

Learning a category 0 is equivalent to teaching a background example or a counter example. The intend of this instruction is to correct neurons which overgeneralize excessively.

3.4.4 DEGENERATION OF A FIRING NEURON

If the Active Influence Field of a neuron must be reduced to a value lesser than the Minimum Influence Field value (MINIF), its AIF is set to the MINIF and the neuron is also flagged as degenerated.

A degenerated neuron behaves as any other committed neuron. If it fires, its distance and category can be read, but bit 15 of the category will be flagged to notify that the neuron was prevented from shrinking during the training phase. This can be an indication that its response should be weighted differently than the response of another firing neuron which is not degenerated.

One interest of the degenerated flag is to obtain statistics on neurons' content at the end of a learning phase:

(1) Degenerated neurons might indicate that the learned patterns contain insufficient information to discriminate their categories. It might also pinpoint errors in the categories assigned to the input patterns.

(2) The significant number of degenerated neurons for a specific category can help establish that an additional feature should be extracted and taught under a different context to classify this category.

Example 1:

In an OCR application, the U and V letters can have very similar signature. It would not be a surprise that the learning of examples of V and U degenerate some neurons because an area of uncertainty between the models of U and V exists (depending on the font of the characters).

Example 2:

A common cause for degenerating neurons is to send contradictory learning instructions. For example, if a knowledge already holds reference patterns of the character V and an operator teaches a new V pattern as an A character by mistake, this erroneous instruction will probably degenerate some of the neurons holding a correct V pattern.

3.4.5 WHAT HAPPENS WHEN THE NETWORK IS FULL?

Once the network is full, a learning command can no longer trigger the commitment of a new neuron, but it can still trigger the reduction of the influence field of committed neurons. This is like teaching the “null” category and modeling a conservative decision space.

3.5 NEURON PART 5: AN ELEMENT IN AN INFINITE CHAIN

A neuron is an element in a chain of neurons. The neurons are daisy-chained to compose a network of neurons, but they all receive and decode commands in parallel as described in the next chapter.

Exceptionally, neurons can be accessed individually when their network is switch from the interactive Learn and Recognize mode to the passive Save and Restore mode.

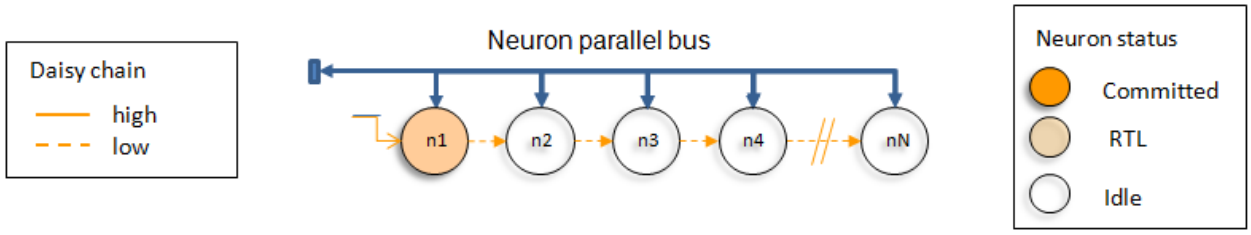
4 NETWORK ARCHITECTURE

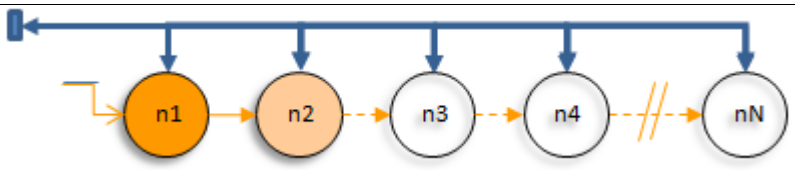
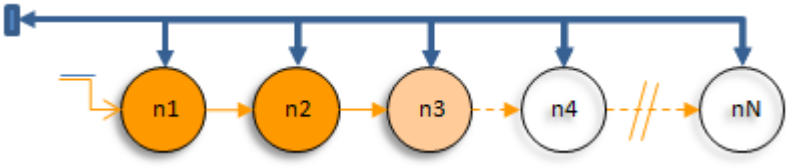
The fully parallel architecture of the NeuroMem chip is made possible because all the neurons are identical and do not require any controller or supervisor to interact with one another. They receive the same instructions and data over the neuron parallel bus and execute them at the same time. The execution of certain instructions requires that the Ready-To-Learn (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.

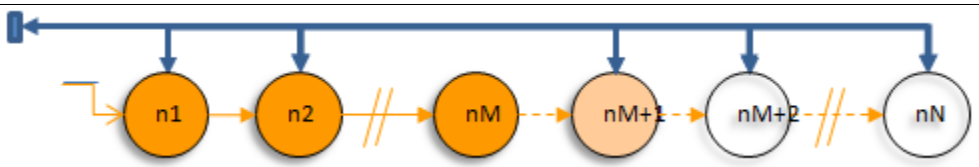
4.1 A CHAIN OF IDENTICAL NEURONS

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 (RTL). As examples are learned by the network, neurons are progressively used to store reference patterns along with their associated category and become Committed.

The state of a neuron in the chain can be Idle, RTL or Committed. It is defined by the status of its daisy-chain-in (DCI) and daisy-chain-out (DCO) lines. The DCO of a neuron rises if its DCI is high and its category register is different from 0. The commitment of neurons is propagated automatically as examples are taught and retained. The RTL neuron also moves along until no idle neuron is available in the chain.



Commitment of 1st neuron	
Commitment of 2nd neuron	

Commitment of Mth neuron	

The neural network is composed of N neurons:

- M committed neurons holding a reference pattern and a category value

- 1 ready-to-learn (RTL) neuron

- N-(M-1) idle neurons

The behavior of a neuron is function of its state in the chain:

Neuron State	**Idle**	**Ready-to-Learn**	**Committed**
Behavior	Does not respond to input patterns, but updates its global registers such as Context, Minimum Influence Field and Maximum Influence Field.	Holds the last input pattern. If a category is taught and not recognized by any of the committed neurons, the RTL neuron stores the category and becomes committed. The next Idle neuron in the chain becomes the RTL neuron.	All committed neurons attempt to recognize an incoming vector. Once committed, a neuron can only shrink its Influence Field and set its Degenerated flag. Its status can return to Idle when it is instructed to forget.
Memory		x	x
Distance calculator		x	x
Associative logic			x
Learning logic		x	x
Save and Restore		x	x

4.2 SAVE AND RESTORE OF THE NEURONS

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

Saving the knowledge consists of saving the contents of the committed neurons which are the neurons holding a pattern with a category different from 0. Another very important information which is part of the neurons' content is their context and influence field registers which are adjusted autonomously by the neurons during learning operations.

A Save and Restore mode controlled through the Network Status Register allows to access the neurons sequentially and read/write their internal neuron registers and memory. The content of each neuron is saved as follows:

- Neuron memory (N bytes)
- Neuron context
- Neuron minimum influence field
- Neuron active influence field
- Neuron category.

The knowledge which is defined as the contents of all the committed neurons times the number of committed neurons.

Remark 1:

A knowledge file can be built in advance using a system interfaced to a NeuroMem chip or a simulation of NeuroMem. The VLEN+8 bytes of information necessary per neuron can be loaded later. 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.

Remark 2:

It is important to associate the knowledge to the description of the feature extraction techniques producing the patterns stored in the neurons as well as the values of Minimum and Maximum Influence fields used during the training. This information can be stored in a header of the knowledge file.

Remark 3:

The content of the neurons must be considered as a whole (a knowledge). Indeed, they auto-adjust their influence fields based on what they already know and thanks to their full interconnectivity. Saving the knowledge, editing or removing a single neuron and restoring the knowledge can be very detrimental! The only possible manipulation which is acceptable and will not deteriorate the knowledge is to separate the neurons by context.

Remark 4:

If under some circumstances, the original data saved from the neurons was edited and restoring it in Save-Restore mode happens to set the category register of a neuron to the value 0, then the activation of the committed neurons will stop at this neuron. Indeed, when the NN is set back to normal mode, the 1st neuron in the chain with category 0 becomes the next “ready-to-learn” and any neuron down the chain is considered “idle”.

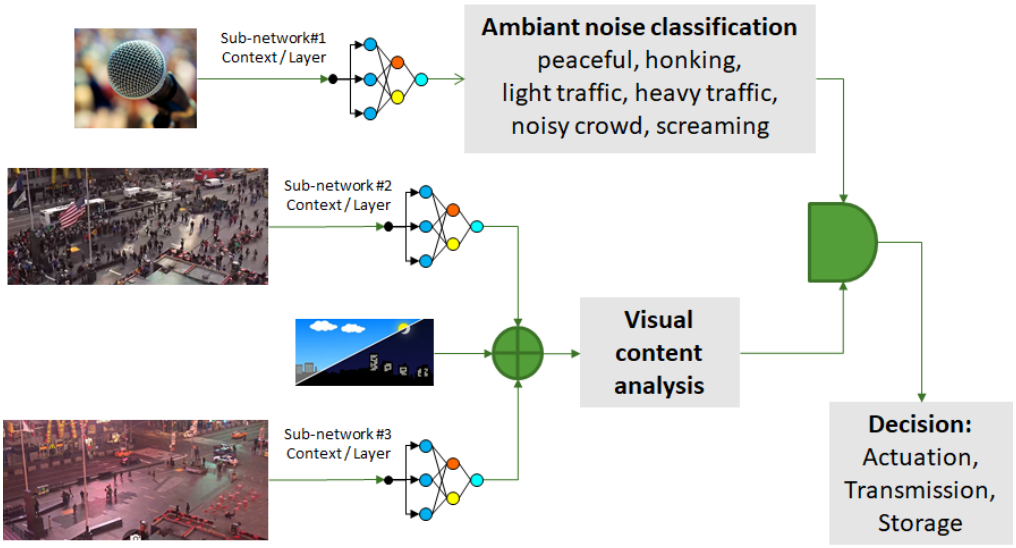
4.3 NETWORK FULL

Once the network is full, a learning command can no longer trigger the commitment of a new neuron, but it can still trigger the reduction of the influence field of committed neurons. This is like teaching the “null” category and modeling a conservative decision space.

Also, reading the values of the registers NCOUNT, GCR, MINIF and MAXIF returns 0xFFFF.

5 MANAGING MULTIPLE NETWORKS

The neurons can be associated to different contexts and their use can be enabled or disabled by selecting a context value. For example, if an application uses two sensors such as a microphone to identify noises or voices and a camera to analyze a video content, two contexts must be defined to toggle between two sub-networks: the neurons trained to recognize feature vectors extracted from audio signal and the ones trained to recognize feature vectors extracted from a video signal. Furthermore, if the application involves outdoor video monitoring, it might be useful to train the neurons differently as a function of time of the day. Indeed, the images will have nothing in common between day and night and might even require the use of different cameras. A context Day and context Night will allow training two sub networks based on the time of the day. In summary, usage of multiple contexts allows segmenting the network per data type. This segmentation can be based on many different factors including but not limited to the sensor model; the sensor settings such as a focal length or frequency range; the type of feature extracted from the data such as a time series or frequency series; the contextual environment of the experiment; the time of collection of the data and more.



5.1 USE OF THE CONTEXT REGISTER

A context is selected by writing a context value to the Global Context Register (GCR) of the chip. Any committed neuron with its own context register different from the global context register turns idle and does not participate in any learning or recognition operation. One exception: the context value 0 enables all the neurons without regards of their context.

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 neurons belonging to a given context define a feature space. If the network has neurons belonging to N different contexts, it means that it contains N different feature spaces. Finally, if the Global Context is set to the value zero, all feature space will be used in conjunction to recognize the input pattern.

There can be many reasons to build a knowledge featuring neurons with multiple contexts:

- Example 1: multiple sensors to monitor a same event (ex: new cameras are now equipped with camera, microphone and possibly gyro)
- Example 2: same audio sensor but multiple sampling rates
- Example 3: same video sensor but different zooming range
- Example 4: same sensor but multiple features extracted from the raw data source (ex: color histogram and histogram of gradient in a patch of pixels)

Since the NeuroMem neurons have their built-in model generator, they (not you) will decide and control when a new neuron must be committed and this for each context. For example, you may very well see cases where learning an example generates the commitment of a new neuron in context 1, but not in context 2. The 1st explanation is that the neurons belonging to context 2 already recognize the feature type 2 with the correct category. The 2nd explanation is that no neuron belonging to context 1 recognizes the feature type 1, or if they recognize it their category does not match the category to learn.

Therefore, the segmentation of the neural network into contexts is made dynamically as new examples are taught. Each context models a different decision space. The only thing the contexts have in common are the categories they model (power-up, normal, over-loaded, power-down). Their dimension and the number of examples retained as novelty by the neurons are all independent.

5.2 MULTIPLE NETWORKS FOR MULTIPLE EXPERTS

A single example (annotated at a given time, at a specific location, etc.) can be used to extract N different feature vectors. These feature vectors represent different dimensions, have different lengths, etc. and basically build N different decision spaces.

Example #1:

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.

Example #2:

An audio signal can be monitored under different frequencies. If we assume that 3 frequencies are of interest, three sets of vectors can be continuously extracted from the same signal but using three frequencies and durations. The classification of the signal then relies on three contexts, or three decision spaces.

5.2.1 ASSIGNING A USER-DEFINED CONTEXT

A context value should be considered as an index to a list of experts.

You can have multiple experts trained on the same images and annotations but using different feature extractions

- Context 1: subsample 32x64 with 2X4 blocks
- Context 2: subsample 32x64 with block 6x32
- Context 3: HistogramRGB

You can have multiple experts trained using a same feature but on different types of images

- Context Zoom 1, Context Zoom 5, Context Zoom 10
- Context Daytime and Context Nighttime

The context is a user-defined index value tying together the nominal size of the ROI, a type of feature, a scale, etc.

The context must be set to the correct value prior to broadcasting a vector. This operation is only necessary if the vector represents a new dimension (aka a different feature, or ROI size, or image scale, etc.)

The context of a neuron is "frozen" at the time it gets committed. This is equivalent to assigning its index to a list of experts in use.

5.2.2 USING CONTEXT 0

The context 0 is a special value in the sense that it activates all the neurons regardless of their individual context value.

The context 0 should not be used during learning, except if your application uses only one expert.

Example #1

- You train 3 experts called Context Zoom 1, Context Zoom 5, Context Zoom 10, and using the same feature extraction. At the time of the recognition, you select Context 0 and build a consensus if at the maximum 2 of the experts agree with the recognized category. For example, you would consider a response as false positive if Context Zoom 1 and Context Zoom 10 agree, but not Context Zoom 5. The fact that one scale is skipped could be suspicious

Example #2

- You train 2 experts Context Daytime and Context Nighttime to recognize a target but activate them both at dawn and dusk.

5.2.3 BUILDING INTER-EXPERTS ROBUST DECISION

The use of multiple networks trained on multiple features allows generating hypotheses and building robust decision schemes. For example, an application with a high cost of mistake may require that at least N sub-networks produce a same classification in order consider the overall response as a positive identification.

5.2.3.1 Example of combinatorial decision

A practical approach to recognizing moving targets such as vehicles in an outdoor scene may consist of learning relevant subsets of these targets such as a silhouette, a hood, headlights, a wheel, a tire, etc. Each subset is associated to a different network trained to deal with changes of scale and orientation. When analyzing the content of a new image, each network reports a map of the locations where it recognizes a pattern. The combination of these maps produces a “Transform” image which is much simpler than the original image and can be classified by a higher-level context trained to verify the spatial distribution of the different subsets to produce

a final decision. For example, it recognizes a car if it has a silhouette of type “Front View” which contains a hood, 2 headlights and 2 front views of a tire. It also recognizes a car if it has a silhouette of type “Side View” which contains 2 side views of a tire.

5.2.3.2 Example of hierarchical decision

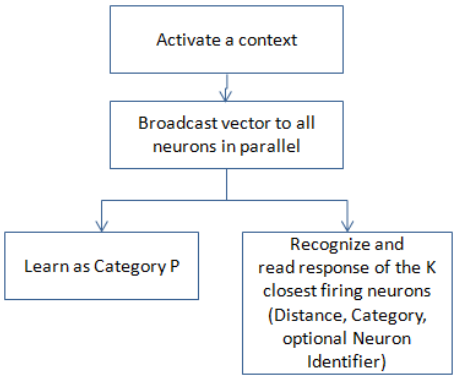
In predictive maintenance, the classification of anomalies can be processed hierarchically starting with the detection of any novelty by a top “conservative” engine (neurons of context#1) to make sure that nothing is discarded. The samples recognized as novelty trigger the use of a second engine trained 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.

6 OPERATIONS IN NORMAL MODE

Under Normal operations, the neurons learn and recognize patterns as a Radial Basis Function (RBF) classifier and more precisely a Restricted Coulomb Energy (RCE) classifier. They can also recognize patterns as a K-Nearest Neighbor (KNN) classifier.

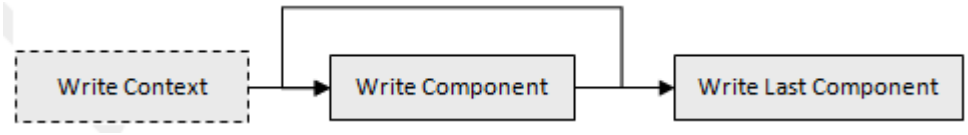
Under the SR mode, the automatic model generator and search-and-sort logic are disabled. The neurons become dummy memories but can be read or written in the least amount of time. This SR mode is essential to transfer knowledge bases between hardware platforms, or make backup prior to learning additional examples.

This section describes the possible sequences of operations in Normal mode and how the neurons handles them differently depending on their status as Idle, Ready-To-Learn or Committed.



6.1 VECTOR BROADCASTING

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



1) Write Context (optional)
   a) If the new vector must be associated to a context different than the current value of the Global Context or if the distance norm must be changed
2) Loop to write the N-1 components of the input vector

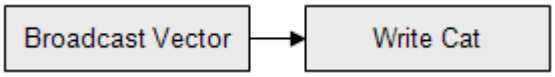
a) Write all the components of the input vector but the last one in the Ready-To-Learn. For all the committed neurons with a context equal to the Global Context, their distance register is updated after each Write Component per the Norm in use.

3) Write Last Component

a) For all the committed neurons with a context value equal to the Global Context register, their distance register is updated and represents the distance between the input vector and the prototype stored in their memory. If the distance of a neuron is less than its influence field, the neuron "fires" meaning that it is ready to respond to further inquiries such as a Read DIST or Read CAT commands. Also at the end of the Write Last Component, the entire neural network has been able to evaluate if the vector is recognized or not, and with uncertainty or not. Recognition exists is at least one neuron fires. Uncertainty exists if at least two of the firing neurons have a different category register.

6.2 VECTOR LEARNING

All the neurons have their internal learning logic and teaching a vector is as simple as broadcasting its components and then writing its category value.



If the pair (vector and category) represents novelty to the existing neurons, the Ready-To-Learn neuron becomes committed. It stores the instructed category in its category register. Its influence field of the new neuron is set to the smallest distance register of the firing neurons or the Minimum Influence Field whichever is greater. If they are no firing neurons at all, its influence field is set to the current value of the Maximum influence field. The next neuron in the chain turns from idle to RTL (ready-to-learn). If there are neurons which recognized the vector with a category other than the instructed category, they automatically reduce their influence field to prevent such erroneous recognition in the future.

If the AIF of a neuron reaches the Minimum Influence Field, the bit 15 of its category register is set to 1. The neuron is said "degenerated". It still reacts to input patterns as any other committed neuron but the bit 15 of its category indicates that the neuron was prevented from shrinking its AIF to a smaller value during training and its response should be weighted differently than the response of another firing neuron which is not degenerated.

6.2.1 GLOBAL SETTINGS PRIOR TO LEARNING

The following global registers affect the way the neurons will learn new vectors and model a decision space. Changing them should be done prior to broadcasting the vectors to learn.

Global Context	(to segment the network)
Maximum Influence Field	(to adjust conservatism)
Minimum Influence Field	(to control uncertainty domain)
KNN bit in Network Status	Must be set to 0. KNN is not a learning mode.

Remark: the minimum and maximum influence fields must be expressed in a dimension relevant to the dimension of the input vector.

The KNN mode is not appropriate for learning since it will create a new neuron each time a new category value is taught and do nothing more. The RBF mode must be selected to build a decision space modeling multiple categories and also with areas of “unknown” or “uncertainties” which are essential for true artificial intelligence with voluntary redundancy for accuracy, context awareness, hypothesis generation and more.

6.2.2 READING THE NUMBER OF COMMITTED NEURONS

The NCOUNT register returns the number of committed neurons in the chain. When the chain is full, NCOUNT=0xFFFF.

6.2.3 BUILDING A KNOWLEDGE INDEPENDENT OF THE TRAINING SEQUENCE

The decision space is modeled as examples are taught and consequently its shape depends on the sequence of the training examples. Indeed, their order determines the commitment of new neurons and the shrinking of existing committed neurons. This temporal dependency is not ideal 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.

The ability to execute an iterative learning requires that the training examples are stored and not streamed. The repetitive broadcast of a significant number of vectors can be time consuming, but the actual learning and modeling of the decision space triggered by the Write CAT instructions will always take a constant number of clock cycles regardless of the number of committed neurons (3 or 19 cycles depending on the recognition status).

For more information please refer to the NeuroMem Decision Space Mapping manual:

http://www.general-vision.com/documentation/TM_NeuroMem_Decision_Space_Mapping.pdf

6.2.4 CONTROLLING THE GENERALIZATION CAPABILITIES OF THE NETWORK

The generalization capability of the neurons is a significant strength since it means that learning a single relevant example can be enough to recognize many other similar cases. However, this strength can also lead to a lack of discrimination between subtle variations and become a weakness.

Three methods can be used to control the generalization capabilities of the neurons:

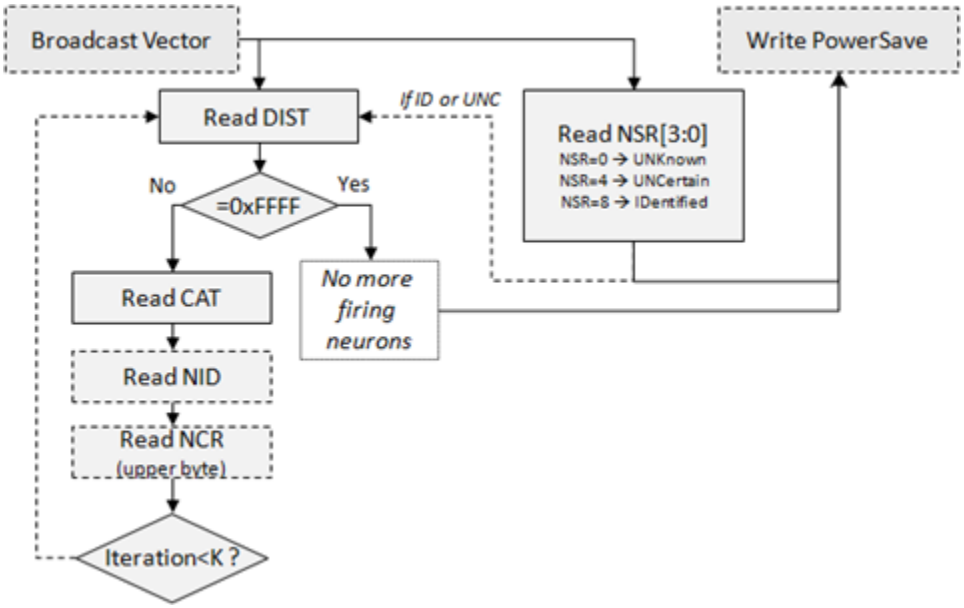
1. Learn many examples representing a broad range of contextual variations and, if possible, learn them in an iterative manner as described in the previous paragraph
2. Use the category 0 to learn counter examples for shrinking the influence field of neurons firing erroneously. This method is a dynamic and interactive correction which allows to only restrict the overgeneralization of certain neurons, as opposed to the next method based on the MAXIF which is a limitation imposed on all the neurons.
3. Reduce the value of the Maximum Influence Field (MAXIF) which becomes the default active influence field value of the non-committed neurons. The smaller the MAXIF, the lesser generalization or over-fitting of the neurons. Refer to the Mapping Decision Space Reference guide for more information. Note that the value of the MAXIF should be set in relation to the dimension represented by the feature vectors.

6.3 VECTOR RECOGNITION IN RBF MODE

The recognition of a vector is readily available after its broadcast to the neurons. Indeed, the neurons which identify the vector as falling within the similarity domain of the vector stored in their memory have their “fire” flag raised. The response of the firing neurons can be accessed with successive Read DIST, followed by Read CAT and optionally Read NID registers). The first distance quantifies the difference between the input 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 input 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. In the case of the RBF classifier, all the firing neurons have been read when Read DIST returns the value 0xFFFF.

The following diagram illustrates the three levels of response which can be delivered by the neurons through the readout of the registers NSR, DIST, CAT and NID:

- Conformity, or status of the recognition (identified, uncertain or unknown)
- Best match in distance and its associated category
- All possible matches listed per increasing distance values.



6.3.1 RESPONSE TYPE 1: CONFORMITY DETECTION

As soon as a vector is broadcasted to the neurons, the following recognition status is known:

- Unknown: no neuron recognizes the input vector
- Identified: one or several neurons recognize the vector and agree with its category
- Uncertain: one or several neurons recognize the vector but disagree about its category

6.3.2 RESPONSE TYPE 2: BEST-MATCH

The first neuron to respond is the firing neuron with the smallest distance value is equivalent to a best match. If its distance is equal to 0, it means that the vector matches exactly the prototype stored in the neuron. If the Recognition Status is Identified, the Best match is the only possible response. On the other hand, if the Recognition Status is Uncertain, other responses can be read from the neurons as described in the next paragraph.

6.3.3 RESPONSE TYPE 3: DETAILED MATCHES

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. The first distance quantifies the difference between the input 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 input 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 until all firing neurons have reported their response and the distance reads 0xFFFF.

If two neurons fire with the same distance but different category, their individual responses are read as follows: Read DIST, Read CAT, Read DIST, Read CAT. The second Read DIST returns the same value as the first Read DIST but is necessary to access the category register of the second neuron.

If two neurons fire with the same distance and same category, only the response of the first one is read. The first Read DIST will notify both neurons to stay in query, but both will output their category at the following Read CAT and therefore exclude themselves from the next query. A second Read DIST will return the next higher distance value if applicable.

If the category value is greater than 0x8000 or 32768 (bit 15=1) you have a warning that the neuron is “degenerated”. Masking bit 15 returns the real category value (AND with 0x7FFF). The degenerated flag simply indicates that the neuron was prevented from shrinking its AIF to a smaller value during training and that its response should be weighted with care, or simply differently than the response of a neuron which is not degenerated.

Reading the identifier of the neuron is optional. This feature can be useful to review the content of the neuron(s) which recognize the vector.

In the case of multiple firing neurons, a global response can then be established using probability functions, dispersion of the distances, minimum number of aggregates, etc.

Case study:

Let's take the example of a recognition where multiple firing neurons recognize a vector and return the following responses:

Distance	5	6	9	10	11	15	39
Category	8	8	7	7	7	7	5

The best match is a reference pattern of category 8 recognized with a distance 5. However, if we look at the response of all the firing neurons from a statistical stand point we can observe that the first two closest neurons report a category 7, but the next four firing neurons report a category 7 with a distance which is not that much bigger. 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. 8,8,7,7,7,7,3,5) and to be classified by another set of neurons taught to classify the "response" vectors. The CM1K chip can handle up to 127 subsets of neurons trained for different purposes. These subsets are called Contexts.

6.4 VECTOR RECOGNITION IN KNN MODE

In KNN mode, a vector is always recognized since the classifier discards the relationship between the distance and influence field of a neuron. Consequently, all the neurons fire and their distance and category can be read in sequence per increasing order of distance. The first distance quantifies the difference between the input 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 input 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 for K iterations.

Remark 1:

Using the neurons as a KNN classifier does not require to learn the vectors, but rather to load them to the neurons with or without category labels. This can be done in Save and Restore mode and executes in lesser time since it does not require the use of the model generator internal to the neurons.

Remark 2:

The KNN mode is not appropriate for learning since it will create one new neuron each time a new category value is taught and do nothing more.

Remark 3:

Technically, it is possible to teach the neurons with the RBF model generator and switch their behavior to KNN for the classification. However, you must be cautious with possible artifacts:

1) The closest neuron may have an influence field in the RBF decision space requiring the neuron NOT to fire
2) The closest neuron might not be that close, and it would be best to consider the stimuli as Unknown.

For more details, refer to the https://www.general-vision.com/documentation/TM_NeuroMem_Decision_Space_Mapping.pdf

6.5 PRACTICE

The following examples are simple test cases illustrating different behaviors of the neurons while learning and classifying patterns.

6.5.1 EXAMPLE 1

Step 1: Learn

Learn vector1 [11,11,11,11] (1) with category 55

Learn vector2 [15,15,15,15] with category 33(2)

Learn vector3 [20,20,20,20] with category 100

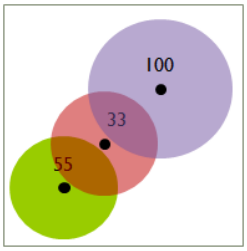
(1) The learned vectors are purposely set to arrays of constant values so their representation and relationship are easy to understand. The distance between two "flat" vectors is indeed the difference between their constant value times their number of components. For example, the distance between [11,11,11,11] and [15,15,15,15] is equal to 4 * 4. This simple arithmetic makes it easy to understand the different cases of recognition illustrated in this test script.

(2) The category of the second neuron is purposely set to a lesser value than the category of the first neuron to verify that if both neurons fire with a same distance, the category of the neuron on the 2nd chip is still the first the be read out

Fig1 is a representation of the decision space modeled by the 3 neurons where Neuron1 is shown in red, Neuron2 in green and Neuron3 in blue. In the following 2D graph, we limit the length of the models to 2 components instead of 4, so they can be positioned in an (X, Y) plot. X=1st component and Y=Last component, and a surrounding diamond shape with the side equal to their influence field.

Committed neurons= 3

NeuronID=1	Components=11, 11, 11, 11	AIF=16,	CAT=55
NeuronID=2	Components=15, 15, 15, 15,	AIF=16,	CAT=33
NeuronID=3	Components=20, 20, 20, 20,	AIF=20,	CAT=100

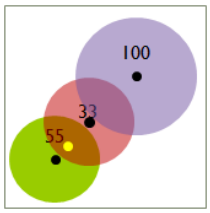


The influence fields of Neuron#0 and Neuron#1 overlap, as well as Neuron#1 and Neuron#2 overlap but differently since their distances from one another are different.

Step2: Recognition

The vectors submitted for recognition are selected purposely to illustrate cases of positive recognition with or without uncertainty, as well as cases of non-recognition. The program reads the responses of all the firing neurons, which is until the distance register returns a value 0xFFFF or 65553.

Case of uncertainty, closer to Neuron#1

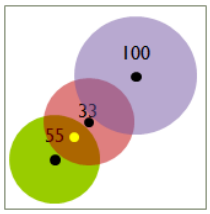


Vector=12, 12, 12, 12

Neuron 1 and 2 fire. Vector is closer to Neuron1

Response#1	Distance= 4	Category= 55	NeuronID= 1
Response#2	Distance= 12	Category= 33	NeuronID= 2
Response#3	Distance= 0xFFFF	Category= 0xFFFF	NeuronID= 0xFFFF

Case of uncertainty, equi-distant to Neuron#1 and Neuron#2

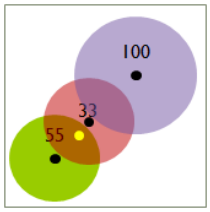


Vector=13, 13, 13, 13,

Neuron 1 and 2 fire. Vector is equi-distant to both neurons

Response#1	Distance= 8	Category= 33	NeuronID= 2
Response#2	Distance= 8	Category= 55	NeuronID= 1
Response#3	Distance= 0xFFFF	Category= 0xFFFF	NeuronID= 0xFFFF

Case of uncertainty, closer to Neuron#2

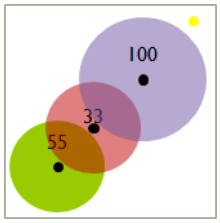


Vector=14, 14, 14, 14,

Neuron 1 and 2 fire. Vector is closer to Neuron2

Response#1	Distance= 4	Category= 33	NeuronID= 2
Response#2	Distance= 12	Category= 55	NeuronID= 1
Response#3	Distance= 0xFFFF	Category= 0xFFFF	NeuronID= 0xFFFF

Case of unknown



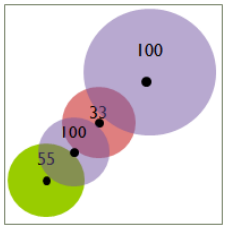
Vector=30, 30, 30, 30,

No neuron fire

Response#1	Distance= 0xFFFF	Category= 0xFFFF	NeuronID= 0xFFFF

Step3: Adaptive learning

Learn vector[13,13,13,13] with category 100. This vector is equidistant to both Neuron1 and Neuron2. Learning it as a new category, will force Neuron1 and 2 to shrink from their AIF=16 to an AIF=8 to make room for a new neuron which will hold the new model and its category.



Committed neurons= 4

NeuronID=1	Components=11, 11, 11, 11,	AIF=8,	CAT=55
NeuronID=2	Components=15, 15, 15, 15,	AIF=8	CAT=33
NeuronID=3	Components=20, 20, 20, 20,	AIF=20	CAT=100
NeuronID=4	Components=13, 13, 13, 13,	AIF=8,	CAT=100

Note that if the vector to learn was [12,12,12,12], Neuron1 would shrink to 4 and Neuron2 to 12.

6.5.2 EXAMPLE 2

Vector	Cmd	Description	1st best match	2nd best match
Vector 1= 0,1,2,3,4,5,6,7,8,9	Learn as 1	The 1st vector is stored in a first neuron		
	Reco	Its recognition generates an "exact match"	CAT=1 DIST=0	CAT=0xFFFF DIST=0xFFFF
Vector 2= 0,1,2,**6**,4,5,6,7,8,9	Reco	The 4th components of the 2nd vector is different by a value 3. All other components are identical. Still the 1st neuron recognizes the second vector.	CAT=1 DIST=3 NID=1	CAT=0xFFFF DIST=0xFFFF
Vector 3= 0,1,**4**,3,**8**,5,**12**,7,**16**,9	Learn as 2	The 3rd vector is stored in a second neuron		
	Reco	Its recognition generates an "exact match" with the second neuron.	CAT=2 DIST=0 NID=1	CAT=0xFFFF DIST=0xFFFF
Vector 4= 0,1,**2**,3,**4**,5,12,7,16,9	Reco	The 1st half of the 4th vector matches V1 and the 2nd half matches V3. Both neurons knowing V1 and V3 recognize V4. Neuron 2 gives the best match because it is closer with a distance of 6.	CAT=2, DIST=6 NID=2	CAT=1, DIST=14 NID=1
Vector 5= 0,1,2,3,4,5,6,7,16,9	Reco	2/3 of the components matches the one of V1 and 1/3 matches V3. Neuron 1 gives the best match with a distance of 8.	CAT=1, DIST=8 NID1	CAT=2, DIST=12 NID2

7 OPERATIONS IN SAVE_RESTORE MODE

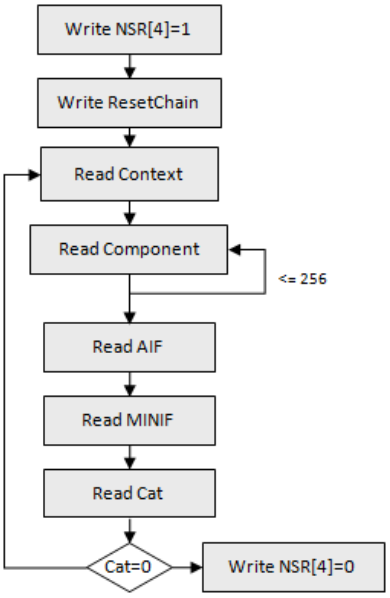
Under the SR mode, the neurons become dummy memories which can be read and written sequentially. This SR mode is essential to transfer knowledge bases between hardware platforms, or make backup prior to learning additional examples.

7.1 SAVE AND RESTORE OF THE NEURONS' CONTENT

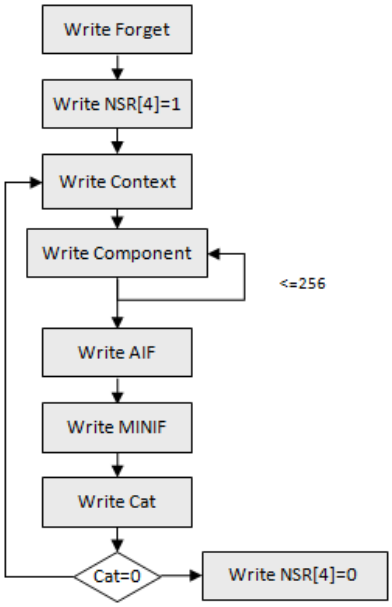
The content of the committed neurons describes a knowledge which can be saved and restored. This functionality is useful for backup purposes, but also to transfer and duplicate knowledge between NeuroMem networks.

The two functions require to set the neurons in Save_and_Restore mode and point to the first neuron of the chain. For each neuron, you can read or write its components, context, minimum influence field and active influence field in any order, except for the category register which must be read or written last to point to the next neuron in the chain. Finally, when the neurons have been saved or restored the last operation consists of setting the neurons back to their normal operation mode.

Save sequence



Restore sequence



In both sequences, once the neurons are set to the Save and Restore mode by writing bit 4 of the NSR to 1, the Category register must be the last register to be read or written per neuron since access to this register increments the neuron pointer automatically. The order in which the other neuron registers (COMP, AIF, and Context) are read is not important.

7.1.1 IMPORTANT REMARKS

Note that in Save_and_Restore mode the last component is read and written to the COMP register and not to the LCOMP register.

If it is known that all neurons hold a pattern with only M significant components with M<256, the number of Read COMP can be limited to M, thus speeding the Save operation.

If an application does not use the notion of context, saving the context register might not necessary, saving one clock cycle per saved neuron.

Saving and restoring the MINIF of each neuron is necessary if additional training will be done later to complete or expand the knowledge. This will ensure that the degenerated status of the neurons is properly flagged if appropriate.

7.1.2 WARNING ABOUT MERGING KNOWLEDGE

Appending several knowledge bases in Save and Restore mode is relevant only when the neurons of each knowledge base are associated to different contexts and thus trained independently.

Usually, loading a knowledge to the neurons is preceded with a FORGET command, and even a function clearing the 256 bytes of memory of all neurons using the Write TESTCOMP. This ensures that the first neuron to be written in SR mode will be the first neuron of the chain.

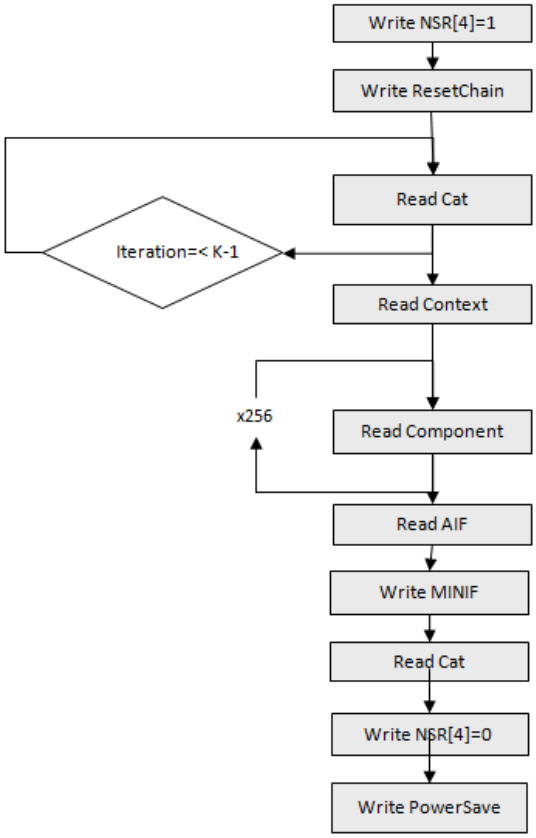
If this precaution is not taken, then the first neuron to be written after switching the network in SR mode will be the RTL neuron.

You need to be very cautious and clearly understand the consequences of appending the content of neurons to an existing knowledge already residing in a chip. Refer to the next chapter about Knowledge Management.

7.2 READING THE CONTENTS OF A SPECIFIC NEURON

Reading the contents of a specific neuron is made in the following order:

- The first operation consists of setting the neurons in Save_and_Restore mode and pointing to the first neuron of the chain
- In order to point to the i^{th} neuron in the chain, (i-1) consecutives Read CAT are necessary
- You can then read the i^{th} neuron's components, context, minimum influence field and active influence field in any order. The category register must be read last because the instruction automatically points to the next neuron in the chain.
- Finally, the last operation consists of setting the neurons back to the normal mode.



8 KNOWLEDGE BASE

8.1 FORMATTING KNOWLEDGE FILES

The content of the committed neurons describes a knowledge which can be saved and restored. This functionality is useful for backup purposes, but also to transfer and duplicate knowledge between NeuroMem networks.

For portability and re-use, the knowledge file must also be accompanied with a header or associated file describing the feature extraction functions and parameters used to generate the models stored in the neurons per context value.

8.2 MERGING KNOWLEDGE BASES

If you understand the powerful autonomous adaptive learning capabilities of the neurons, you will not be surprised that merging knowledge bases built by different networks (residing or not in a same physical chain of chips) cannot be as simple as appending two files together and should be handled with extreme caution.

8.2.1 CASE #1: INDEPENDENT NETWORKS

The 2 knowledge files, knA and knB, describe neurons trained with feature vectors of type A and B which have no relationship, that is issued from different feature extraction functions or parameters. The original data source used to extract these feature vectors can be the same but their dimensions have nothing in common.

Example #1:

> Visual Objects learned using Feature #1= monochrome subsample and Feature #2= color histogram
>
> Visual Objects learned using Feature #1= subsample at scale x1 and Feature #2= subsample at scale x5
>
> Persons identified with Faces learned with Feature#1 and Voices learned with Feature#2

Consequently, the learning of the feature vectors of type A has no impact on the influence fields of the neurons holding feature vectors of type B, and vice versa.

The merging of the two knowledge files can be made simply under the Save and Restore mode.

The only imperative is that, if not already the case, the Neuron Context Register (NCR) of the neurons of knA and knB will be respectively assigned to 2 different values a and b.

8.2.2 CASE #2: RELATED NETWORKS

The 2 knowledge files, knA and knB, describe neurons trained separately but with some neurons of knA featuring vectors extracted with the same function and parameters as in knB, and/or vice versa.

Consequently, if the feature vectors of knA were to be taught to the neurons holding the knB, there is a possibility that these neurons will adjust their influence fields and commit new neurons, and vice versa.

The merging of the two knowledge files requires that all the models be re-learned with their categories. Another imperative is that the models extracted with the function and parameters be taught under a same value of the Global Context Register (GCR).

IMPORTANT: Merging the knowledge knA and knB will never be as rich and robust as re-learning the original examples used to build these 2 knowledges. Indeed, knA and knB are already an expression of a decision space which has been modeled by learning examples in a given sequence, with given values of the MINIF and MAXIF. Whenever possible, it is recommended to merge the learning process rather than their resulting knowledge bases.

9 NEUROMEM REGISTERS

The following table describes the 15 registers controlling the entire behavior of the neurons under the Normal and Save-and-Restore mode.

	Description	**Normal mode**	**SR mode**
NSR	Network Status Register	RW	W
GCR	Global Control Register	RW	
MINIF	Minimum Influence Field	RW	RW
MAXIF	Maximum Influence Field	RW	
NCR	Neuron Context Register		RW
COMP	Component	W	RW
LCOMP	Last Component	W	

	Description	Normal mode	SR mode
INDEXCOMP	Component index	W	W
DIST	Distance register	R	R
CAT	Category register	RW	RW
AIF	Active Influence Field		RW
NID	Neuron Identifier	R	R
FORGET	Forget	W	
NCOUNT	Count of committed neurons	R	R
RESETCHAIN	Points to the first neuron		W
TESTCOMP	Test Component		W
TESTCAT	Test Category		W

Refer to the description of your NeuroMem chip for a detailed description of each register, its dimension, default value, bit flags and more.

9.1 NEURON BEHAVIOR PER STATUS PER INSTRUCTION

The following table describes how the memory of the neurons is updated depending on its state in the chain of neurons.

Memory	**Idle neuron**	**Ready to Learn neuron**	**Committed neuron**
COMP 0	Do nothing	Takes the value of the 1st Write COMP occurring after a Write LCOMP. The memory index is incremented by 1 to point to the next component.	Can only be changed by a reset or restore operation. Reset the distance register The memory index is incremented by 1 to point to the next component.
COMP 1	Do nothing	Takes the value of the next Write Comp or Write LCOMP. The memory index is incremented by 1 after a Write Comp, or is reset to 0 after a Write LCOMP.	Can only be changed by a reset or restore operation. The memory index is incremented by 1 after a Write Comp, or is reset to 0 after a Write LCOMP.
…			
COMP N	Do nothing	Takes the value of the next Write Comp or Write LCOMP. The memory index is reset to 0.	Can only be changed by a reset or restore operation.

The following table describes how the registers of the neurons is updated depending on its state in the chain of neurons.

Registers	**Idle neuron**	**Ready to Learn neuron**	**Committed**
GCR	Takes the value of the Write GCR.	Takes the value of the Write GCR.	Do nothing
		Current value is saved if the neuron gets committed after a Write CAT.	Can only be changed by a reset or restore operation.

MINIF	Takes the value of the Write MINIF.	Takes the value of the Write GCR.	Do nothing
MAXIF	Takes the value of Write MAXIF.	Takes the value of the Write GCR.	Do nothing
DIST			If its NCR=GCR, Accumulates the difference between the pointed Component and the input value after each Write Comp or Write LCOMP.
CAT		Value is written if no committed neuron fires and has its own category equal to value. The neuron status switches from RTL to Committed.	
AIF			If its NCR=GCR, Inherits the smallest distance value of the firing neurons

9.2 COMMANDS CHANGING THE RTL NEURON IN CHAIN

Memory cell index change	**Normal mode**	**Save and Restore mode**
Write COMP	Index + 1	Index + 1
Write LCOMP	Index =0	
Write INDEXCOMP	Index=k	Index=k
Write TESTCOMP		Index + 1
Write NSR	Index=0	Index=0
Write CAT		Index=0
Read CAT		Index=0