

An Overview of Unsupervised Neural Networks By Dan Schlegel

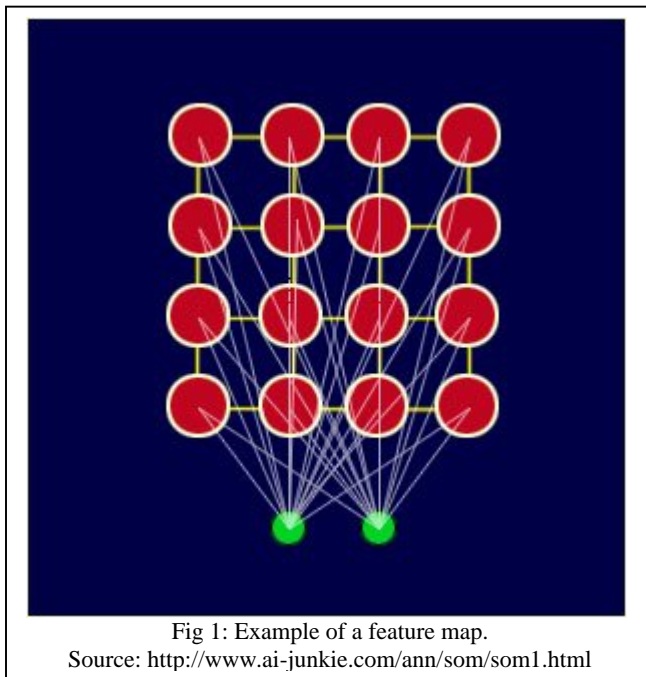
Neural networks in a data retrieval system have been theorized for everything from high speed internet search, to desktop search applications, to biologically faithful associative memory systems. This paper will serve to discuss the current state of development of architectures of retrieval of data from a neural network. Discussed will be two unsupervised networks – self organized maps and adaptive resonance theory, along with the PCA-CG model of a trained neural net. Finally we will discuss the biological faithfulness of the discussed algorithms.

At the core of an efficient learning data retrieval system lies either Hebbian or error driven models – or perhaps even both. The Hebbian component aims to generalize amongst data sets and find commonalities in large sets of data over a long period of time. The biological counterpart to this would be the thought of what a chair is. When you think of a chair you see it in many different contexts, shapes, and sizes. This embodies a kind of synthesis of large amounts of data over time. This same principle is needed for an efficient, thorough and accurate search system. The error driven component learns through positive or negative re-enforcement from the user. This is learned quickly and can immediately affect results. This is a far less abstract component and is used in the brain every time information is learned, corrected, or reinforced.

As an introduction to data retrieval systems, it is important to understand how a most basic of these could be defined. In terms of graph theory a neural network is a series of nodes connected to each other by edges. Related nodes are connected to each other, and each edge is given a weight which defines how strongly one node relates to another.

If inputs to a node reaches that nodes activation value than that node will fire off a signal to connected nodes and the processes will continue until an output (or no output) is reached. In a data retrieval system the nodes would contain data of some nature, and related ones would be connected.

One method for maintaining connections in a meaningful manner is known as a



self organizing map. One example of such a system is Kohonen's Feature Map. Input to the feature map is a set of objects represented by N-dimensional vectors, then mapped onto a two dimensional grid.¹ Each node is described by an n-dimensional vector and an n-dimensional weight. From the set of all inputs, a random input vector is

selected. The node whose weight is closest to that of the input value is then identified and referred to as the winning node, or the Best Matching Unit (BMU). The radius of the "neighborhood" of the BMU is identified – this is typically the full radius of the lattice to begin with, growing smaller with each iteration of the algorithm. The weights of the nodes in the neighborhood are modified to be closer to the input weight.² As you can imagine as the radius gets smaller, those nodes closer to the input node have weights

¹Xia Lin, Gary Marchionini, and Dagobert Soergel. "A Self-organizing Semantic Map for Information Retrieval." In *Proceedings of the 14th Annual international ACM SIGIR Conference on Research and Development in information Retrieval* (Chicago, Illinois, United States, October 13 - 16, 1991). SIGIR '91. ACM Press, New York, NY, 262-269.

²"SOM Tutorial Part 1," *ai-junkie*. <<http://www.ai-junkie.com/ann/som/som2.html>>

closer to it as well. Each time this process is carried out, it is referred to as an epoch.

“This process goes through many iterations until it converges, i.e., the adjustments all approach zero. Each input vector is then mapped to a grid node closest to it in the N-dimensional space. ... It results in an orderly feature map.”³

This organization is possible because of an inherent property of neural networks – that they are extremely good at producing useful categories of objects. It is not necessarily known how the objects will arrange themselves prior to the simulation (although using some statistical calculations it can be predicted) and is in this aspect at least a “black box” algorithm. At the core of neural networks is a great deal of statistical measures which determine over time how to organize objects in a meaningful manner to the network.

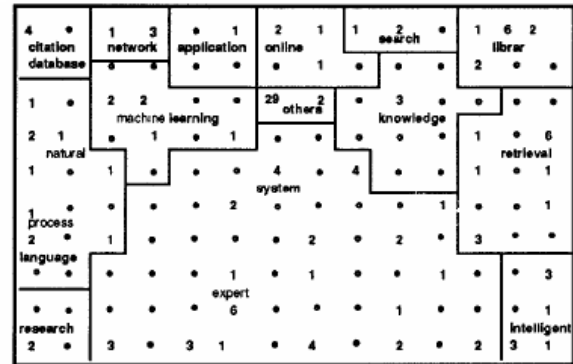
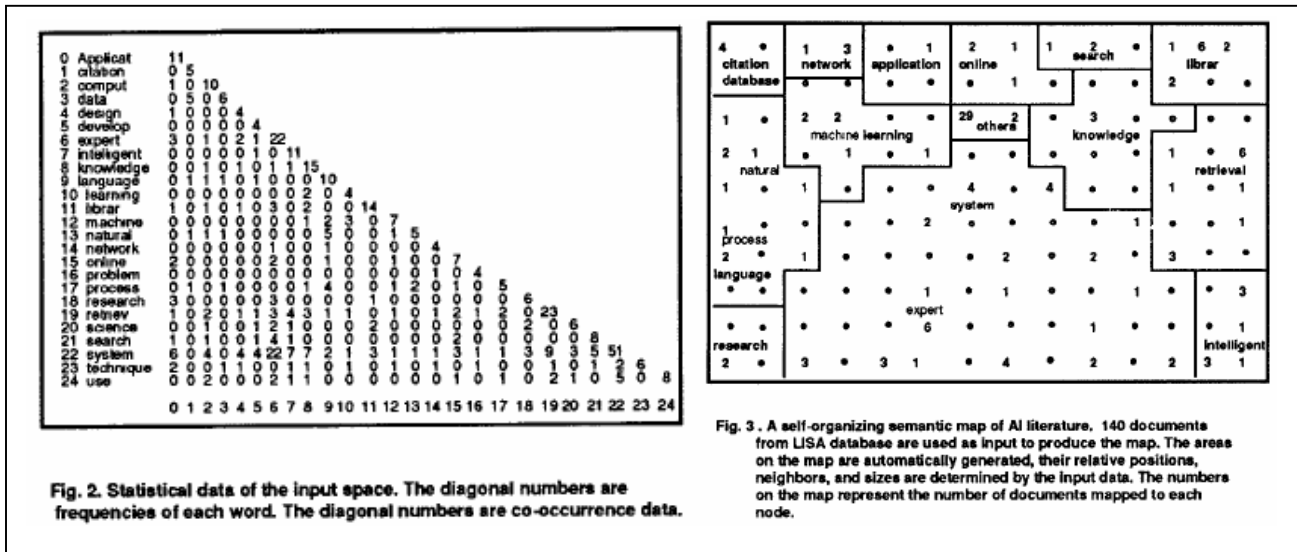
The feature map requires no outside interaction to complete its organization, and therefore is referred to as unsupervised. In order to ensure the algorithm does in fact result in convergence, two controls are placed on the algorithm. The first of which is the fact that the neighborhood grows smaller with each iteration. The large neighborhood results in the ordering of the network, and the smaller one leads to stable convergence. A second control is that of the “adaptive gain parameter.” This determines how much adaptation is occurring across epochs, and therefore as it approaches zero the map has converged.⁴

It is conceivable that a map such as this could be used to semantically organize words to assist in the identification of relevant documents in a data retrieval system. This is similar to the experiment carried out by Lin and Soergel. In the experiment words in

³ Lin, Marchionini, and Soergel, 263

⁴ Lin, Marchionini, and Soergel, 263

titles of AI literature were given to the network, each word is its own input, with a vector indicating the entire title. Figure 2 above represents the statistical data of the occurrence



of each word in the titles. Figure 3 is the final output of the network after 2,500 iterations.

You can see that words that would be related in a title are closer together, and that the amount of times a word is used leads to a larger box being generated for it.⁵

There are very significant drawbacks to this kind of algorithm though. Being unsupervised it trains itself when data is modified in the network. Computationally you can see that this is not a simple algorithm, and it requires many iterations to be useful. Unless this could be calculated more quickly and even then in the computers “spare time”, then it is not a feasible algorithm to use for every day personal computer information storage and retrieval. Also being that it is unsupervised any mistakes it makes will have to be manually corrected by the user to better obtain usable results.

A second method of unsupervised data categorization utilizes what is known as the Adaptive Resonance Theory, or ART. In our discussion of ART we will be talking

⁵ Lin, Marchionini, and Soergel, 264-265.

about ART-1 which is the most simple of the ART algorithms and deals in binary classification. In a discussion of ART it is necessary to discuss first what ART is meant to solve in neural network systems. The primary issue at hand is one of plasticity versus stability. To be plastic means that the network is able to constantly modify itself based on input vectors to categorize them properly. This plasticity is something that self organizing maps lack – they require recategorization very often to remain valid. The issue is that as this happens the network becomes unstable very quickly – meaning it “loses” or “unlearns” information that was used before to create proper categories. ART is an algorithm meant to balance these two problems.⁶

ART is based on two marginally complex algorithms, one for clustering, and one for competition. The clustering in ART-1 has an “incremental update of prototype vectors and a variable number of clusters.”⁷ This means that there is a vector describing a cluster known as the prototype vector. This vector is modified if an input vector is “close enough” to the prototype vector. Grossberg and Carpenter call this concept *match-based learning*. “Match-based learning allows memories to change only when input from the external world is close enough to internal expectations or when something completely new occurs.”⁸ In the event mentioned by Grossberg and Carpenter when something completely new occurs, a new prototype vector is generated describing a new cluster. As you can see this allows for stability within clusters, while still allowing plasticity by the creation of both new clusters and slight modification of existent ones.

⁶ Heins, L.G. and Tauritz, D.R.. “Adaptive Resonance Theory: An Introduction.” In *Technical Report*, 95-35, Leiden University, 1995, < http://www.cs.brandeis.edu/~cs113/docs/other/heins_tauritz.pdf>, p1-4.

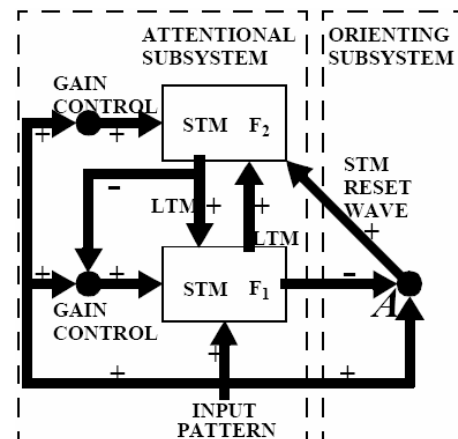
⁷ Heins, and Tauritz, p1-4.

⁸ Gail A. Carpenter and Stephen Grossberg. “Adaptive Resonance Theory” September 1998, p2.

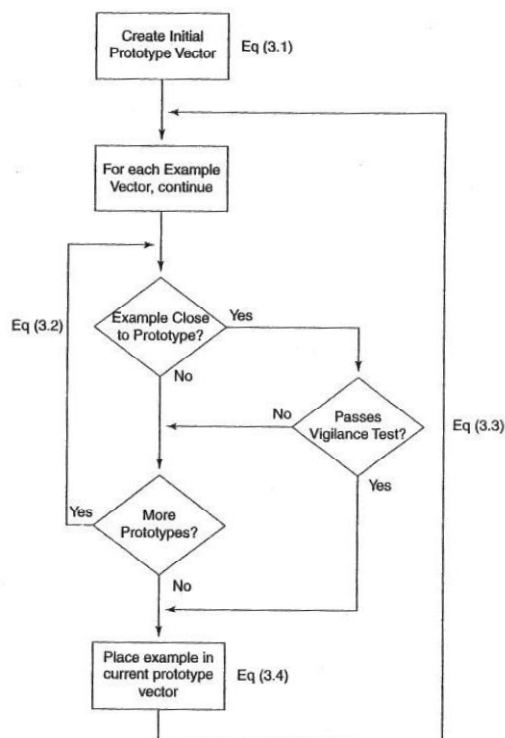
ART uses a competitive network algorithm to determine where best to place a node in the network. A standard two-layer competitive network has a layer which is activated in a unique way based on the input that is shown to it. This is a short term memory layer. This is then sent to a second layer which is attached by connections with specific weights. The input to layer 2 is the input multiplied by these weights. The output with the largest dot product of the input and the prototype is the most closely matched cluster. The weights connecting layers one and two are modified to better accommodate this input pattern. This turns the short term memory into a long term memory pattern. This is a highly simplified description, but is the general idea of a competitive network. Unfortunately this is not a stable solution, and ART seeks to solve this.⁹

The ART-1 competition algorithm bases its learning on a “hypothesis,” which is generated as follows. The input is scaled such that it only barely activates (supraliminally activates) the layer one

nodes.



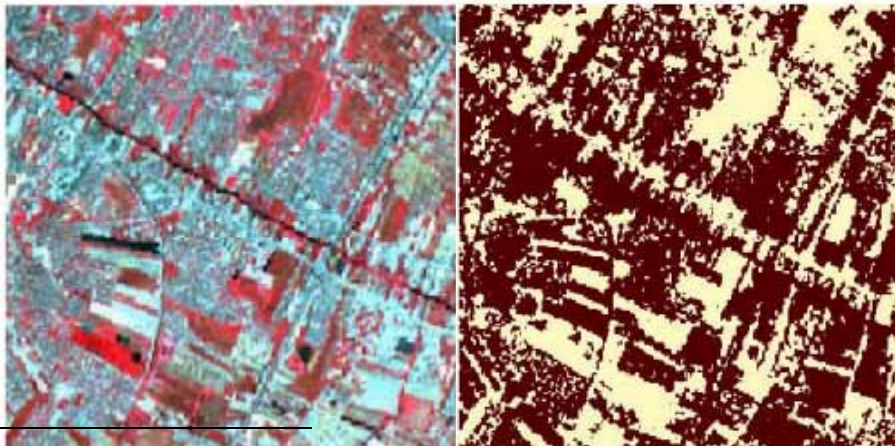
This is generated using a “bottom-up” procedure. ART-1 has both bottom-up and top-down procedures to form a hypothesis. Starting at the other end, the top-down procedure generates a vector V . The nodes where the I and V vectors intersect remain active. If V



mismatches I , the inhibition between the first layer and A in the image to the above is decreased. After the inhibition in the layer 2 node has been increased, the process starts again, activating a different node. Note that this rapidly repeats (note rapidly means very little or no learning takes place) until a layer 2 node is chosen whose top-down expectation V closely matches I , until a uncommitted layer 2 node is selected, or finally until it is determined I cannot be accommodated by the system.

Now that the method of competition and categorization of the ART1 algorithm is understood, we can discuss how the actual algorithm uses this to carry out the algorithm. This can be described fairly basically in the above rendering. The prototype vector mentioned is the "center" of the cluster - or ideal case. The example vector is a feature vector in this case. The categorization section of the algorithm is used in determining of the prototype is close to the example vector (this is referred to as "resonance"). You can see that the competition algorithm can be used here in the "vigilance test" portion of the diagram.¹⁰

ART has many applications, as many if not more than self organizing maps. One relevant example I wish to discuss here is an experiment carried out by Supoj Mongkolworaphol, Yuttapong Rangsanteri and Punya Thitimajhima from the



¹⁰ M. Tim Jones. AI Application Programming, (Mass: 2005), p35-58.

Department of Telecommunications Engineering at King Mongkut's Institute of Technology Ladkrabang in Bangkok. Their experiment used ADEOS satellite images of Bangkok to analyze urban structure and an ART algorithm to pick out relevant parts of the images. As you can see in the two included images the ART algorithm successfully picked out urban structures. For their experiment they used an ART2 algorithm (which uses floating point data) to analyze each pixel of the image and categorize it. The pixels in the category of interest were then dumped to the screen in a meaningful manner to be analyzed.¹¹ It can easily be seen from this the application to data retrieval. The ability to pick out relevant details and allow visualization in a meaningful way is crucial to a usable system for data retrieval.

There are many similarities between these two algorithms, and quite a few differences. Both algorithms serve to organize data into a useful ordering for quick, associative retrieval. The primary difference comes when data is added to these networks. Self organizing maps require a complete set of data to sort, and this reduces the plasticity of the network. There is no real mechanism for adding data later to the network without a complete reorganization of the network unless something very similar to that data is already accounted for. ART on the other hand is better prepared to add data at a later time – with the ability to on the fly create new categories and easily update old ones.

Unsupervised neural networks can live entirely unsupervised, but they are far more useful if they also have an error driven (supervised) component. This can allow correction of any mistakes the network happens to make without a full recalculation of the network after changes to the code have been manually made. This forces a

¹¹ Supoj Mongkolworaphol, Yuttapong Rangsanseri and Punya Thitimajhima, Asian Conference on Remote Sensing 2000 Proceedings, <<http://www.gisdevelopment.net/aars/acrs/2000/ts9/imgp0002.shtml>>

readjustment to the weights when it would not normally have been done by the network itself. This can be done without a training set - metaphors for user response to the output via the GUI are possible.

The issue of training a neural network in a supervised situation is one which is difficult to carry out successfully. Training the neural network generally creates main categories where information will fall. A training set is generally used that will create these categories in a way that will lead to a functional neural network that carries out its very specific tasks accordingly. In a training set the desired output is known, along with a set of input values. Once the network is trained, it can then be used in large scale situations. Choosing a training set is particularly difficult – several cases of failure can occur. One failure case for example is the network learning what shouldn't fall into a category, rather than what should, which will lead to further problems as the network is utilized outside the test set. Another would be if the test set was not varied properly and the network learned about the incorrect features of a data set. An example could be a neural net to detect a tank in the woods. Several pictures would be used as training both with and without the tank, but if the pictures with the tank were taken on a sunny day and the pictures without were taken on a cloudy day the neural net could just as well be a weather detector as a tank detector.

The networks discussed before were not at all the conventional style network used for a trained neural network in data retrieval; in fact they weren't even what we usually think of in terms of a neural network in general. In this situation a standard model is one of only three layers - an input, hidden, and output layer. The number of hidden nodes can be defined by the designer or heuristically or algorithmically generated. This is a simple

fully connected feed-forward network with weights defined for each connection between neurons. "A sigmoid shaped activation function is usually used as the elements in the processing layers called 'artificial neurons'."¹² The input and output layers correlate with the number of inputs and outputs required in the situation for which the network is being designed. The weights connecting the network are generally random prior to being trained. In order to properly set these weights to useful values the network is then given a known set of inputs and outputs and the weights are adjusted accordingly over time.

There is an extremely large number of ways to create a neural network for data storage and retrieval that is trained using error-driven methods. Only one method will be discussed here but it is a very good example of the possible combination of algorithms that is possible for data storage and retrieval in a neural network system. The situation we will discuss uses the PCA-CG algorithm to suggest the optimal number of hidden nodes in our network to begin training. The training will be done using conjugate gradients - a very complex yet efficient way of organizing large amounts of complex data. Because our network will have a large number of inputs and outputs a third algorithm can be used to separate out relevant and irrelevant input data - even for small training sets that do not validate particularly well.¹³

The PCA-CG algorithm takes care of many parts of the standard network model for data retrieval. It suggests an optimal number of hidden nodes to be used in the model and generates non-random weights prior to training (in contrast with the random ones of the standard model). Unfortunately there is a large amount of math involved in these

¹² Zvi Boger and Rene Weber. "Finding an Optimal Artificial Neural Network Topology in Real-Life Modeling - Two Approaches." In Proceedings of the ICSC Symposium on Neural Computation (NC'2000), <<http://www.dr-rene-weber.de/files/nc2000.paper1403-109.pdf>>, p2

¹³ Zvi Boger and Rene Weber, p2.

calculations, so the explanations here will be somewhat sparse. The entire system is approximated as being linear - different from standard neural nets which are described as non-linear normally. The number of hidden nodes is then generated from "the number of the major principle component analysis (PCA) axis in the data." The number of hidden nodes is optimally around 5 for very large networks even. It has been shown that too many nodes in the network degrade performance and accuracy. The fully connected nature of the network leads to a huge increase in connections (and weights) for each node added. This is a case where "less is more" applies to a great extent. It is worth noting though that too few nodes will not properly capture the complexity of the input data. An alternate method for measuring the number of hidden nodes is by the use of trial and error in genetic algorithms - which tends to require more development effort and more computing power but may yield a better result.¹⁴

Since the PCA-CG algorithm reduces the non-linear nature of neural networks into a system of linear equations, the conjugate gradient method can be used to solve the system of equations. The conjugate gradient method is very complex, but is recognized as the most popular method for solving a system of linear equations. Johnathan Shewchuk from Carnegie Mellon University explains that CG is effective for equations of the form $Ax=b$ where x is an unknown vector, b is a known vector, and A is a matrix. The matrix A should be a sparse matrix if this algorithm is to be used. If it is a dense matrix it can better be solved using other - including iterative - methods. One method for solving the system with a sparse matrix is the method of steepest decent ("sliding" down the paraboloid until the algorithm is confident it is "close enough" to the solution).¹⁵

¹⁴ Zvi Boger and Rene Weber, p1-2

¹⁵ Zvi Boger and Rene Weber, p1-3

The algorithm used to reduce the number of inputs is a statistical network reduction algorithm. In general these algorithms take the values from each output node and determine which are significantly activated. "Significant" is statistically measured based on the entire set of outputs with the possibility here for user intervention on the specificity or generality of the results. Insignificant outputs are mapped to their respective inputs and those inputs are disregarded for this search. This remaining input data can now be ranked and sorted for display to the user.¹⁶

As you can see the trained model feels much more natural to someone versed in neural networks already as it uses many concepts we are already used to. Here you can see that the algorithms for a trained system could be swapped out for others that may better fit a task, or be more optimal. It is very convenient to the programmer that the architecture of this kind of network permits such a thing – in contrast to the unsupervised networks discussed which are fairly non-variable models.

I believe an important point to make about all of these neural network models for data retrieval is that there can be no strictly biological claims made about any of them but that there is a possibility that they can be used as metaphorical modeling tools. All of the networks shown have at least sections of them that simply have no biological precedent. The trained neural net mentioned here is close to a biological representation, stemming from the multilayer perceptron model. This too though has the constraint placed on it that it is feed-forward only, which is not a biologically faithful model. It has been shown that biological systems have a method of feedback, but not the feedback visible in a standard backpropagation algorithm seen in many data retrieval neural networks (which sends an

¹⁶ Zvi Boger and Rene Weber, p2-4.

“error signal” backwards through the network).¹⁷ The feedback visible in biological networks is more similar to the GeneRec algorithm which uses inhibition. Of the unsupervised networks self organizing maps in particular are not biologically faithful in their structure or modeling techniques. The iteration over a lattice would never be seen in a biological network of neurons, but as is explained by O’Reilly and Munakata below it can be used as a valid metaphor. The ART algorithm mentioned was developed to mimic biological processes in its unsupervised nature and self organization techniques.¹⁸ As explained by O’Reilly and Munakata ART along with SOM’s ability to classify and categorize data can be seen as an interpretation of conditional principal components analysis algorithms (as seen in the trained PCA-CG model shown above). The biological metaphor here breaks down because there is no inhibitory function affecting activation as seen in the k-Winner Takes All model for Hebbian learning.¹⁹ This substantiates the claim that these networks may be good models while not necessarily biologically accurate. As you can see from this these networks are not only useful for non biological applications, they can be used as a metaphor for modeling neural processes.²⁰

All the networks shown throughout this paper have their own flaws and advantages. Self organizing maps are very slow and are not particularly plastic but can be used for visualization and organization of static data very well (perhaps in a library catalog system as seen with the book titles above). Adaptive Resonance Theory is plastic, stable, and very good at organizing data, but it is extremely complex and difficult to

¹⁷ Thomas P. Trappenberg, “Continuous Attractor Neural Networks” in Recent Developments in Biologically Inspired Computing, ed. Leandro Nunes de Castro and Fernando J Von Zuben. <<http://users.cs.dal.ca/~tt/papers/CANNchapter.pdf>> , p4

¹⁸ Ke Chen, “Lecture Slides for Nature Inspired Learning,” <http://www.ifi.unizh.ch/ailab/teaching/NN2003/ART/Lecture_on_ART.pdf>

¹⁹ Randall C. O’Reilly and Yuko Munakata. Computational Explorations in Cognitive Neuroscience, MIT Press, Mass: 2000, p143

²⁰ Trappenberg, p4-5

implement. An extension of a standard model such as PCA-CG is stable, yet not particularly plastic and difficult to train. Also none of these tend to be strictly biologically faithful tools for modeling biological processes. As you can see there is still much work to be done in the field of data retrieval using neural networks, and the years ahead shall show a large amount of improvement.

Works Cited

- Boger, Zvi and Weber, Rene. "Finding an Optimal Artificial Neural Network Topology in Real-Life Modeling - Two Approaches." In Proceedings of the ICSC Symposium on Neural Computation (NC'2000), <http://www.dr-rene-weber.de/files/nc2000.paper1403-109.pdf>.
- Chen, Ke. "Lecture Slides for Nature Inspired Learning,"
http://www.ifi.unizh.ch/ailab/teaching/NN2003/ART/Lecture_on_ART.pdf.
- Carpenter, Gail A. and Grossberg, Stephen. "Adaptive Resonance Theory" September 1998.
- Heins, L.G. and Tauritz, D.R.. "Adaptive Resonance Theory: An Introduction." In *Technical Report*, 95-35, Leiden University, 1995,
http://www.cs.brandeis.edu/~cs113/docs/other/heins_tauritz.pdf.
- Jones, M. Tim. AI Application Programming, (Mass: 2005).
- Lin, Xia, Marchionini, Gary, and Soergel, Dagobert. "A Self-organizing Semantic Map for Information Retrieval." In *Proceedings of the 14th Annual international ACM SIGIR Conference on Research and Development in information Retrieval* (Chicago, Illinois, United States, October 13 - 16, 1991). SIGIR '91. ACM Press, New York, NY.
- Mongkolworaphol, Rangsaneri and Thitimajhima, Asian Conference on Remote Sensing 2000 Proceedings,
<http://www.gisdevelopment.net/aars/acrs/2000/ts9/imgp0002.shtml>.

Randall C. O'Reilly and Yuko Munakata. Computational Explorations in Cognitive Neuroscience, MIT Press, Mass: 2000.

“SOM Tutorial Part 1,” *ai-junkie*. <http://www.ai-junkie.com/ann/som/som2.html>

Trappenberg, Thomas P. “Continuous Attractor Neural Networks” in Recent Developments in Biologically Inspired Computing, ed. Leandro Nunes de Castro and Fernando J Von Zuben, <http://users.cs.dal.ca/~tt/papers/CANNchapter.pdf>.