Neural Networks in Analog Hardware - Design and Implementation Issues

Sorin Draghici
431 State Hall, Department of Computer Science,
Wayne State University, Detroit, MI, 48202, USA

February 7, 2000

Abstract

This paper presents a brief review of some analog hardware implementations of neural networks. Several criteria for the classification of general neural networks implementations are discussed and a taxonomy induced by these criteria is presented. The paper also discusses some characteristics of analog implementations as well as some trade-offs and issues identified in the work reviewed. Parameters such as precision, chip area, power consumption, speed and noise susceptibility are discussed in the context of neural implementations. A unified review of various 'VLSI friendly' algorithms is also presented. The paper concludes with some conclusions drawn from the analysis of the implementations presented.

Keywords: analog hardware implementations, area, precision, speed, trade-offs, VLSI. friendly learning algorithms, neural networks, CMOS technology, VLSI

1 Introduction

A necessary and essential step in continuing the diffusion of the neural network paradigms in our day by day use is their hardware implementation. The ability of the neural systems to be implemented in hardware is crucial because the normal requirements for many applications (e.g. aerospace, military, etc.) include: i) a small volume, ii) a reduced weight, iii) a high resistance to shocks, vibrations and adverse environmental conditions and iv) a high execution speed. Furthermore, the hardware implementation is by far the most cost effective solution for large scale use.

The paper has four goals as follows: i) to place analog implementations in the context of other types of implementations by pointing out their advantages and disadvantages; ii) to review a few of the existing neural network implementations in analog hardware; iii) to extract and discuss some issues important to all such implementations; and iv) to provide a comprehensive list of references related to hardware implementations.

Due to the large number of neurocomputers designs, the length of the paper was a limitating factor. More information about neural network hardware can be found in other review papers such as [84], [88], [94], [95], [108], [126], [158], [159], [113],[79], chapters [30] and books such as [68], [74], [114], [134], [136], [157], [96].

The paper is organized as follows. First, we discuss several general criteria used in the classification of neural networks. Based on these criteria, we present a taxonomy of existing neural implementations. This taxonomy is not limited to analog hardware. Section 2 is the core of the paper. Section 2.1 presents some characteristics of analog hardware and a comparison with digital hardware. A review of the existing analog hardware implementations is presented in sections 2.2 and 2.3. Section 2.4 discusses briefly some trends and directions that can be identified in the work reviewed. Section 2.5 presents some important issues resulted from the analysis of the existing hardware. Such issues include the various trade-off between several parameters important for an analog implementations and VLSI friendly learning techniques. Some conclusions are presented in section 3.

1.1 A taxonomy of neural network hardware

A few criteria that can be used to classify neural network hardware and the different categories that they determine will be reviewed briefly in the following. The various criteria and the classification they determine are presented in Fig 1.

- Implementation type
  - digital
  - analog
  - mixed (digital/analog)
  - optical
  - pulse-stream
  - RAM-based, etc.
- Interneuron communication paradigm:
  - layered architecture
  - fully connected network
  - reconfigurable architecture
  - locally connected network (e.g cellular neural networks, feature maps)
- Type of architecture:
  - chip with on-chip controller
  - stand-alone neurocomputer
  - accelerator board
- Degree of parallelism:
  - low parallelism (less than $10 - 15$ processors)
  - medium parallelism ($10 - 10^2$ processors)
  - high parallelism ($10^2 - 10^3$ processors)
  - massive parallelism (more than 10^3 processors)
- Neural network or biological architecture implemented:
  - model-free learning (e.g. learning by perturbation)
  - general purpose but implementing a specific paradigm:
    - backpropagation
    - radial basis functions
    - Boltzmann machines
    - Kohonen networks, etc.
  - emulating special biological circuits or functions:
    - retina
    - cohlea
    - visual flow computation or filtering circuits, etc.
- The partitioning of the neural network per processing element:
  - synapse
  - neuron
  - subnetwork
- Type of learning
  - on-chip learning
  - off-chip learning
  - chip-in-the-loop learning

Figure 1: A taxonomy of neural network hardware

1.1.1 Implementation type

The implementation type refers to the type of signals processed in the circuit. A circuit can use a purely analog approach in which all processing and signals are analog, a purely digital approach in which all processing and signals are digital and finally, a mixed approach in which there is some processing or information of each type. A typical example of a mixed signal approach is the use of a local digital memory for storing the values of the weights. Such digital weight values can use just one [78, 189] or several bits and a D/A converter [151, 46, 91].

Optical implementations of neural networks use optical encoding and processing of information. Optical implementations can use multiple coherent optical beams to represent signals, holographic gratings for interconnections and non-linear optical or optoelectronic arrays for processing [82, 193, 150]. An alternate approach uses incoherent light and optical cross bars for interconnections. The weights can be implemented as a spatial light modulator or a photographic transparency mask [66]. A parallel optical associative memory system with feedback was implemented with holograms in [170]. Jet Propulsion Laboratory has an optoelectronic neural group that aims to develop an optical implementation of Fukushima's neocognitron [32]. Other efforts to build optical or optoelectronic implementations include sophisticated architectures such as ART-1 [199] and parts of ART-2 [102].

Among the advantages of such implementations one can mention the ability to process in parallel various signals and the high processing speed. Comparisons between the performances of optical and VLSI networks are presented in [67] and [106].

Pulse-stream implementations of neural networks were proposed in 1987 by Murray and Smith [144, 145]. In this type of implementation, the information is carried by trains of pulses. In turn, pulse-stream neural networks can be subdivided into circuits using pulse amplitude modulation, width modulation, frequency modulation and phase modulation. The characteristics of such implementations include immunity to noise, good inter-chip interfacing properties, the signals are easily regenerated, etc. The precision is, as in all analog systems closely related to the area, as will be discussed in Section 2.5.1. This type of implementation can be seen as combining the advantages of the digital and analog approaches by processing in analog hardware a digitally coded signal (pulses). For an in-depth treatment of pulse coded neural networks one could refer to [146, 203] and references therein.

The RAM-based implementations are based on the N-tuple method introduced in 1959 by Bledsoe and Browning [20] and was exploited by Aleksander [2, 3, 1]. The idea behind this approach is that generalization does not necessarily come from a non-linear processing but can also be obtained by simply associating unseen patterns to similar patterns in the training set. The patterns in the training set are simply stored in a RAM memory during the training. Different variations of this idea include multilayer RAM networks, reinforcement learning using tri-state storage in probabilistic logic nodes (PLN) [101], the Goal Seeking Neuron (GSN) [69] and probabilistic RAMs (pRAM) [76, 77, 36].

1.1.2 Interneuron communication

This criterion refers to the way neurons are organized and communicate to each other. In most cases, the network implements a fixed architecture in which neurons communicate with all, as in a fully connected network, or a subset of the remaining neurons. If a fully connected architecture is used, the number of synapses increases quadratically with the number of neurons. If a layered architecture is used, each neuron will communicate only with the neurons on the layers up-stream and down-stream from its own layer. There are also architectures in which neurons are simply arranged in a matrix in which each neuron communicates only with a few neighbours. Such architectures are used in certain self-organization paradigms, cellular neural networks, etc.

A very interesting solution is that of a reconfigurable chip [78, 22, 116, 161]. In such a chip, not only the values stored in the weights of the circuit are adjustable but one can also change the connectivity of the individual units. In [161] for instance, the reconfigurability of the network is achieved by using a matrix of NxN neuron-synapse pairs and a matrix of switches that can be used to configure the network. The switch matrix also contains shift registers for storing the configuration information.

A similar idea is presented in [116]. This implementation uses two cascadable chips: a neuron chip and a synapse chip. The neurons on the neuron chip can be interconnected as desired using the synapses on the second chip. The synapse test chip presented used a matrix-vector multiplier with variable, 10 bit resolution for the matrix elements.

A reconfigurable chip has the advantage that it provides a general purpose hardware which can be

adapted to various specific configurations that may be required in different problems. From the VLSI point of view, reconfigurable circuits have the advantage that defects of the circuit can be neutralized by a suitable configuration. Also, such a chip is easily tested because any internal node can be made accessible. The price paid for these positive features includes the higher complexity of the circuit and the redundancy that appears if less than a fully connected network is deployed.

1.1.3 Architecture

From the point of view of the system's architecture, hardware implementations of neural networks can be divided into integrated circuits with on-chip controllers, stand-alone neurocomputers and accelerator boards. A neurochip can perform all the computation on the chip. They are fed the input signals and they produce the desired output. Stand-alone neurocomputers do the same only that they are realized with several integrated circuits and the information processing cycle includes interchip communication. However, the whole processing is performed in the neurocomputer. Accelerator boards are designed to be used in conjunction with some other processor(s) in order to accelerate specifically the processing of the neural network part. In such cases, there is often some sort of preprocessing in the system before the data is presented to the neural hardware. Sometimes, the processing involves some dialog between the neural hardware and the rest of the system.

Another architectural issue is the approach used to increase the processing capacity. Different methods such as wafer-scale integration [132, 174], multi-chip modules [100, 180, 186] and die stacking [56, 165] have been proposed. The die stacking approach for instance, uses several neural networks integrated circuit planes stacked vertically in a unique circuit. The inputs can be controlled by sequencer circuits and switching matrices. Each plane can be programmed separately to process the same input in different ways or to process several disjunct parts of the input in the same way. The production yield which is normally a major problem for circuits with a large number of elements can be addressed in this case by testing each plane individually, before packaging.

1.1.4 Degree of parallelism

The number of processing elements in a hardware implementation can be used to characterize the degree of parallelism achieved. The boundaries between different categories can probably be set more or less arbitrarily. A reasonable classification could divide neural network implementations into low parallelism implementations (less than 10-15 processors), medium parallelism implementations (10-10^2 processors), highly parallel implementations ($10^2 - 10^3$ processors) and massively parallel implementations (more than 10^3 processors). The degree of parallelism that we defined here as the number of processing elements cannot be compared directly to the degree of parallelism used in the context of general purpose digital systems due to the very different capabilities of a single processing unit. Thus, in our context, a processing unit implements a simple weighted sum of its inputs and a non-linear transfer function. In the context of parallel computers, a processing unit is usually a full microprocessor able to do general purpose computation. Because of this, it would be rather misleading to characterize a digital parallel machine using 10 microprocessors as having a low degree of parallelism.

Therefore, our notion of degree of parallelism and the classification suggested above should be used only in the context of neural systems. On the other hand, it is perfectly acceptable to compare different types of neural network implementations (such as analog, digital, optical, etc.) using the classification suggested above.

1.1.5 Purpose

Another criterion useful in the classification of neural network implementations is their design purpose. Going from the most specific to the most general, various implementations can be divided into:

1. Circuits emulating specific biological circuits or functions. Examples of such implementation include retina [133, pp.239–246], [130, 198, 21, 44, 19] and Chapters 5–7 in [114], cohlea (see Chapters 1–4 in [114]), edge detection [11], etc.

2. Circuits implementing more complex functions integrating several circuits or functions from above. Examples here could include the large range of vision circuits (see [75] and Chapter 12 in [165] for some examples), character recognition [155], etc.

3. General purpose circuits that use a specific neural paradigm or architecture (e.g. backpropagation [137, 188, 166, 138] and Chap. 10

in [165]), radial basis functions [194], Kohonen networks [115, 129, 85, 87], bidirectional associative memories (BAMs) [123, 124] and learning vector quantization (LVQ) [127, 29], etc.)

4. Model-free general purpose circuits (fixed architecture but free from a specific neural or architectural paradigm e.g. the Tarana chip [96, chapter 8] and the whole approach of weight and node perturbation or stochastic learning [97, 197, 71, 28, 5])

5. Reconfigurable circuits (flexible architecture able to be used with various learning techniques [78, 22, 116, 161].

1.1.6 Types of learning

A hardware implementation of a neural network can be trained in a supervised manner (in which the desired output is known for each input pattern) in several ways. In the case of *on-chip* learning, the weight updates are calculated and applied on the chip. This particular approach has the disadvantage that training circuitry (e.g. error calculation and backpropagation) has to be provided in the structure of the chip even if the training is infrequent. Furthermore, various researchers have reported problems related to the precision available on the chip in the case of training algorithms such as backpropagation [90, 86, 201, 179].

The *off-chip* learning performs all computation off the chip. Once the solution weight state has been found, the weights are downloaded to the chip. Due to the difference in the precision used on the training machine and the precision used on the chip, some truncation may be necessary when the weights are downloaded. This approach has the advantage that the training is performed off-chip and with a high precision but has the disadvantage that the actual performance of the chip might be poor after the weights are downloaded. Such poor performance can be caused by the weight truncation and the chip's non-idealities.

Finally, in the *chip-in-the-loop* situation, the errors are calculated with the output of the chip (thus taking into consideration all imperfections of the particular chip trained and the effect of the lower chip precision) but the weight updates are calculated and performed off the chip. The weights on the chip are updated periodically according to the learning strategy used (batch, on-line, etc.) [96].

1.1.7 Other issues

We should also mention more specific issues related to particular types of implementation. In the case of digital implementation for instance, issues such as weight representation (floating point, fixed point, integers, etc.) can be of particular interest. Another important issue is the weight resolution (bits per weight) and/or the accuracy of the computation [55, 86, 104, 105, 110, 111, 51, 52, 50, 53].

We would like to discuss briefly the terms "accuracy" and "precision" since they are central in any comparison of different implementations. The two terms seem to be synonymous in the common use of the English language. Thus, [183] defines them both as "degree of refinement in measurement". However, some researchers tend to make a distinction between the two terms when they are used in a hardware implementation context. *Accuracy* can be used to refer to the closeness of a measurement or result to the appropriate value. *Precision* can be used to refer to the ability of reproducing the results when experiments are repeated in similar circumstances. If such distinction is made, digital systems can be considered precise, since they will always produce the same results in the same circumstances. However, digital systems will be considered accurate only to the extent to which they have enough digits to represent exactly the appropriate value. On the other hand, analog implementations are potentially accurate because they are able to produce any specific value within their range. However, analog circuits can be considered imprecise since they are unlikely to produce the same results in different experiments [1].

Many of the papers reviewed here do not make such distinction between accuracy and precision. Furthermore, one can find very often implementations described as having "an accuracy of 6 bit per synapse" or "a storage capacitor with 5 bit precision". The confusion can increase further if the context is that of a circuit in which the weights are stored in digital form and the processing is done in analog form. In order to eliminate any potential confusion, we will refer to such cases as *weight resolution*. The meaning of this is that the weights are stored as binary numbers represented using the number of bits specified, without any implication as to whether desired values can be actually reached or whether the results can be reproduced.

[1] The distinction between these terms has been suggested by one of the anonymous reviewers. We would like to thank the reviewer both for pointing out a very common source of confusion and for suggesting the clarifications adopted here.

2 Analog hardware

2.1 Analog versus other types of implementation

Both analog and digital types of implementations have been investigated by many researchers. Each type has its own advantages and disadvantages. We have decided to concentrate on analog hardware because we believe that such implementations have important advantages over other types of implementations. These advantages include:

- The ability to process several bits of information per processing element
- A very compact circuit, inherently parallel and with a high degree of interconnectivity
- Very convenient implementation for summation, weight multiplication, sigmoid nonlinearities, etc.
- Very low power consumption (especially if using CMOS elements in weak inversion)
- A high speed due to asynchronous updating
- Analog hardware is easy to interface in many real-world applications (no need for A/D and D/A converters)

However, analog implementations have certain clear disadvantages as well. Several such disadvantages can be mentioned:

- Susceptibility to temperature variations, electrical noise, crosstalk, power supply variations, etc.
- Some analog implementations are not easily programmable
- The precision is often limited by area considerations
- They exhibit several trade-offs related to noise immunity, power consumption, area, etc.

Various issues related to the trade-offs mentioned above will be discussed in section 2.5.

When trying to compare analog implementations with other types of implementations, it is interesting to consider the relative importance of the advantages and disadvantages mentioned above.

We believe that a crucial advantage of analog implementations is the fact that they can be easily interfaced with a large set of sensors which, by their very nature, are analog. Analog implementations will continue to be attractive as long as analog sensors will be used. Analog implementations of neural networks offer a very compact and elegant solution for the pre-processing or processing of the signals coming from such sensors and in some applications they will probably never by replaced by other types of implementations.

Another important advantage is their overall potential for compactness resulting from the combination of the first three advantages mentioned above. The basic operations used in neural networks, summation and multiplication, can both be implemented with few transistors in an analog circuit. If we consider that the transfer functions of the neurons can be implemented directly by physical phenomena, analog circuits have a very good potential for realizing very compact circuits. The combination of the factors mentioned above make analog implementations the natural choice for systems that require simultaneously a high parallelism, a high processing speed, low power consumption and an interface with analog sensors. Typical applications might include artificial vision systems, navigation systems for autonomous robots, implantable biomedical devices, etc.

On the other hand, the inherent lack of precision (in the sense of reproducibility of the results) can be somehow compensated by the distributed nature of the neural architectures in general.

Some purely theoretical arguments in favour of analog circuits can also be made. The main thrust of such arguments is that analog circuits allow continuous variations whether digital circuits allow only discrete variations. This is rather intuitive: when moving from real to rational weights, some computational power is lost. In a somehow extreme comparison along these lines, [190] investigates the properties of analog and binary (i.e. having only $+/-1$ weights) networks. The paper shows that for some problems, if an analog network is used to optimize an objective function then every local maximum has a non-trivial basin of attraction, and conversely, the only equilibria that are attractive correspond to local maxima of the objective function. The paper also shows that for binary networks, all global maxima of the objective function have a zero radius of attraction and it is possible to start the training in a state adjacent to a global maximum and yet fail to converge to that maximum. Several papers studied the effect of reducing the weight resolution upon the learning process [55, 90, 128, 135, 152]. Although such papers have demonstrated that in certain conditions,

one can cope with a reduced weight resolution, it seems clear that analog circuits offer an elegant and inexpensive implementation of properties like continuity and differentiability that are at the core of so many neural paradigms.

There is also at least one advantage of analog implementations that has started to lose its importance. This fading advantage is the higher speed of analog circuits. Due to their wide spread use and the huge development efforts characteristic to the last few years, digital circuits have become so fast that the - once huge - speed gap between digital and analog circuits is on the verge of oblivion.

2.2 Neural networks implementations in analog hardware

The interest in hardware implementations of neural networks started practically at the same time with the field of neural networks itself. Pioneers of neural network research such as F. Rosenblatt and B. Widrow not only designed fundamental learning mechanisms but also implemented them in hardware. Rosenblatt's Perceptron and Widrow's Memistor [196] are early instances of hardware neural networks.

The beginning of a considerable effort towards hardware implementations of neural networks should probably be dated in the late 80's and followed closely the wave of renewed interest in the general field of neural networks. An initial direction justified by the relative lack of practice with neural network paradigms was to implement in hardware dedicated architectures directly inspired from biological information processing devices such as the retina [130, 167], the cochlea [117] or a combination of both as in the seehear chip [134]. A related interest was directed towards designing chips that implement certain functions such as calculating the optical flow [65] or detecting motion [178].

More or less at the same time, major research centers became interested in the neural network technology and started to design dedicated chips based on neural networks ideas. In 1986, AT&T Bell Laboratories built an analog VLSI chip in 2.5 μm CMOS technology with 256 neurons and 100,000 synapses. The chip was based on a content addressable memory idea, had an area of 5.7x5.7mm and its main purpose was to compress bandwidth for video images transmission over telephone lines [80]. In 1989, Lee and Sheu at University of Southern California proposed an A/D converter based on a Hopfield network [118].

Implementations aimed at a more general neural processing started to appear in the same period. The initial amorphous-Si implementation used by AT&T Bell Labs researchers could be programmed only once, using electron beams during the fabrication of the chip. This prevented it from being used in a general purpose chip. Re-programmable devices such as DRAMs and EEPROMs were much more suited for such an application. In 1989, Intel researchers presented a general purpose "electrically trainable artificial neural network" with 64 neurons and 10,240 synapses realized in EEPROM CMOS technology. This chip was a general purpose device implementing a single layer network with feedback and using a precision between 4 and 8 bits [89, 27]. The ETANN (Electrically Trainable Analog Neural Network) chip is available as a commercial circuit (80170NW) from Intel.

In 1990, Matsushita proposed a DRAM based, general purpose VLSI chip with 64 neurons and 768 synapses. The circuit was realized in 2.2μm BiCMOS technology and had a chip surface of 18x13.5mm [138]. The chip implemented the back-propagation algorithm but had a 5% error at the multiplication level. Another DRAM based, general purpose neural chip realized in 2.0μm CMOS technology and integrating 64 neurons and 4,096 synapses on a surface of 4.6x6.8mm was propose by Lee and Sheu [119]. The chip was a general purpose device with an architecture of a single layer, fully connected and using an 8 bit weight resolution.

In 1990, Bell Labs reiterated the idea of a neural chip, this time aiming for a reconfigurable device [78]. This chip was a general-purpose, SRAM based neural network with 256 neurons and 32K synapses. It was designed in the 0.9μm CMOS technology with an chip area of 4.5x7mm.

The same year, [91] described a digital memory implemented in 1.25μm CMOS technology. This chip had a single layer, fully connected architecture with 81 neurons and 6,561 synapses and used a 6 bit weight resolution.

In 1991, Mitsubishi implemented an unsupervised learning chip in 1.0μm CMOS technology [9, 10]. The chip had a one layer, fully connected architecture with feedback. Its 10,240 synapses used capacitors with a precision limited to 1 bit and the chip reached a speed of 5ms per pattern. In the same year, Lockheed unveiled a neural network implemented as a board with discrete elements [70]. This network was designed for pattern recognition and had a single layer, fully connected architecture with 2,048 5 bit resolution synapses.

Three implementations which followed the idea of a reconfigurable network presented above [78]

are described in [22, 161, 46]. The device presented in [22] was realized at the AT&T Bell Labs and was designed for character recognition. This device could be used in single layer, multi-layer or Hopfield configurations and had a resolution of 6 bits per synapse and 3 bits per neuron. The chip was implemented using 0.9μm CMOS technology and had 4,096 synapses and between 16 and 256 neurons. The circuit was able to provide a processing speed of 5 GCPS (giga connections per second). Satyanarayana [161] built a general purpose neural network implementation using 0.9μm CMOS technology. This circuit provided 1,024 neurons and 1,024 synapses organized on one, two or three layers. The circuit used reconfigurable dynamic capacitors which could be refreshed in 74msec. Spiegel [46] presented a neural computer realized in 2.0μm CMOS technology and using chip-modules and digital memory. Using 100 such chip-modules, Spiegel built a network with 72 neurons and 2,466 synapses with a 6 bit resolution. This neurocomputer was reported to achieve 1011 FLOPS (floating point operations per second).

In 1992, Jet Propulsion Laboratories developed an analog neural network chip [24]. The chip has two versions. One version has a 32 neuron fully interconnected architecture. The neurons are placed on a diagonal of a 32x32 matrix of synapses. The processing is asynchronous analog and fully parallel with a giga-connection per second capability (1,000 MCUPS). The second version of the chip contains only synapses. These chips can be cascaded together to make larger configurations. The synapse design is based on a static random access memory (SRAM) with 7 bits (6-bit + sign bit) of resolution. The chip uses current multipliers for synapse processing. The neurons can implement wide-range, variable slope sigmoids. Due to the bounded range and finite resolution of the weights, the chip is trained in a chip-in-the-loop manner.

High speed devices were proposed by Toshiba [166] and USC [33]. The Toshiba implementation was realized in 0.8μm CMOS technology and used two different chips, one for the synapse matrix and one for the neuron array. The architecture was designed to implement backpropagation and hebbian learning. The chips formed a network with 24 neurons and 576 synapses and used an 8 bit resolution. The paper reports a speed of 36 GOPS/480 neurons. The chip built at USC was realized in 2.0μm CMOS technology and had a single layer, fully connected architecture with 32 neurons and 960 synapses. The synapses were able to implement an 8 bit precision computation.

As the technology matures, several analog neural network implementations have reached the stage of successful practical applications. A few such successful applications will be mentioned in the following.

Coggins et.al. implemented a micro-power intracardiac electrogram classifier that uses an analog neural network to classify arrhythmia [38, 37, 96]. The network is a 10:6:3 multilayer perceptron with on-chip digital weight storage. The chip is implemented in 1.2 μm CMOS, occupies an area of 2.2x2.2mm^2 and consumes less than 200nW from a 3V supply.

Intel manufactured a board using an analog VLSI neural network for impact signal processing [23]. The board uses a discrete Fourier transform to preprocess an accelerometer output waveform that is subsequently recognized through a multilayer perceptron. Both the discrete Fourier transform and the NN are realized with 80170NX (ETANN) circuits. Due to a response time of approx. 15 μs, the device is suitable for deployment in applications like intelligent airbags and vibration cancellation systems.

Another successful application that uses the ETANN chip is reported in [42]. In this application the chip is used to implement linear and non-linear controllers for smart structural systems. The capabilieties of the neural network are used to embed a priori information on the smart structural systems such as sensor bandwidth limits and control effort limits.

An example of the latest design achievements in neuromorphic circuits is the low-power wide-dynamic-range analog VLSI cochlea [160] from Carver Mead's group. Particularly interesting is the comparison between the analog and digital cochleas. Although the analog one is implemented in 1.2μm technology and the digital one might use a 0.5μm technology[2], the analog implementation is 300 times more efficient in power consumption, 3 times more efficient in area comparisons and offers a dynamic range of 61dB on a 100Hz to 10kHz frequency range.

A team from Jet Propulsion Laboratory and Irvine Sensors Corp. has developed a 3-dimensional neural network circuit aimed at target recognition for missile interception application (see Chap. 15 in [165]). The circuit has several planes and each plane is a reconfigurable multi-layered perceptron with a total of 4,864 seven-bit programmable

[2]The digital cochlea was not implemented. The power consumption and area are calculated (as opposed to measured) for the digital implementation

synapses, 70 sigmoidal neurons and auxiliary circuitry. In 1998, Irvine Sensors won a US army contract to demonstrate the feasibility of its 3D Artificial Neural Network(TM) technology.

Another successful development of analog hardware is the ANNA neural network board from AT&T Bell Labs [155, 156]. ANNA contains two analog VLSI chips, four Xilink 4005-55PG156 field programmable gate arrays (FPGA) chips, memory and interfaces. The board is able to perform tasks such as character recognition, text location, segmentation and filtering. The neural network chip contains 180,000 transistors on a 4.5x7mm^2, implemented in a 0.9mum CMOS technology. The circuit needs an external controller to form a usable system. The ANNA board is able to process 2 frames per second on 512x512 images at a processing speed of 2 billion connections per second.

Bell Communication Reseach developed a neural network co-processor board using analog neural network circuits [6, 98, 99]. The board can contain up to six neural network chips. The chips support mean-field annealing and Boltzmann learning for on-chip weight changes. The weights are stored and updated digitally and use a 5 bit resolution. Each chip contains 32 neurons and 496 synapses and is able to do 100 million CUPS. The annealing time can be varied from 10 to 100μs depending on the desired accuracy. In recall mode, the chip can work at 3μs per pattern.

A chip targeted at high-energy physics research applications was proposed by Masa [131] and is presently available as a commercial product. The chip can classify vectors of up to 70 components within 50ns. The network architecture has 70 inputs, 6 hidden and 1 output neuron. The weights are stored digitally using 5 bit resolution. The chip was fabricated in 2.5μm technology, can perform 6 billion multiplications per second and consumes 2W.

Synaptics Inc. in Santa Clara, Ca has produced the I-1000 chip, an industrial version of Carver Mead's silicon retina for the recognition of the numerical characters on the bottom of US bank checks. The circuit contains 20,000 synapses and the power comsumption is in the mW range. An interesting feature of this product is that the weights are hard-wired in the network. This is possible because the character set is known in advance and not likely to change.

A group of researchers from Univ. of Pennsylvania and Corticon Inc. has developed a "large scale programmable analog neural computer" [64, 139, 47]. The architure contains 1024 neurons each with 96 inputs and a total of 98,304 synapses. The device was developed for temporal pattern analysis in real-time. The network has been tested in speech recognition, image analysis, etc. The device is particularly suitable for any task that requires the incorporation of programmable time constants. It supports 6,656 time constants ranging from 0.4ms to 1s with a 6 bit resolution. The transfer function of the neuron is selectable between a linear, step or stepwise linear function. The operating speed of the neuron can vary between 16 settings from 1MHz to 8Hz. This device is available commercially, as well.

As opposed to describing a certain architecture or chip, some papers concentrated on issues which are common to certain analog technologies or approaches very useful or specific to neural network architectures. Some problems related to analog storage are presented in [88, 93, 192]. Some non-volatile analog storage technique and circuits are presented in [109, 169, 103, 121, 57, 18, 73, 184, 83]. Some volatile analog storage techniques are presented in [163, 87, 26, 31, 29, 58].

Initially, designers of analog implementations used voltages to represent neuron excitation and output values, operational amplifiers to implement the neurons and resistors or transconductances to implement the weights. Papers such as [81, 181, 185] present such designs. Some of the difficulties introduced in such circuits by less than ideal parameter values such as the output impedance of the opamps and the reciprocal influence of the weights are analyzed in [15]. Other problems introduced by less than ideal components and several error models are discussed in papers such as [168, 72, 143, 25, 48].

The most important analog hardware properties taken in consideration in such studies include variations in multiplier gains (both due to noise and due to fabrication), function approximations (bounded signal values, non-linearities of multipliers, approximate implementations of the sigmoid, etc.), variations in multiplier zero offsets and weight decay.

The problem of weight accuracy is particularly important since it affects various types of implementations (thermal effects and weight decay in analog storage, weight quantization in digital storage, etc.). Several papers concentrate on weight imprecision issues [8, 140, 200, 201, 152, 182].

Weight imprecision and component variation have sometimes been modeled as noisy weight changes [72, 140]. One particularly clever approach in dealing with such non-idealities is to use an ubiquitous factor - the noise - to actually enhance

the reliability of the circuit [92, 141, 142, 143].

2.3 Analog hardware implementations of specific paradigms

As the amount of theoretical knowledge about neural systems increased, so did the efforts to apply advanced ideas directly in hardware. Such efforts included algorithms designed for analog VLSI implementation of specific techniques such as backpropagation [187] or actual chips tailored to specific training techniques. Some learning techniques seem to be particularly interesting from a hardware point of view because some mechanisms can be modeled by physical phenomena. Such technique could include simulated annealing [120] and Boltzmann machines [112, 4]. However, hardware implementations have been proposed for most major neural learning paradigms such as radial basis functions [194], Kohonen networks [115, 129, 85, 87], bidirectional associative memories (BAMs) [124] and learning vector quantization (LVQ) [127, 29].

One particular technique which has had a major contribution to the renewed wave of interest in the neural network field is error backpropagation. Several papers present analog chips dedicated to backpropagation [137, 188] [166, 138] and Chap. 10 in [165].

A neural network paradigm that appeared very suitable for analog VLSI implementations since its very beginning is the cellular neural network (CNN) paradigm [35, 34]. CNNs are very attractive for such implementations because they use analog processing cells with continuous signal values arranged on a regular grid and interacting with each other only within a finite radius. It has been shown that one can build a universal computing machine using CNNs [153, 40, 147]. However, it is more convenient to use CNNs for specific tasks because if the template that defines the local processing is fixed, one can design very efficient implementations. CNNs have been used for character recognition [175, 173], the implementation of a bionic eye [195], object oriented video coding [171, 172], concurrent image aquisition and processing [59, 62], image compression [176], optical detection of breaks and short circuits on printed circuit boards [177], etc. Various issue related to VLSI implementation of CNNs can be found in [59, 63]. Other CNN implementations are presented in [41, 49, 60]. The circuit presented in [60] is designed for parallel acquisition and concurrent analog processing of two-dimensional (2-D) binary images. The weights defining the processing are stored with a 7 bit resolution and the internal processing is fully analog. The chip also features a small on-chip RAM memory for storing processing parameters and a fully digital external interface. The chip was manufactured in a 0.8 μm single-poly double-metal technology and is able to process an image in approximately 2μs. The chip presented in [60] implements a programmable cellular neural network with functionalities similar to those of the theoretical CNN machine described in [153]. The circuit contains 1024 processing cells and is implemented in 1.0 μm n-well CMOS technology. It offers around 7-8 bits resolution for the weight values and an average density of 31 cells/mm^2.

2.4 Trends and directions in analog hardware

It is always very challenging to try to identify trends and directions in a field that is characterized by an impressive variety of approaches and technologies. Due to the fast evolution of integrated circuit technology it is both risky and of limited value to try to select the particular technological solution for a specific problem (e.g. weight storage). What might have been successful in the past few years might be made obsolete very soon by new developments. Therefore, we will concentrate to those higher level trends that we expect to continue in the future.

The most noticeable trends are:

- A tendency towards increased flexibility. There seems to be a trend from fixed weights towards programmable weights and from fixed architecture to configurable or-reconfigurable architecture;

- The way imperfections are treated. Initially we tried to ignore them, then they started to be taken into account (e.g. chip-in-the-loop learning) and now researchers are starting to use them (e.g. use the noise to improve reliability);

- The development of a large number of theoretical training algorithms being inspired or improved by VLSI realities (e.g weight perturbation, noise-enhanced learning, etc.)

- A trend towards scalability and connectivity. A recent interest in the ability to interface such neural implementations with other circuits (either of the same kind or different).

Another interesting comparison can be made between the parallel development of software vs. hardware implementations. After the re-birth of the neural network field in the late 80s, the most popular technique in software simulations was definitely the error backpropagation. However, hardware implementations reflected more accurately the great variety of theoretical paradigms that have been proposed. Perhaps somehow surprisingly, full implementations of backpropagation are quite rare and appeared rather late. We believe that there are several reasons for this. On the one hand, backpropagation benefits from high precision computation more than other techniques. This would explain why software simulations were so diffused. Fully fledged analog implementations of backpropagation have been much more rare than software simulations of the same because the imprecision of the analog hardware was seen initially as a handicap against obtaining good results with backpropagation.

Another promising approach is that of the mixed-signal implementations. Such implementations can combine some of the advantages of the digital and analog approaches. This can be done in at least two different ways. A first approach is to combine the two signals on the chip itself (e.g storing the weights digitally, using analog signals to code for binary numbers, etc.). This approach might offer valid solutions for combining the advantages of the two approaches. Although not included in this review, the very successful pulse-coded neural networks would be an example of such an approach. The second approach would use a purely analog circuit but embed it in a system with a digital interface. This approach has been chosen by many of the successful applications mentioned above. This is required by the overhwhelming dominance of digital circuits in modern technology. We expect both types of mixed signal approach to expand greatly in the future.

Other approaches that are very promising for future development include programmable synapses, gain-adjustable neurons and reconfigurable networks. The flexibility offered by such approaches can compensate for device mismatches, limited weight precision and imprecise transfer functions which may continue to be a characteristic of the analog hardware.

2.5 Design and implementation issues in analog hardware

In this section, we shall discuss briefly several issues related to analog VLSI implementations of neural networks. In this discussion, we will follow the lines of [96]. A detailed discussion about the area and time efficiency of VLSI implementations can be found in [16, 68].

As it can be noted from the implementations described above, the most widely used fabrication process for VLSI systems is the CMOS technology. The CMOS fabrication process provides the means to fabricate N and P type MOS active (transistors, diodes) and passive (capacitors and resistors) circuit elements. Some processes also allow the implementation of floating gate transistors and floating capacitors.

Due to its fabrication process, the CMOS technology embeds several intrinsic trade-offs.

2.5.1 Precision-area trade-offs

The chip area occupied by a circuit element is related in several important ways with parameters such as precision and speed both at the component as well as at the circuit and system design levels.

For instance at the component level, we can consider the example of an implementation that stores the weights in capacitors. The area of the capacitor is directly proportional to the amount of charge that can be stored in such an element. If the area occupied by a given capacitor is increased, the errors introduced by the fabrication process (which are independent of the size of the component being fabricated) will represent a smaller percentage of the total area of the component. When this element is replicated in various parts of the circuit and in various exemplaries of the same circuit, there will be a smaller percentage variation for elements with a larger area. In consequence, such elements will provide a better precision.

Another example of the precision-area trade-off at the component level is the very widely used differential amplifier. The correct functioning of this circuit depends on the two transistors of the differential pair having the same characteristics. However, fabrication process errors determine small differences in the surface occupied by the two transistors and these differences translate into different gains. The smaller the surface occupied by such transistors, the larger the percentage error introduced by the fabrication process and reciprocally, the larger the area, the smaller the error and hence, the better the precision.

Although the above precision-area trade-offs are always present, there exist several techniques that can reduce the severity of the problems mentioned. Most such techniques are related to the layout of the circuit and will be briefly mentioned in Section 2.5.5.

At the circuit level, the area-precision trade-off is even more apparent since sometimes, the precision of the circuit is directly proportional to the number of components. For instance, the precision of a sigmoid implemented by piece-wise linear approximations, as in [16], will be directly proportional to the number of intervals used in the linear approximation. Each interval will be implemented by a certain (constant) number of components which will require a certain constant chip area. From here, it follows that the precision of the sigmoid implementation will be directly proportional to the area. While the example of the sigmoid can be rebutted quite easily (e.g. the sigmoid can be implemented more elegantly as a component nonlinearity), the area-precision trade-off remains an issue to be considered when designing neural circuits.

The relationship between area and the precision appears also at the system design level. For instance, if the neural network is used in a function approximation application, the precision of the approximation will be directly proportional (up to a certain point) to the number of neurons in the network and therefore to the area of the chip.

2.5.2 Speed-area trade-offs

The relationship between the area of a circuit and its speed of operation is one of inverse proportionality. In general, the smaller the chip, the shorter the paths between different circuit components and therefore, the less time needed for signals to propagate from one element to another. Also, the smaller the circuit, the smaller the parasitic capacitances and the faster the signal propagation through the circuit.

Another factor that influences the precision-area trade-off is the thermal noise. The thermal noise increases with the size of the circuit. This thermal noise will reduce the overall signal-to-noise ratio and will therefore reduce the positive impact of the increased precision obtained through a larger area.

2.5.3 Power-precision trade-offs

The power consumption of a circuit is directly proportional to the speed at which the circuit operates. A large percentage of the power consumed is dissipated as heat during the normal operation of the circuit. For a given speed and circuit architecture, the efficiency of power dissipation decreases with the size of the chip. Of course, the efficiency of the power dissipation increases with the temperature difference between the circuit and the environment. However, operating at high temperatures, deteriorates the signal-to-noise ratio due to an increased thermal noise and reduces the life of the circuit. For high density, high speed VLSI circuits, the power dissipation problem is non-trivial.

One commonly used approach to reduce the power requirements is to lower the operating voltages. However, an immediate consequence of this approach is to reduce the dynamic range of all signals in the circuit and hence, the precision of the circuit.

Some problems arising from power dissipation issues are particularly important for analog circuits. For instance a gradient of temperature can modify the characteristics of paired transistors, the thermal noise can reduce the operating range of individual components and deteriorate the signal-to-noise ratio throughout the circuit, etc. Such problems are somehow less important in digital or pulse-stream approaches when the signal can be regenerated by individual logical components (e.g. gates or neurons).

Thinking along these lines, it is probably no coincidence that the use of MOSFET transistors in weak inversion was pioneered in analog design [191, 134]. This approach is very interesting from a power consumption point of view because in subthreshold regime, a transistor will operate with typical currents of the order of 10^{-9}A. However, in weak inversion, the problem of noise resurfaces and needs particular attention. Furthermore, in weak inversion, the bandwidth of the transistor is reduced, which limits somehow the utility of a single transistor.

2.5.4 System design trade-offs

The performance and the design of a neural network VLSI chips are also influenced by the interface between the chip itself and the rest of the system in which the chips functions.

A first issue is that of the packaging. The overall computational capabilities of a chip depend not only on its intrinsic capabilities but also on its ability to transfer the information over its I/O terminals. Therefore, the higher the computational power, the higher the I/O bandwidth required. This of-

ten contrasts with the requirement of a relative low number of pins.

Another issue is the communication with the rest of the system. Usually, the communication interface requires multiplexers and buffers which contribute to increasing the area and the power consumption of the circuit. Also, in order to obtain a good signal-to-noise ratio, the voltage need to be maintained to a certain minimum level at the level of the whole system. As discussed above, if the same relatively high power supply voltages are used on the chip, the area and power consumption increase. On the other hand, if we use a different internal level for the power supply, more elements are needed for the chip to board interface.

Also, the testability of the circuit require external access to its internal states. In most cases, this requires multiplexing some signal paths at the level of the circuit pins which again requires more interface dedicated chip area. In analog circuits, one can even contemplate adding some redundant components and/or using (at least partially) reconfigurable systems to alleviate this problem but of course, these approaches need circuit area as well.

2.5.5 Layout issues

The layout of a circuit can affect significantly the performance due to factors such as component matching, sensitivity to process variations, noise injection from power lines, clock lines and the substrate, clock feed through, etc.

Apart from component matching, all factors above can be improved by minimizing the parasitic effects present in the circuit. Such effects can include leakage currents, capacitive coupling, series resistance and long distance coupling. These problems can be addressed by first identifying the components affected and then minimizing the parasitics or compensate for them with dummy structures. Classical techniques used in minimizing parasitic effects include shielding using grounded lines or polysilicon layers, having wide supply lines, using separate power lines for the digital and analog parts of the circuits in a mixed circuit, etc.

Component matching is critical to the correct functioning of some circuit elements. For instance, differential amplifiers rely on paired transistors with equal gain. Matched capacitors and/or resistors is array configurations are used in D/A and A/D converters or in constructing transfer functions (such as the sigmoid) through piece-wise linear interpolation.

[96] presents a good summary of the techniques used to minimize the effect of component mismatch and the other issues enumerated above. A few techniques used to improve component matching are as follows:

- Matched components should be realized using the same type of structure, same shape and same size. This offers more resilience to fabrication process parameter variations. Also, the size has a direct influence upon the parasitic capacities introduced.

- Matched components should be located as close to each other as possible and placed in the same orientation. This minimizes the effect of surface or process non-uniformities.

- Locate components in regions of equal temperature i.e. perpendicular on forseeable gradients of temperature. This technique minimizes differences determined by the temperature dependency of the drain current in the MOS transistor.

- Place components in the same surrondings (e.g. by adding dummy elements). Again, this influences the parasitic capacities formed between the components and their surrounding elements and the leakage effects.

2.6 Learning in analog hardware

Although analog implementations have many advantages when compared to digital ones, they also have some characteristics which can raise problem during the training. One such major issue is the precision of the weights. If they are stored as analog values, the weights have a limited precision. The limitation comes from various factors as discussed above.

On the other hand, most neural network training algorithms have been derived theoretically and then simulated in software where such limitations do not exist. A natural question is whether such algorithms will converge equally well (or at all) in a hardware situation in which the precision is limited and there are also imperfections which determine computational errors. In general, it has been shown that the learning process can indeed be successful in spite of the various implementation problems [152, 72, 25, 48]. It has also been shown that in some cases, hardware related imperfections such as synaptic noise can actually have a beneficial effect both on the learning itself and on the generalization abilities of the trained network [143].

A different approach of particular interest for analog hardware is that of the model-free learning [45]. Model-free learning does not make any assumptions about the system or the environment. Thus, it is particularly suitable for situations in which the internal structure of the system varies (as in reconfigurable circuits), is difficult to model (as in a complex recurrent network) or is not known in great detail (as in a technologically imperfect chip whose structure is somehow different from the designed one due to the production errors). Such algorithms rely on approximate methods which include the response of the chip in calculating the performance, thus taking into consideration all imperfections present on the chip to be trained. Of course, such algorithms can only be used for on-chip or chip-in-the-loop training.

The model-free learning calculates a performance index for each weight state. The weights are perturbed continuously and the variation of the performance index is correlated to the perturbations. Instead of calculating a gradient of the error with respect to the weights, as in backpropagation for instance, the method uses parallel weight perturbations followed by correlations to calculate a gradient of the performance index which in turn is used to calculate the weight changes.

Several other learning algorithms that can be used in the training of a VLSI analog implementation of a neural network will be described in the following. One apparently trivial detail that turns out to be important in the VLSI implementation setup is the manner in which the weights are updated. Following the classification suggested in [96], we shall distinguish between the following types of weight updating:

- On-line weight updating: the weight updates are computed for each individual weight and pattern and applied immediately
- Batch weight updating: the weights updates are computed for each weight across the whole pattern set; the weights are updated only at the end of each pattern set. If performed in hardware, this particular updating needs more storage for the computation of the error over the whole pattern set and scales badly with the size of the network.
- Individual conditional updating: the error is only calculated for the current pattern and the weight update is performed only if a certain condition is satisfied (e.g. the error decreases).
- Batch conditional updating: the error is calculated over the whole pattern set and the update is performed only if a given condition is satisfied.
- On-line perturbation updating: the error is calculated for each individual pattern; the weight(s) are changed with a small quantity and the effect of that change on the pattern error is calculated. The weight update(s) is(are) calculated and applied.
- Batch perturbation updating: similar with on-line perturbation updating only that the error and its change determined by the chosen perturbation(s) are calculated over the whole pattern set, in a batch manner.

One algorithm which has been used in analog hardware implementations is the standard backpropagation (BP) algorithm [154]. Due to the fact that standard error backpropagation usually needs over 12 bits of resolution [90], [86], [202], [201], [179], the calculation of the gradient of the error with respect to the weights is usually calculated on a host computer with the errors provided by the actual outputs of the chip in a chip-in-the-loop manner [122, 96]. The backpropagation can be used with on-line and batch weight updating.

In the weight perturbation training [97], the error is calculated with the current weights. Subsequently, one or more weights are perturbated with a small known quantity. Then, the error is calculated again with the output produced by the perturbated weights. This effectively produces an estimate of the partial derivative of the pattern error with respect to the perturbed weight(s) as follows:

$$\frac{\partial E}{\partial w_{ij}} \approx \frac{E(\mathbf{w}) - E(\mathbf{w}+\mathbf{p})}{p_{ij}}$$

In this algorithm, the perturbation matrix $\mathbf{p}$ has only the element p_{ij} different from zero. Then, the weight change is calculated as in backpropagation i.e. as a fraction of the gradient:

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}}$$

This algorithm can use on-line and batch perturbation updating. A particular interesting property of the class of algorithms using perturbations is that like the model-free learning, they take into consideration automatically all imperfections of the circuit.

The weight perturbation described above calculates the weight updates by perturbing just one

weight at any one time. However, the weights can be perturbed in a parallel manner as in the stochastic error descent proposed by Cauwenberghs [28]. This algorithm updates the weights as follows:

$$\Delta \mathbf{w} = -\eta \left(E\left(\mathbf{w} + \pi \right) - E(\mathbf{w}) \right) \pi$$

where an element π_{ij} of the matrix π corresponds to a perturbation p_{ij}. Since all the weights are perturbed simultaneously, the elements of the perturbation matrix π need to be uncorrelated in space and time i.e. orthogonal. If this condition is not respected, the efficiency of the training will be decreased due to the impossibility of distinguishing the contribution of each individual weight in producing the error.

Another algorithm using parallel weight perturbations has been proposed by Alspector [7]. This algorithm uses the idea that in an analog environment in which the precision is limited, the perturbation can have the magnitude of the smallest change above the noise threshold and can be kept constant. Thus, one can apply always the same perturbation but with a random sign as follows:

$$\Delta \mathbf{w} = -\eta \left(E\left(\mathbf{w} + \mathbf{p} \right) - E(\mathbf{w}) \right) \mathbf{p}'$$

Here, $\mathbf{p}$ is the perturbation matrix and $\mathbf{p}'$ is the matrix of the inverses of individual perturbations $p'_{ij} = \frac{1}{p_{ij}}$. Again, all weights in the network are perturbed simultaneously and the weights can be update with either one of batch and on-line strategies.

An intermediate strategy between perturbing a single weight and perturbing all of them in parallel is to perturb only the weights feeding into one single neuron. This is the idea of the summed weight neuron perturbation developed by Flower and Jabri [71]. A training cycle will iterate over all neurons calculating the neuron weight changes as follows:

$$\Delta \mathbf{w}_{\rho_j} = -\eta \left(E\left(\mathbf{w} + \rho_j \right) - E(\mathbf{w}) \right) \rho'_j$$

where ρ_j is a perturbation matrix in which only the elements corresponding to the weights feeding into neuron j are different from zero. The matrix ρ'_j is the matrix formed with the inverses of the individual perturbations, as above.

The disadvantages of error perturbation include a slow convergence speed and the need to store various quantities such as error/weight gradients. Furthermore, due to the reduced weight resolution in analog hardware, the algorithms above can run into difficulties including discontinuities and flat areas in the weight surface. Such difficulties may cause the training to fail due to either oscillations or local minima. In order to avoid such convergence problems, several techniques combined a local search technique such as a gradient descent with more global optimization techniques such as simulated annealing [120] or even random walks [200]. Other techniques that use a more global type of search in the weight space include stochastic learning [5], mean field learning [162] and real-time clustering [164].

VLSI motivations have motivated a recent interest towards neural networks using limited precision and/or integer weights. The idea behind such networks is that limited precision integer weights can be implemented more efficiently in hardware, especially in terms of die area. [148, 149] deal with limited precision neural networks modelled by multilinear threshold functions. Papers such as [104, 105] present training algorithms that can provide solutions using exclusively integer weights. The computational power of such networks has been analyzed in [54, 52, 17]. Constructive algorithms using limited precision integer weights have also been presented in [51, 53]. Such algorithms have several interesting features such as a guaranteed convergence, a fast training and the ability to determine an architecture able to solve the problem during the training. The latter property also means that such algorithms are very suitable for use with hardware implementations able to deal with variable configurations. Another alternative is to have a larger network with a fixed architecture but to set various weights to zero as required by the architecture found by the algorithm.

3 Conclusions

This paper reviewed some analog implementations of neural network and analyzed and discussed several issues related to their development. Also, the paper presented several criteria for classifying neural implementation together with the taxonomy induced by such criteria.

The analog type of implementations appears to have been extremely attractive for researchers and companies alike. Since the very beginning of the neural network field, analog implementations have accompanied theoretical developments sometimes with amazingly short delays. Analog chips implementing neural networks have been also shown to be very effective in many practical applications such as intelligent control [43, 42], nuclear physics [12, 13, 14, 39, 125], optical character recognition

[107], tracking and target recognition (the Irvine Sensors-JPL chip mentioned in Section 2.2), etc. Due to some of their properties including high speed and efficiency and to the analog nature of the signals processed, they can be used directly (i.e. without any supplementary interface) in many real-world applications.

However, analog implementations also have disadvantages including a limited precision, susceptibility to noise, susceptibility to thermal and power supply variations, etc. The designs reviewed here have shown that such disadvantages can be overcome. The existence of several commercial products incorporating analog neural network technology as well as the use of such chips in several scientific applications, as shown above, allow us to conclude that analog implementations have successfully overcome the exploratory phase. This success can be attributed to i) the development of the CMOS fabrication process which now allows a density which was unconceivable only a few years ago; the design of advanced circuitry orientated towards the necessities of neural networks implementations and ii) the development of new 'VLSI friendly' training algorithms which were designed to address some weaknesses of the analog approach. Weight perturbations and limited precision weights algorithms allow elegant solutions to some of the problems created by imperfections resulted from the fabrication process.

References

[1] I. Aleksander and H. B. Morton. *Introduction to neural computing*. International Thompson Press, 1995.

[2] I. Aleksander and T. J. Stonham. Guide to pattern recognition using random-access memories. *Computer and Digital Techniques*, 2(1):29–40, 1979.

[3] I. Aleksander, W. V. Thomas, and P. A. Bowden. Wisard: A radical step forward in image recognition. *Sensor Review*, pages 120–124, 1984.

[4] J. Alspector. A Bolzmann learning system. In *Adaptive Analog VLSI Neural Systems*. Chapman and Hall, 1996.

[5] J. Alspector, B. Gupta, and R. B. Allen. Performance of a stochastic learning microchip. In *Advances in Neural Information Processing Systems*, volume 1, pages 748–760, San Mateo, CA, 1989. Morgan Kaufman.

[6] J. Alspector, A. Jayakumar, and S. Luna. Experimental evaluation of learning in a neural microsystem. In *Proc. of Neural Information Processing Systems (NIPS), vol. 4*, pages 871–878. Morgan Kaufmann, San Mateo, Ca, 1991.

[7] J. Alspector, R. Meir, B. Yuhas, and A. Jayakumar. A parallel gradient descent method for learning in analog VLSI neural networks. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems*, volume 5, pages 836–844. Morgan Kaufmann Publisher, 1993.

[8] D. Anquita, S. Ridella, and S. Rovetta. Worst case analysis of weight inaccuracy effects in multilayer perceptrons. *IEEE Transactions on Neural Networks*, 10(2):415–418, March 1999.

[9] Y. Arima, K. Mashito, K. Okada, T. Yamada, A. Maeda, H. Notani, H. Kondou, and S. Kayano. A 335-neuron 28K-synapse self-learning neural network chip with branch-neuron-unit-architecture. *IEEE Journal of Solid-State Circuits*, 27(11):1637–1644, November 1991.

[10] Y. Arima, M. Murasaki, T. Yamada, A. Maeda, and H. Shinohara. A refreshable analog VLSI neural network chip with 400 neurons and 40K synapses. *IEEE Journal of Solid-State Circuits*, 27(12):1854–1861, Dec. 1992.

[11] W. Bair and C. Koch. An analog VLSI chip for finding edges from zero-crossing. In J. E. Moody and D. S. Touretzky, editors, *Neural Information Processing Systems*, volume 2, pages 399–405, Palo Alto, CA, 1991. Morgan Kaufmann.

[12] C. Baldanza, J. Beichter, F. Bisi, N. Bruels, C. Bruschini, A. Cotta-Ramusino, I. D'Antone, L. Malferrari, P. Mazzanti, P. Musico, P. Novelli, F. Odorici, R. Odorico, M. Passaseo, and M. Zuffa. Results from a ma16-based neural trigger in an experiment looking for beauty. *Nuclear Instruments and Methods in Physics research, Section A - Accelerators, spectrometers, detectors and associated equipment*, 376(3):411–419, July 1996.

[13] C. Baldanza, J. Beichter, F. Bisi, N. Bruels, A. Cotta-Ramusino, I. D'Antone, L. Malferrari, P. Mazzanti, F. Odorici, R. Odorico, and M. Zuffa. Hardware and software structure of a neural network trigger based on the ETANN and MA16 chips. *Nuclear Instruments and Methods in Physics Research A – Accelerators, Spectrometers, Detectors and Associated Equipmente*, 373(2):261–281, April 1996.

[14] C. Baldanza, F. Bisi, A. Cotta-Ramusino, I. D'Antone, L. Malferrari, P. Mazzanti, F. Odorici, R. Odorico, M. Zuffa, C. Bruschini, P. Musico, P. Novelli, and M. Passaseo. Results from an online nonleptonic neural trigger implemented in an experiment looking for beauty. *Nuclear Instruments and Methods in Physics Research A*, 361(3):506–518, July 1995.

[15] O. Barkan, W. R. Smith, and G. Persky. Design of coupling resistor networks for neural network hardware. *IEEE Transactions on Circuits and Systems*, 37(6):756–765, June 1990.

[16] V. Beiu. *Neural Networks using Threshold Gates: A complexity analysis of their area and time efficient VLSI implementations*. PhD thesis, Katholieke Universiteit Leuven, May 1994.

[17] V. Beiu, S. Draghici, and H. E. Makaruk. On limited fan-in optimal neural networks. Technical Report LA-UR-97-2873, Los Alamos National Laboratory, 1997.

[18] R. G. Benson. *Analog VLSI supervised Learning System*. PhD thesis, California Institute of Technology, 1993.

[19] T. M. Bernard, B. Y. Zavidovique, and F. J. Devos. A programmable artificial retina. *IEEE Journal of Solid-State Circuits*, 28(7):789–798, July 1993.

[20] W. W. Bledsoe and I. Browning. Pattern recognition and reading by machine. In *Proc. Joint Comp. Conference*, pages 255–232, 1959.

[21] K. Boahen and A. Andreou. A contrast sensitive silicon retina with reciprocal synapses. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems*, volume 4, pages 764–774, San Mateo, CA, 1992. Morgan Kaufmann.

[22] B. Boser, E. Sáckinger, J. Bromley, Y. L. Cun, and L. D. Jackel. An analog neural network processor with programmable topology. *IEEE Journal of Solid-State Circuits*, 26:2017–2025, December 1991.

[23] J. Brauch, S. M. Tam, M. A. Holler, and A. L. Shmurun. Analog VLSI neural networks for impact signal processing. *IEEE Micro*, 12(6):34–45, December 1992.

[24] T. X. Brown, M. D. Tran, T. Duong, T. Daud, and A. P. Thakoor. Cascaded VLSI neural network chips: Hardware learning for pattern recognition and classification. *Simulation*, 58(5):340–346, 1992.

[25] G. Cairns and L. Tarassenko. Precision issues for learning with analog VLSI multilayer perceptrons. *IEEE Micro*, 15(3):54–56, June 1995.

[26] R. Castello, D. D. Caviglia, M. Franciotta, and F. Montecchi. Self-refreshing analog memory cell for variable synaptic weights. *Electronics Letters*, 27(20):1871–1873, 1991.

[27] H. A. Castro, S. M. Tam, and M. A. Holler. Implementation and performance of an analog nonvolatile neural-network. *Analog Integrated Circuits and Signal Processing*, 4(2):97–113, Sep 1993.

[28] G. Cauwenberghs. A fast stochastic error-descent algorithm for supervised learning and optimisation. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems*, volume 5, pages 244–251. Morgan Kaufmann Publisher, 1993.

[29] G. Cauwenberghs. A micropower CMOS algorithmic A/D/A converter. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 42(11):913–919, 1995.

[30] G. Cauwenberghs. Neuromorphic learning VLSI systems: A survey. In T. S. Lande, editor, *Neuromorphic Systems Engineering*, pages 381–408. Kluwer Academic Publishers, 1998.

[31] G. Cauwenberghs and A. Yariv. Fault-tolerant dynamic multi-level storage in analog VLSI. *IEEE Transactions on Circuits and Systems II*, 41(12):827–829, 1994.

[32] T.-H. Chao. An integrated optoelectronic ATR processor. In *Proc. of JPL Workshop on Neural Network Practical Applications and Prospects*, pages 193–208, Pasadena, CA, May 1994.

[33] J. Choi and B. J. Sheu. VLSI design of compact and high-precision analog neural network processors. In *Proc. IEEE/INNS Inter. Joint Conf. Neural Networks*, volume 2, pages 637–641, Baltimore, MD, July 1992.

[34] L. O. Chua and L. Yang. Cellular neural networks: Applications. *IEEE Transactions on Circuits and Systems*, 35(10):1273–1290, Oct. 1988.

[35] L. O. Chua and L. Yang. Cellular neural networks: Theory. *IEEE Transactions on Circuits and Systems*, 35(10):1257–1272, Oct. 1988.

[36] T. G. Clarkson, C. K. Ng, and Y. Guan. The pRAM: An adaptive VLSI chip. *IEEE Transactions on Neural Networks*, 4(3):408–412, May 1993.

[37] R. J. Coggins, M. A. Jabri, B. F. Flower, and S. J. Pickard. A hybrid analog and digital neural network for intracardiac morphology classification. *IEEE Journal of Solid State Circuits*, 30(5):542–550, 1995.

[38] R. J. Coggins, M. A. Jabri, B. F. Flower, and S. J. Pickard. A low power network for on-line diagnosis of heart patients. *IEEE Micro Magazine*, 15(3):18–25, 1995.

[39] J. S. Conway and C. Loomis. Using an analog neural-network to trigger on Tau-Leptons at CDF. *International Journal of Modern Physics C - Physics and Computers*, 6(4):549–554, August 1995.

[40] K. R. Crounse and L. O. Chua. The CNN universal machine is as universal as a Turing machine. *IEEE Transactions on Circuits and Systems I*, 43(4):353–355, April 1996.

[41] J. M. Cruz. A CNN chip for connected component detection. *IEEE Transactions on Circuits and Systems*, 38(7):812–817, July 1991.

[42] R. Damle and V. Rao. Neural network based optimizing controllers for smart structural systems. *Smart Materials and Structures*, 7(1):23–30, Feb. 1998.

[43] R. Damle, V. Rao, and F. Kern. Robust control of smart structures using neural network hardware. *Smart Materials and Structures*, 6(3):301–314, June 1997.

[44] T. Delbrück. Silicon retina with correlation-based, velocity-tuned pixels. *IEEE Transactions on Neural Networks*, 4(3):529–541, May 1993.

[45] A. Dembo and T. Kailath. Model-free distributed learning. *IEEE Transactions on Neural Networks*, 1(1):58–70, 1990.

[46] J. V. der Spiegel, P. Mueller, D. Blackman, P. Chance, C. Donham, R. Etienne-Cummings, and P. Kinget. An analog neural computer with modular architecture for real-time dynamic computation. *IEEE Journal of Solid-State Circuits*, 27:82–91, January 1992.

[47] J. V. der Spiegel et.al. An analog neural computer with modular architecture for real-time dynamic computations. *IEEE Journal of Solid-State Circuits*, 27(1):82–92, Jan 1992.

[48] B. K. Dolenko and H. C. Card. Tolerance to analog hardware of on-chip learning in backpropagation networks. *IEEE Transactions on Neural Networks*, 6(5):1045–1052, 1995.

[49] R. Dominguez-Castro, S. Espejo, A. Rodriguez-Vazquez, R. A. Carmona, P. Foldesy, A. Zarandy, P. Szolgay, and T. Roska. A 0.8 μm CMOS two-dimensional programmable mixed-signal focal-plane array processor with on-chip binary imaging and instructions storage. *IEEE Journal of Solid-State Circuits*, 32(7):1013–1026, Jul. 1997.

[50] S. Draghici. On the complexity-capability relationship of adaptive systems based on VLSI friendly neural networks. In *Lecture Notes in Artificial Intelligence, Proceedings of XII Canadian Conference on Artificial Intelligence AI'98*, volume 1418, pages 285–297. Springer-Verlag, 18-20 June 1998.

[51] S. Draghici. On VLSI-optimal constructive algorithms for classification problems. In *Proc. of EIS'98 International Symposium on Engineering of Intelligent Systems*, pages 456–462. ICSC Academic Press, 1998.

[52] S. Draghici. Using information entropy bounds for the design and implementation of VLSI friendly neural networks. In *Proceedings of of 1998 IEEE World Congress on Computational Intelligence IJCNN'98*, pages 547–552. IEEE, 4-9 May 1998.

[53] S. Draghici, V. Beiu, and I. Sethi. A VLSI optimal constructive algorithm for classification problems. In *Smart Engineering Systems: Neural Networks, Fuzzy Logic, Data Mining and Evolutionary Programming*, pages 141–151. ASME Press, 1997.

[54] S. Draghici and I. K. Sethi. On the possibilities of the limited precision weights neural networks in classification problems. In J. Mira, R. Moreno-Diaz, and J. Cabestany, editors, *Biological and Artificial Computation: From Neuroscience to Technology, Lecture Notes in Computer Science*, pages 753–762. Springer-Verlag, 1997.

[55] G. Dundar and K. Rose. The effect of quantization on multilayer neural networks. *IEEE Transactions on Neural Networks*, 6:1446–1451, 1995.

[56] T. Duong, S. Kemeny, M. Tran, T. Daud, A. Thakor, D. Ludwig, C. Saunders, and J. Carson. Low power analog neurosynapse chips for a 3-D Sugarcube neuroprocessor. In *IEEE International Conference on Neural Networks*, volume 3, pages 1907–1911, Orlando, FL, June 1994.

[57] D. A. Durfee and F. S. Shoucair. Low programming voltage floating gate analog memory cells in standard VLSI CMOS technology. *Electronics Letters*, 28(10):925–927, May 1992.

[58] J. G. Elias, D. P. M. Northmore, and W. Westerman. An analog memory device for spiking silicon neurons. *Neural Computation*, 9:419–440, 1997.

[59] S. Espejo. *VLSI Design and Modeling of CNNs*. PhD thesis, University of Sevilla, Spain, Apr. 1994.

[60] S. Espejo, R. Carmona, R. Dominquez-Castro, and A. R. Vazquez. A CNN universal chip in CMOS technology. *International Journal of Circuit Theory and Applications*, 24(1):93–109, Jan-Feb 1996.

[61] S. Espejo, R. Dominguez-Castro, R. Carmona, and A. Rodriguez-Vazquez. CMOS optical-sensor array with high-output current levels and automatic signal-range centering. *Electronics Letters*, 30(22):1847–1849, Oct. 1994.

[62] S. Espejo, A. Rodriquez-Vasquez, R. Dominquez-Castro, J. L. Huertas, and E. Sanchez-Sinencio. Smart-pixel cellular neural networks in analog current-mode CMOS technology. *IEEE Journal of Solid-State Circuits*, 29(8):895–905, Aug. 1994.

[63] R. Etienne-Cummings, C. Donham, and J. V. der Spiegel. Spatiotemporal computations with a general purpose analog neural computer: real-time visual motion estimation. In *IEEE Int. Conf. on Neural Networks*, pages 1836–1841, Orlando, Fl, June 1994.

[64] W.-C. Fang, B. J. Sheu, and J.-C. Lee. Real-time computing of optical flow using adaptive VLSI neuroprocessors. In *Proc. of IEEE International Conference on Computer Design*, Cambridge, MA, 1990.

[65] N. H. Farhat. Optoelectronic neural networks and learning machines. *IEEE Circuits and Devices Magazine*, 5(5):32–41, September 1989.

[66] M. R. Feldman, S. C. Esener, C. C. Guest, and S. H. Lee. Comparison between optical and electrical interconnects based on power and speed considerations. *Applied Optics*, 27(9):1742–1751, May 1988.

[67] E. Fiesler and R. Beale, editors. *Handbook of Neural Computation*. Oxford University Press and the Inst. of Physics Publ., 1996.

[68] E. Filho, M. Fairhurst, and D. L. Bisset. Adaptive pattern recognition using the goal seeking neuron. *Pattern Recognition Letters*, 12:131–138, 1991.

[69] W. A. Fisher, R. J. Fujimoto, and R. C. Smithson. A programmable analog neural network processor. *IEEE Transactions on Neural Networks*, 2:222–229, March 1991.

[70] B. Flower and M. Jabri. Summed weight neuron perturbation: an O(n) improvement over weight perturbation. In *Advances in Neural Information Processing Systems*, volume 5, pages 212–219. Morgan Kaufmann Publisher, 1993.

[71] R. C. Frye, E. A. Rietman, and C. C. Wong. Back-propagation learning and nonidealities in analog neural network hardware. *IEEE Transactions on Neural Networks*, 2(1):110–117, 1991.

[72] O. Fujita and Y. Amemiya. A floating-gate analog memory device for neural networks. *IEEE Device*, 40(11):2029–2055, November 1993.

[73] M. Glessner and W. Pöchmüller. *An Overview of Neural Networks in VLSI*. Chapman and Hall, London, 1994.

[74] M. A. Glover and W. T. Miller. A massively-parallel SIMD processor for neural network and machine vision applications. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 843–849. Morgan Kaufmann, 1994.

[75] D. Gorse. Review of the theory of pRAMs. In N. M. Allinson, editor, *Weightless Neural Network Conference*, pages 13–17, 1993.

[76] D. Gorse, J. G. Taylor, and T. G. Clarkson. Extended functionality for pRAMs. In *Proc. of Int. Conf. on Artificial Neural Networks ICANN'94*, pages 705–708, 1994.

[77] H. P. Graf and D. Henderson. A reconfigurable CMOS neural network. In *Technical Digest of IEEE International Solid-State Circuits Conference*, pages 144–145, February 1990.

[78] H. P. Graf and L. D. Jackel. Analog electronic neural network circuits. *IEEE Circuits and Device Mag.*, 5:44–49, 1989.

[79] H. P. Graf, L. D. Jackel, R. E. Howard, and B. Straughn. VLSI implementation of a neural network memory with several hundreds of neurons. In J. S. Denker, editor, *Neural Networks for Computing, AIP Conference Proceedings*, volume 151, pages 182–187, Snowbird, UT, 1986.

[80] H. P. Graf, L. D. Jackel, and W. E. Hubbard. VLSI implementation of a neural network model. *Computer*, pages 41–49, March 1988.

[81] S. C. Gustafson and G. R. Litte. Neural network models based on optical resonator designs. In *Technical Digest, Topical Meeting on Optical Computing*, Washington, DC, 1989. Optical Society of America.

[82] P. Hasler, C. Diorio, B. A. Minch, and C. Mead. Single transistor learning synapses. In *Advances in Neural Information Processing Systems*, volume 7, pages 817–824, Cambridge, MA, 1995. MIT Press.

[83] J. Heemskerk. *Neurocomputers for Brain-Style Processing.* PhD thesis, Leiden University, Leiden, The Netherlands, 1995.

[84] P. Heim and E. A. Vittoz. Precise analog synapse for Kohonen feature maps. *IEEE Journal of Solid-State Circuits*, 28(8):982–985, August 1994.

[85] M. Höhfeld and S. E. Fahlman. Learning with limited numerical precision using the Cascade-Correlation algorithm. *IEEE Transactions on Neural Networks*, 3(4):602–611, 1992.

[86] B. Hochet, V. Peiris, S. Abdo, and M. J. Declercq. Implementation of a learning Kohonen neuron based on a new multilevel storage technique. *IEEE Journal of Solid-State Circuits*, 26(3):262–267, 1991.

[87] M. Holler. VLSI implementations of learning and memory systems. In *Advances in Neural Information Processing Systems*, volume 3, pages 993–1000, San Mateo, CA, 1991. Morgan Kaufman.

[88] M. Holler, S. Tam, H. Castro, and R. Benson. An electrically trainable artificial neural network (ETANN) with 10240 floating gate synapses. In *Proc. of IEEE/INNS Inter. Joint Conf. on Neural Networks*, volume 2, pages 191–196, 1989.

[89] P. Hollis. The effects of precision constraints in a backpropagation learning network. *Neural Computation*, 2(3):363–373, 1990.

[90] P. Hollis and J. J. Paulos. Artificial neural networks using MOS analog multiplier. *IEEE Journal of Solid State Circuits*, 25:849–855, June 1990.

[91] L. Holmstrom and P. Koistinen. Using additive noise in back-propagation training. *IEEE Transactions on Neural Networks*, 3(1):24–38, 1992.

[92] Y. Horio and S. Nakamura. Analog memories for VLSI neurocomputing. In E. Sánchez-Sinencio and C. Lau, editors, *Artificial Neural Networks: Paradigms, Applications and Hardware Implementations*, pages 344–363. IEEE Press, 1992.

[93] P. Ienne. Digital connectionist hardware: Current problems and future challenges. In *Lecture Notes in Computer Science, Biological and Artificial Computation: From Neuroscience to Technology*, volume 1240, pages 688–713. Springer Verlag, 1997.

[94] P. Ienne, T. Cornu, and G. Kuhn. Special-purpose digital hardware for neural networks: An architectural survey. *Journal of VLSI Signal Processing*, 13(1):5–25, 1996.

[95] M. A. Jabri, R. J. Coggins, and B. Flower. *Adaptive Analog VLSI Neural Systems.* Chapman and Hall, London, UK, 1996.

[96] M. A. Jabri and B. Flower. Weight perturbation: An optimal architecture and learning technique for analog VLSI feedforward and recurrent multilayer networks. *IEEE Transactions on Neural Networks*, 3(1):154–157, 1992.

[97] A. Jayakumar and J. Alspector. A cascadable neural network chip set with on-chip learning using noise and gain annealing. In *Proceedings of the 1992 IEEE Custom Integrated Circuits Conference*, pages 19.5.1–19.5.4, Boston, MA, May 1992.

[98] A. Jayakumar and J. Aspector. An analog neural-network co-processor system for rapid prototyping of telecommunications applications. In *Proceedings of the IEEE/INNS International Workshop on Application of Neural Networks to Telecommunications*, pages 13–19, Hillsdale, New Jersey, October 18-20 1993. Lawrence Erlbaum Associates.

[99] R. W. Johnson, R. K. Teng, and J. W. Blade, editors. *Multichip Modules: Systems Advantages, Major Constructions and Material Technologies.* IEEE Press, Piscataway, NJ, 1991.

[100] W. K. Kan and I. Aleksander. A probabilistic logical neural network for associative learning. In *Proc . of IEEE Conference on Neural Networks*, pages 541–548, San Diego, CA, 1987.

[101] J. Kane and M. Paquin. POPART: practical optical implementation of adaptive resonance theory 2. *IEEE Transactions of Neural Networks*, 4(4):695–702, July 1993.

[102] D. Kerns, J. E. Tanner, M. A. Silviotti, and J. Luo. CMOS UV-writable non-volatile analog storage. In *Proceedings of Advanced Research in VLSI International Conference*, Santa Cruz, CA, 1991.

[103] A. Khan and E. L. Hines. Integer-weights neural nets. *Electronics Letters*, 30(15):1237–1238, 1994.

[104] A. H. Khan and R. G. Wilson. Integer-weight approximation of continuous-weight multilayer feedforward nets. In *Proceedings. of the IEEE International Conference on Neural Networks*, volume 1, pages 392–397. IEEE Press, 1996.

[105] F. Kiamilev, P. Marchand, A. V. Krishnamoorthy, and S. C. E. adn S. H. Lee. Performance comparison between VLSI and optoelectronic multistage interconnection networks. *IEEE Journal of Lightwave Technology*, 9(12):1674–1692, December 1991.

[106] J. N. Kidder and D. Seligson. Fast recognition of noisy digits. *Neural Computation*, 5(6):885–892, Nov 1993.

[107] A. König. Survey and current status of neural network hardware. In F. Fogelman-Soulie and P. Gallinari, editors, *Proceedings of the International Conference on Artificial Neural Networks*, pages 391–410, October 1995.

[108] A. Kramer, C. K. Sin, R. Chu, and P. K. Ko. Compact EEPROM-based weight functions. In *Advances in Neural Information Processing Systems*, volume 3, pages 1001–1007, San Mateo, CA, 1991. Morgan Kaufman.

[109] H. Kwan and C. Z. Tang. Designing multilayer feedforward neural networks using simplified activation functions and one-power-of-two weights. *Electronics Letters*, 28(25):2343–2344, 1992.

[110] H. Kwan and C. Z. Tang. Multiplierless multilayer feedforward neural networks design suitable for continuous input-output mapping. *Electronics Letters*, 29(14):1259–1260, 1993.

[111] V. Lafargue, P. Garda, and E. Belhaire. An analog implementation of the Boltzmann machine with programmable learning algorithms. In J. G. Delgado-Frias and W. Moore, editors, *VLSI for neural networks and artificial intelligence*, pages 45–52. Plenum Press, 1994.

[112] T. S. Lande. Special issue on neuromorphic engineering. *Int. J. Analog Int. Circ. Signal Proc.*, March 1997.

[113] T. S. Lande, editor. *Neuromorphic Systems Engineering*. Kluwer Academic Publishers, 1998.

[114] O. Landolt. An analog CMOS implementation of a Kohonen network with learning capability. In J. G. Delgado-Frias and W. Moore, editors, *VLSI for neural networks and artificial intelligence*, pages 25–34. Plenum Press, 1994.

[115] J. A. Lansner and T. Lehmann. An analog CMOS chip set for neural networks with arbitrary topology. *IEEE Transactions on Neural Networks*, 4(3):441–444, May 1993.

[116] J. Lazzaro and C. Mead. Circuit models of sensory transduction in the cohlea. In C. Mead and M. Ismail, editors, *Analog VLSI Implementations of Neural Systems*, pages 85–102. Kluwer Academic, 1989.

[117] B. Lee and B. Sheu. Design of a neural-based A/D converted using modified Hopfield network. *IEEE Journal of Solid-State Circuits*, SC-24(4):1129–1135, August 1989.

[118] B. Lee and B. J. Sheu. A compact and general-purpose neural chip with electrically programmable synapses. In *Proceedings of IEEE Custom Integrated Circuits Conference*, pages 26.6.1–26.6.4, Boston, MA, May 1990.

[119] B. W. Lee and B. J. Sheu. *Hardware Annealing in Analog VLSI Neurocomputing*. Kluwer Academic Publisher, Boston, MA, 1991.

[120] B. W. Lee, B. J. Sheu, and H. Yang. Analog floating-gate synapses for general-purpose VLSI neural computation. *IEEE Transactions on Circuits and Systems*, 38:654–658, 1991.

[121] P. H. W. Leong and M. A. Jabri. A VLSI neural network for arrhythmia classification. In *Proc. of the International Joint Conference on Neural Networks*, volume 2, pages 678–683, Baltimore, MA, 1992.

[122] B. Linares-Barranco and E. S.-A. R.-V.-J. L. Huertas. A modular T-mode design approach for analog neural network hardware implementations. *IEEE Journal of Solid-State Circuits*, 27(5):701–713, May 1992.

[123] B. Linares-Barranco, E. Sánchez-Sinencio, A. Rodriguez-Vazquez, and J. L. Huertas. A CMOS analog adaptive BAM with on-chip learning and weight refreshing. *IEEE Transactions on Neural Networks*, 6(4):445–457, 1993.

[124] T. Lindblad, C. S. Lindsey, F. Block, and A. Jayakumar. Using software and hardware neural networks in a Higgs search. *Nuclear Instruments and Methods in Physics Research A - Accelerators, Spectrometers, Detectors and Associate Equipment*, 356(2–3):498–506, March 1995.

[125] C. S. Lindsey and T. Lindblad. Review of hardware neural networks: A user's perspective. In *Proceedings of Third Workshop on Neural Networks: From Biology to High Energy Physics*, 1993.

[126] R. Y. Liu, C. Y. Wu, and I. C. Jou. A CMOS current-mode design of modified learning-vector-quantization neural networks. *Int. J. Analog Integ. Circ. Signal Proc.*, 8(2):157–181, 1995.

[127] T. Lundin, E. Fiesler, and P. Moerland. Connectionist quantization functions. In *SIPAR Workshop '96 Proceedings*, pages 33–36. CUI, 1996.

[128] D. Macq, M. Verleysen, P. Jespers, and J. D. Legat. Analog implementation of a Kohonen map with on-chip learning. *IEEE Transactions on Neural Networks*, 4(3):456–461, May 1993.

[129] M. A. Mahowald and C. Mead. Silicon retina. In C. Mead, editor, *Analog VLSI and Neural Systems*. Addison-Wesley, 1989.

[130] P. Masa, K. Hoen, and H. Wallinga. A high-speed analog neural processor. *IEEE Micro*, 14(3):40–50, June 1994.

[131] J. McCanny, J. McWhirter, and E. Swarzlander, editors. *Systolic Array Processors*. Prentice Hall, 1989.

[132] C. Mead and M. Ismail, editors. *Analog VLSI Implementations of Neural Systems*. Kluwer Academic Publisher, 1989.

[133] C. A. Mead. *Analog VLSI and Neural Systems*. Addison-Wesley, 1989.

[134] P. Moerland and E. Fiesler. *Neural Network Adaptations to Hardware Implementations*, chapter E1.2 of the Handbook of Neural Computation. Oxford University Press and Institute of Physics, Bristol, UK and New York, USA, 1997.

[135] N. Morgan, editor. *Artificial Neural Networks: Electronic Implementations*. IEEE Computer Society Press, Los Alamitos, CA, 1990.

[136] T. Morie and Y. Amemiya. An all-analog expandable neural-network LSI with on-chip backpropagation learning. *IEEE Journal of Solid-State Circuits*, 29(9):1086–1093, September 1994.

[137] T. Morishita, Y. Tamura, and T. Otsuki. A BiCMOS analog neural network with dynamically updated weights. In *Technical Digest of IEEE International Solid-State Circuits Conference*, pages 142–143, San Francisco, February 1990.

[138] P. Muller and J. V. der Spiegel et. al. Real time decomposition and recognition of acoustical patterns with an analog neural computer. In *SPIE Vol. 1709*, chapter Applications of Artificial Neural Networks III, pages 758–769. 1992.

[139] D. B. Mundie and L. W. Massengill. Weight decay and resolution effects in feedforward artificial neural networks. *IEEE Transactions on Neural Networks*, 2(1):168–170, January 1991.

[140] A. F. Murray. Analogue noise-enhanced learning in neural network circuits. *Electronics Letters*, 2(17):1546–1548, 1991.

[141] A. F. Murray. Multilayer perceptron learning optimized for on-chip implementation: A noise robust system. *Neural Computation*, 4:366–381, 1992.

[142] A. F. Murray and P. J. Edwards. Synaptic noise during MLP training enhances fault-tolerance, generalization and learning trajectory. In *Advances in Neural Information Processing Systems*, volume 5, pages 491–498, San Mateo, CA, 1993. Morgan Kaufman.

[143] A. F. Murray and A. V. W. Smith. Asynchronous arithmetic for VLSI neural systems. *Electronics Letters*, 23(12):642–643, June 1987.

[144] A. F. Murray and A. V. W. Smith. A novel computational and signalling method for VLSI neural networks. In *European Solid State Circuits Conference*, pages 19–22, Berlin, 1987. VDE-Verlag.

[145] A. F. Murray and L. Tarassenko. *Analogue Neural VLSI - A Pulse Stream Approach.* Chapman and Hall, 1994.

[146] L. Nemes, L. O. Chua, and T. Roska. Implementation of arbitrary boolean functions on a CNN universal machine. *International Journal of Circuit Theory and Applications*, 26(6):592–610, Nov-Dec 1998.

[147] Z. Obradovic and I. Parberry. Analog neural networks of limited precision i: Computing with multilinear threshold functions. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems*, pages 702–709, San Mateo, CA, 1990. Morgan Kaufmann.

[148] Z. Obradovic and I. Parberry. Computing with discrete multi-valued neurons. *J. Comp. and System Sci.*, 45:471–492, 1992.

[149] Y. Owechko. Optoelectronic resonator neural network. *Applied Optics*, 26:5104–5111, 1987.

[150] J. I. Raffel. Electronic implementation of neuromorphic systems. In *Proc of Custom Integrated Circuits Conference*, pages 10.1.1–10.1.7, Rochester, NY, May 1988.

[151] L. M. Reyneri and E. Filippi. An analysis on the performance of silicon implementations of backpropagation algorithms for artificial neural networks. *IEEE Comput.*, 40(12):1380–1389, 1991.

[152] T. Roska and L. O. Chua. The CNN universal machine - an analogic array computer. *IEEE Transactions on Circuits and Systems II*, 40(3):163–173, March 1993.

[153] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error backpropagation. In D. E. Rumelhart, J. L. MeClelland, and the PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructures of Cognition*, volume 1. MIT Press/Bradford Books, 1986.

[154] E. Säckinger, B. E. Boser, J. Bromley, Y. LeCun, and L. D. Jackel. Application of the ANNA neural network chip to high-speed character recognition. *IEEE Transactions on Neural Networks*, 3(3):498–505, May 1992.

[155] E. Säckinger and H. P. Graf. A board system for high-speed image analysis and neural networks. *IEEE Transactions on Neural Networks*, 7(1):214–221, Jan 1996.

[156] E. Sánchez-Sinencio and C. Lau, editors. *Artificial Neural Networks: Electronic Implementations.* IEEE Computer Society Press, 1992.

[157] E. Sánchez-Sinencio and R. Newcomb. Special issue on neural network hardware. *IEEE Transactions on Neural Networks*, 3(3), 1992.

[158] E. Sánchez-Sinencio and R. Newcomb. Special issue on neural network hardware. *IEEE Transactions on Neural Networks*, 1993.

[159] R. Sarpeshkar, R. F. Lyon, and C. Mead. *A low-power wide-dynamic-range analog VLSI cochlea*, chapter 3. In Lande [114], 1998.

[160] S. Satyanarayana, Y. P. Tsividis, and H. P. Graf. A reconfigurable VLSI neural network. *IEEE Journal of Solid-State Circuits*, 27:67–81, January 1992.

[161] C. R. Schneider and H. C. Card. CMOS mean field learning. *Electronics Letters*, 27(19):1702–1704, 1991.

[162] D. B. Schwartz, R. E. Howard, and W. E. Hubbard. A programmable analog neural network chip. *IEEE Journal of Solid-State Circuits*, 24:313–319, 1989.

[163] T. Serrano-Gotarredona and B. Linares-Barranco. A real-time clustering microchip neural engine. *IEEE Transactions on VLSI Systems*, 4(2):195–209, 1996.

[164] B. J. Sheu and J. Choi. *Neural Information Processing and VLSI.* Kluwer Academic Publishers, 1995.

[165] T. Shima, T. Kimura, Y. Kamatani, T. Itakura, Y. Fujita, and T. Iida. Neuro chips with on-chip backpropagation and/or Hebbian learning. *IEEE Journal of Solid-State Circuits*, 27:1868–1876, 1992.

[166] M. A. Silviotti, M. A. Mahowald, and C. A. Mead. Real-time visual computations using analog CMOS processing arrays. In *Proc. of the Stanford Advanced Research in VLSI Conference*, Cambridge, MA, 1987. MIT Press.

[167] M. J. S. Smith. An analog integrated neural network capable of learning the Feigenbaum logistic map. *IEEE Transactions on Circuits and Systems*, 37(6):841–844, 1990.

[168] A. Soennecken, U. Hillerignmann, and K. Goser. Floating gate structures as nonvolatile analog memory cells in 1.0μm-LOCOS-CMOS technology with PZT dielectrica. *Microel. Eng.*, 15:633–636, 1991.

[169] B. H. Soffer, G. J. Dunning, Y. Owechko, and E. Marom. Associative holographic memory with feedback using phase-conjugate mirrors. *Optics Letters*, 11:118–120, February 1986.

[170] A. Stoffels, T. Roska, and L. O. Chua. On object-oriented video coding using the CNN universal machine. *IEEE Transactions on Circuits and Systems I*, 43(11):948–952, November 1996.

[171] A. Stoffels, T. Roska, and L. O. Chua. Object-oriented image analysis for very-low-bitrate video-coding systems using the CNN universal machine. *International Journal of Circuit Theory and Applications*, 25(4):235–258, Jul-Aug 1997.

[172] H. Suzuki, T. Matsumoto, and L. O. Chua. A CNN handwritten character recognizer. *International Journal of Circuit Theory and Applications*, 20(9):601–612, Sept. 1992.

[173] E. Swartzlander, editor. *Wafer Scale Integration.* Kluwer Academic Publishers, Boston, MA, 1989.

[174] T. Sziranyi and J. Csicsvari. High-speed character recognition using a dual cellular neural network architecture. *IEEE Transactions on Circuits and Systems II*, 40(3):223–231, March 1993.

[175] T. Sziranyi and L. Czuni. Image compression by orthogonal decomposition using cellular neural network chips. *International Journal of Circuit, Theory and Applications*, 27(1):117–134, Jan-Feb 1999.

[176] P. Szolgay and K. Tomordi. Analogic algorithms for optical detection of breaks and short circuits on the layouts of printed circuit boards using CNN. *International Journal of Circuit Theory and Applications*, 27(1):103–116, Jan-Feb 1999.

[177] J. E. Tanner and C. A. Mead. A correlating optical motion detector. In *Proc. of Conference on Advanced Research in VLSI*, Dedham, MA, 1984. Artech House.

[178] L. Tarassenko, J. Tombs, and G. Cairns. On-chip learning with analogue VLSI neural networks. *International Journal of Neural Systems*, 4(4):419–426, 1993.

[179] S. K. Tewksbury, editor. *Microelectronic System Interconnections: Performance and Modeling*. IEEE Press, Pitscataway, NJ, 1994.

[180] A. P. Thakoor, A. Moopenn, J. Lambe, and S. K. Khanna. Electronic hardware implementation of a neural network model. *Applied Optics*, 26:5085–5092, December 1987.

[181] P. Thiran and M. Hassler. Self-organization of a one-dimensional Kohonen network with quantized weights and inputs. *Neural Networks*, 7(9):1427–1439, 1994.

[182] D. Thompson, editor. *The Oxford Modern English dictionary*. Oxford University Press, 2-nd edition, 1996.

[183] A. Thomsen and M. A. Brooke. Low control voltage programming of floating-gate mosfets and applications. *IEEE Circ. I*, 41(6):443–452, June 1994.

[184] Y. Tsividis and S. Satyanarayana. Analogue circuits for variable synapse electronic neural networks. *Electronics Letters*, 23:1313–1314, November 1987.

[185] R. R. Tummala and E. J. Rynaszewski, editors. *Microelectronics Packaging Handbook*. Van Nostrand Reinhold, New York, 1989.

[186] M. Valle, D. D. Caviglia, and G. M. Bisio. Back-Propagation learning algorithms for analog VLSI implementation. In J. G. Delgado-Frias and W. Moore, editors, *VLSI for Neural Networks and Artificial Intelligence*, pages 35–44. Plenum Press, 1994.

[187] M. Valle, D. D. Caviglia, and G. M. Bisio. An experimental analog VLSI neural network with on-chip backpropagation learning. *Int. Journal of Analog Integ. Circ. Signal Proc.*, 9(3):231–245, 1996.

[188] M. Verleysen, B. Sirletti, A. M. Vandemeulebroecke, and P. G. A. Jespers. Neural networks for high-storage content-addressable memory: VLSI circuits and learning algorithm. *IEEE Journal of Solid-State Circuits*, 24:562–569, June 1989.

[189] M. Vidyasagar. Are analog neural networks better than binary neural networks? *Circuit Systems and Signal Processing*, 17(2):243–270, 1998.

[190] E. Vittoz. *Design of MOS VLSI Circuits for Telecommunications*, chapter Micropower techniques. Prentice-Hall, Englewood Chiffs, NJ, 1985.

[191] E. Vittoz, H. Oguey, M. A. Maher, O. Nys, E. Dijkstra, and M. Chevroulet. Analog storage of adjustable synaptic weights. In *VLSI Design of Neural Networks*, pages 47–63. Kluwer Academic, Norwell, MA, 1991.

[192] K. Wagner and D. Psaltis. Multilayer optical learning. *Applied Optics*, 26:5061–5076, 1987.

[193] S. S. Watkins, P. M. Chau, and R. Tawel. A radial basis function neurocomputer implemented with analog VLSI circuits. In *Proceedings of IEEE/INNS International Joint Conference on Neural Networks*, volume 2, pages 607–612, Baltimore, MD, June 1992.

[194] F. Werblin, T. Roska, and L. O. Chua. The analogic cellular neural network as a bionic eye. *International Journal of Circuit Theory and Applications*, 23(6):541–569, Nov-Dec 1995.

[195] B. Widrow and Bernard. An adaptive Adaline neuron using chemical Memisters. Technical Report 1553–2, Stanford Electronics Lab, 1960.

[196] B. Widrow and M. Lehr. 30 years of adaptive neural networks: perceptron, madaline and backpropagation. In *Proc. of the IEEE*, volume 78, pages 1415–1442. IEEE, 1990.

[197] C.-Y. Wu and C.-F. Chiu. A new structure for the silicon retina. In *Proc. of IEEE International Electron Devices Meeting*, pages 439–442, San Francisco, CA, December 1992.

[198] D. C. Wunsch, T. P. Caudell, C. D. Capps, R. J. Marks, and R. A. Falk. An optoelectronic implementation of the adaptive resonance neural network. *IEEE Transactions on Neural Networks*, 4(4):673–684, July 1993.

[199] Y. Xie and M. A. Jabri. Training algorithms for limited precision feedforward neural networks. Technical Report SEDAL TR 1991-8-3, School of EE, University of Sydney, Australia, 1991.

[200] Y. Xie and M. A. Jabri. Analysis of the effects of quantization in multilayer neural networks using a statistical model. *IEEE Transactions on Neural Networks*, 3(2):334–338, 1992.

[201] Y. Xie and M. A. Jabri. On the training of limited precision multi-layer perceptron. In *Proc. of the International Joint Conference on Neural Networks*, volume 3, pages 942–947, 1992.

[202] M. E. Zaghloul, J. L. Meador, and R. W. Newcomb, editors. *Silicon Implementation of Pulse Coded Neural Networks*. Kluwer Academic Publishers, 1994.