# Chapter 1
# Artificial Neurogenesis: An Introduction and Selective Review

**Taras Kowaliw, Nicolas Bredeche, Sylvain Chevallier and René Doursat**

**Abstract** In this introduction and review—like in the book which follows—we explore the hypothesis that adaptive growth is a means of producing brain-like machines. The emulation of neural development can incorporate desirable characteristics of natural neural systems into engineered designs. The introduction begins with a review of neural development and neural models. Next, artificial development—the use of a developmentally-inspired stage in engineering design—is introduced. Several strategies for performing this "meta-design" for artificial neural systems are reviewed. This work is divided into three main categories: bio-inspired representations; developmental systems; and epigenetic simulations. Several specific network biases and their benefits to neural network design are identified in these contexts. In particular, several recent studies show a strong synergy, sometimes interchangeability, between developmental and epigenetic processes—a topic that has remained largely under-explored in the literature.

T. Kowaliw (✉)
Institut des Systèmes Complexes - Paris Île-de-France, CNRS, Paris, France
e-mail: taras@kowaliw.ca

N. Bredeche
Sorbonne Universités, UPMC University Paris 06,
UMR 7222 ISIR,F-75005 Paris, France
e-mail: nicolas.bredeche@upmc.fr

N. Bredeche
CNRS, UMR 7222 ISIR,F-75005 Paris, France

S. Chevallier
Versailles Systems Engineering Laboratory (LISV), University of Versailles,
Velizy, France
e-mail: sylvain.chevallier@uvsq.fr

R. Doursat
School of Biomedical Engineering, Drexel University, Philadelphia, USA
e-mail: rene.doursat@drexel.edu

T. Kowaliw et al. (eds.), *Growing Adaptive Machines*,
Studies in Computational Intelligence 557, DOI: 10.1007/978-3-642-55337-0_1,

This book is about growing adaptive machines. By this, we mean producing programs that generate neural networks, which, in turn, are capable of learning. We think this is possible because nature routinely does so. And despite the fact that animals—those multicellular organisms that possess a nervous system—are staggeringly complex, they develop from a relatively small set of instructions. Accordingly, our strategy concerns the simulation of biological development as a means of *generating*, in contrast to directly designing, machines that can learn. By creating abstractions of the growth process, we can explore their contribution to neural networks from the viewpoint of complex systems, which self-organize from relatively simple agents, and identify model choices that will help us generate functional and useful artefacts. This pursuit is highly interdisciplinary: it is inspired by, and overlaps with, computational neuroscience, systems biology, machine learning, complex systems science, and artificial life.

Through growing adaptive machines, our ambition is also to contribute to a radical reconception of engineering. We want to focus on the design of component-level behaviour from which higher-level intelligent machines can emerge. The success of this "meta-design" [63] endeavour will be measured by our capacity to generate new learning machines: machines that scale, machines that adapt to novel environments, in short, machines that exhibit the richness we encounter in animals, but presently eludes artificial systems.

This chapter and the book that it introduces are centred around developmental and learning neural networks. It is a timely topic considering the recent resurgence of the neural paradigm as a major representation formalism in many technological areas, such as computer vision, signal processing, and robotic controllers, together with rapid progress in the modelling and applications of complex systems and highly decentralized processes. Researchers generally establish a distinction between *structural* design, focusing on the network topology, and *synaptic* design, defining the weights of the connections in a network [278]. This book examines how one could create a biologically inspired network structure capable of synaptic training, and blend synaptic and structural processes to let functionally suitable networks self-organize. In so doing, the aim is to recreate some of the natural phenomena that have inspired this approach.

The present chapter is organized as follows: it begins with a broad description of neural systems and an overview of existing models in computational neuroscience. This is followed by a discussion of *artificial development* and *artificial neurogenesis* in general terms, with the objective of presenting an introduction and motivation for both. Finally, three high-level strategies related to artificial neurogenesis are explored: first, *bio-inspired representations*, where network organization is inspired by empirical studies and used as a template for network design; then, *developmental simulation*, where networks grow by a process simulating biological embryogenesis; finally, *epigenetic simulation*, where learning is used as the main step in the design of the network. The contributions gathered in this book are written by experts in the field and contain state-of-the-art descriptions of these domains, including reviews of original research. We summarize their work here and place it in the context of the meta-design of developmental learning machines.

# 1 The Brain and Its Models

## *1.1 Generating a Brain*

Natural reproduction is, to date, the only one known way to generate true "intelligence". In humans, a mere six million ($6 \times 10^6$) base pairs, of which the majority is not directly expressed, code for an organism of some hundred trillion ($10^{14}$) cells. Assuming that a great part of this genetic information concerns neural development and function [253], it gives us a rough estimate of a brain-to-genome "compression ratio". In the central nervous system of adult humans, which contains approximately $8.5 \times 10^{10}$ neural cells and an equivalent number of non-neural (mostly glial) cells [8], this ratio would be of the order of $10^4$. However, the mind is not equal to its neurons, but considered to emerge from the specific synaptic connections and transmission efficacies between neurons [234, 255]. Since a neural cell makes contacts with $10^3$ other cells on average,[1] the number of connections in the brain reaches $10^{14}$, raising our compression ratio to $10^8$, a level beyond any of today's compression algorithms.

From there, one is tempted to infer that the brain is not as complex as it appears based solely on the number of its components, and even that something similar might be generated via a relatively simple parallel process. The brain's remarkable structural complexity is the result of several dynamical processes that have emerged over the course of evolution and are often categorized on four levels, based on their time scale and the mechanisms involved:

| level | time scale | change |
|---|---|---|
| phylogenic | generations | genetic: randomly mutated genes propagate or perish with the success of their organisms |
| ontogenic | days to years | cellular: cells follow their genetic instructions, which make them divide, differentiate, or die |
| epigenetic | seconds to days | cellular, connective: cells respond to external stimuli, and behave differently depending on the environment; in neurons, these changes include contact modifications and cell death |
| inferential | milliseconds to seconds | connective, activation: neurons send electrical signals to their neighbours, generating reactions to stimuli |

However, a strict separation between these levels is difficult in neural development and learning processes.[2] Any attempt to estimate the phenotype-to-genotype com-

[1] Further complicating this picture are recent results showing that these connections might themselves be information processing units, which would increase this estimation by several orders of magnitude [196].

[2] By epigenetic, we mean here any heritable and non-genetic changes in cellular expression. (The same term is also used in another context to refer strictly to DNA methylation and transcription-level mechanisms.) This includes processes such as learning for an animal, or growing toward a light source for a plant. The mentioned time scale represents a rough average over cellular responses to environmental stimuli.

pression ratio must also take into account epigenetic, not just genetic, information. More realistic or bio-inspired models of brain development will need to include models of environmental influences as well.

## *1.2 Neural Development*

We briefly describe in this section the development of the human brain, noting that the general pattern is similar in most mammals, despite the fact that size and durations vastly differ. A few weeks after conception, a sheet of cells is formed along the dorsal side of the embryo. This neural plate is the source of all neural and glial cells in the future body. Later, this sheet closes and creates a neural tube whose anterior part develops into the brain, while the posterior part produces the spinal cord. Three bulges appear in the anterior part, eventually becoming the forebrain, midbrain, and hindbrain. A neural crest also forms on both sides of the neural tube, giving rise to the nervous cells outside of the brain, including the spinal cord. After approximately eight weeks, all these structures can be identified: for the next 13-months they grow in size at a fantastic rate, sometimes generating as many as 500,000 neurons per minute.

Between three to six months after birth, the number of neurons in a human reaches a peak. Nearly all of the neural cells used throughout the lifetime of the individual have been produced [69, 93]. Concurrently, they disappear at a rapid rate in various regions of the brain as programmed cell death (apoptosis) sets in. This overproduction of cells is thought to have evolved as a competitive strategy for the establishment of efficient connectivity in axonal outgrowth [34]. It is also regional: for instance, neural death comes later and is less significant in the cortex compared to the spinal cord, which loses a majority of its neurons before birth.

Despite this continual loss of neurons, the total brain mass keeps increasing rapidly until the age of three in humans, then more slowly until about 20. This second peak marks a reversal of the trend, as the brain now undergoes a gradual but steady loss of matter [53]. The primary cause of weight increase can be found in the connective structures: as the size of the neurons increase, so does their dendritic tree and glial support. Most dendritic growth is postnatal, but is not simply about adding more connections: the number of synapses across the whole brain also peaks at eight months of age. Rather, mass is added in a more selective manner through specific phases of neural, dendritic, and glial development.

These phenomena of maturation—neural, dendritic, and glial growth, combined with programmed cell death—do not occur uniformly across the brain, but regionally. This can be measured by the level of myelination, the insulation provided by glial cells that wrap themselves around the axons and greatly improve the propagation of membrane potential. Taken as an indication of more permanent connectivity, myelination reveals that maturation proceeds in the posterior-anterior direction: the

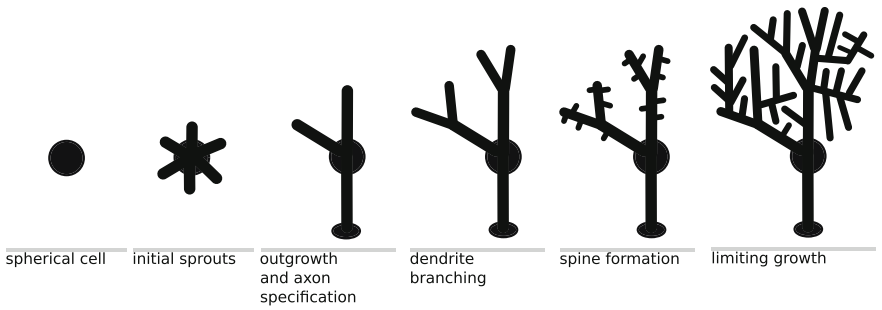


**Fig. 1** Illustration of the general steps in neural dendritic development

spinal cord and brain stem (controlling vital bodily function) are generally mature at birth, the cerebellum and midbrain mature in the few months following birth, and after a couple of years the various parts of the forebrain also begin to mature. The first areas to be completed concern sensory processing, and the last ones are the higher-level "association areas" in the frontal cortex, which are the site of myelination and drastic reorganization until as late as 18-years old [69]. In fact, development in mammals never ends: dendritic growth, myelination, and selective cell death continue throughout the life of an individual, albeit at a reduced pace.

### 1.2.1 Neuronal Morphology

Neurons come in many types and shapes. The particular geometric configuration of a neural cell affects the connectivity patterns that it creates in a given brain region, including the density of synaptic contacts with other neurons and the direction of signal propagation. The shape of a neuron is determined by the outgrowth of *neurites*, an adaptive process steered by a combination of genetic instructions and environmental cues.

Although neurons can differ greatly, there are general steps in dendritic and axonal development that are common to many species. Initially, a neuron begins its life as a roughly spherical body. From there, neurites start sprouting, guided by *growth cones*. Elongation works by addition of material to relatively stable spines. Sprouts extend or retract, and one of them ultimately self-identifies as the cell's axon. Dendrites then continue to grow out, either from branching or from new dendritic spines that seem to pop up randomly along the membrane. Neurites stop developing, for example, when they have encountered a neighbouring cell or have reached a certain size. These general steps are illustrated in Fig. 1 [230, 251].

Dendritic growth is guided by several principles, generally thought to be controlled regionally: a cell's dendrites do not connect to other specific cells but, instead, are drawn to regions of the developing brain defined by diffusive signals. Axonal growth

tends to be more nuanced: some axons grow to a fixed distance in the direction of a simple gradient; others grow to long distances in a multistage process requiring a large number of guidance cells. While dendritic and axonal development is most active during early development, by no means does it end at maturity. The continual generation of dendritic spines plays a crucial role throughout the lifetime of an organism.

Experiments show that neurons isolated in cultures will regenerate neurites. It is also well known that various extracellular molecules can promote, inhibit, or otherwise bias neurite growth. In fact, there is evidence that in some cases context alone can be sufficient to trigger differentiation into specific neural types. For example, the introduction of catalysts can radically alter certain neuron morphologies to the point that they transform into other morphologies [230]. This has important consequences on any attempt to classify and model neural types [268].

In any case, the product of neural growth is a network possessing several key properties that are thought to be conducive to learning. It is an open question in neuroscience how much of neural organization is a result of genetic and epigenetic targeting, and how much is pure randomness. However, it is known that on the mesoscopic scale, seemingly random networks have consistent properties that are thought to be typical of effective networks. For instance, in several species, cortical axonal outgrowth can be modelled by a gamma distribution. Moreover, cortical structures in several species have properties such as relatively high clustering along certain axes, but not other axes [28, 146]. Cortical connectivity patterns are also "small-world" networks (with high local specialization, and minimal wiring lengths), which provide efficient long-range connections [263] and are probably a consequence of dense packing constraints inside a small space.

### 1.2.2 Neural Plasticity

There are also many forms of plasticity in a nervous system. While neural cell behaviour is clearly different during development and maturity (for instance, the drastic changes in programmed cell death), many of the same mechanisms are at play throughout the lifetime of the brain. The remaining differences between developmental and mature plasticity seem to be regulated by a variety of signals, especially in the extracellular matrix, which trigger the end of sensitive periods and a decrease in spine formation dynamics [230].

Originally, it was Hebb who postulated in 1949 what is now called *Hebbian learning*: repeated simultaneous activity (understood as mean-rate firing) between two neurons or assemblies of neurons reinforces the connections between them, further encouraging this co-activity. Since then, biologists have discovered a great variety of mechanisms governing synaptic plasticity in the brain, clearly establishing reciprocal causal relations between wiring patterns and firing patterns. For example, long-term potentiation (LTP) and long-term depression (LTD) refer to ositiveor negative

changes in the probability of successful signal transmission from a resynapticaction potential to the generation of a postsynaptic potential. These "long-term" changes can last for several minutes, but are generally less pronounced over hours or days [230]. Prior to synaptic efficacies, synaptogenesis itself can also be driven by activity-dependent mechanisms, as dendrites "seek out" appropriate partner axons in a process that can take as little as a few hours [310]. Other types of plasticity come from glial cells, which stabilize and accelerate the propagation of signals along mature axons (through myelination and extracellular regulation), and can also depend on activity [135].

Many others forms and functions of plasticity are known, or assumed, to exist. For instance, "fast synaptic plasticity", a type of versatile Hebbian learning on the 1-ms time scale, was posited by von der Malsburg [286–288]. Together with a neural code based on temporal correlations between units rather than individual firing rates, it provides a theoretical framework to solve the well-known "binding problem", the question of how the brain is able to compose sensory information into multi-feature concepts without losing relational information. In collaboration with Bienenstock and Doursat, this assumption led to a format of representation using graphs, and models of pattern recognition based on *graph matching* [19–21]. Similarly, "spike-timing dependent plasticity" (STDP) describes the dependence of transmission efficacies between connected neurons on the *ordering* of neural spikes. Among other effects, this allows for pre-synaptic spikes which precede post-synaptic spikes to have greater influence on the resulting efficacy of the connection, potentially capturing a notion of causality [183]. It is posited that Hebbian-like mechanisms also operate on non-neural cells or neural groups [310]. "Metaplasticity" refers to the ability of neurons to alter the threshold at which LTP and LTD occur [2]. "Homeostatic plasticity" refers to the phenomenon where groups of neurons self-normalize their own level of activity [208].

### 1.2.3 Theories of Neural Organization

Empirical insights into mammalian brain development have spawned several theories regarding neural organization. We briefly present three of them in this section: *nativism*, *selectivism*, and *neural constructivism*.

The nativist view of neural development posits a strong genetic role in the construction of cognitive function. It claims that, after millions of years of evolutionary shaping, development is capable of generating highly specialized, innate neural structures that are appropriate for the various cognitive tasks that humans accomplish. On top of these fundamental neural structures, details can be adjusted by learning, like parameters. In cognitive science, it is argued that since children learn from a relative poverty of data (based on single examples and "one-shot learning"), there must be a native processing unit in the brain that preexists independently of environmental influence. Famously, this hypothesis led to the idea of a "universal grammar" for language [36], and some authors even posit that *all* basic concepts are innate [181]. According to a neurological (and controversial) theory, the cortex

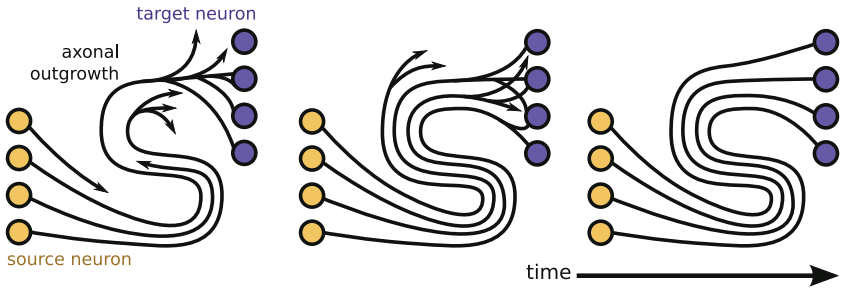


**Fig. 2** Illustration of axonal outgrowth: initial overproduction of axonal connections and competitive selection for efficient branches leads to a globally efficient map (adapted from [294])

is composed of a repetitive lattice of nearly identical "computational units", typically identified with cortical columns [45]. While histological evidence is unclear, this view seems to be supported by physiological evidence that cortical regions can adapt to their input sources, and are somewhat interchangeable or "reusable" by other modalities, especially in vision- or hearing-impaired subjects. Recent neuro-imaging research on the mammalian cortex has revived this perspective. It showed that cortical structure is highly regular, *even across species*: fibre pathways appear to form a rectilinear 3D grid containing parallel sheets of interwoven paths [290]. Imaging also revealed the existence of arrays of assemblies of cells whose connectivity is highly structured and predictable *across species* [227]. Both discoveries suggest a significant role for regular and innate structuring in cortex layout (Fig. 2).

In contrast to nativism, selectivist theories focus on *competitive mechanisms* as the lead principle of structural organization. Here, the brain initially overproduces neurons and neural connections, after which plasticity-based competitive mechanisms choose those that can generate useful representations. For instance, theories such as Changeux's "selective stabilization" [34] and Katz's "epigenetic population matching" [149] describe the competition in growing axons for postsynaptic sites, explaining how the number of projected neurons matches the number of available cells. The quantity of axons and contacts in an embryo can also be artificially decreased or increased by excising target sites or by surgically attaching supernumerary limbs [272]. This is an important reason for the high degree of evolvability of the nervous system, since adaptation can be easily obtained under the same developmental mechanisms without the need for genetic modifications.

The regularities of neocortical connectivity can also be explained as a self-organization process during pre- and post-natal development via epigenetic factors such as ongoing biochemical and electrophysiological activity. These principles have been at the foundation of biological models of "topographically ordered mappings", i.e. the preservation of neighborhood relationships between cells from one sheet to another, most famously the bundle of fibers of the "retinotopic projection" from the retina to the visual cortex, via relays [293]. Bienenstock and Doursat have also proposed a model of selectivist self-structuration of the cortex [61, 65],

showing the possibility of simultaneous emergence of ordered chains of synaptic connectivity together with wave-like propagation of neuronal activity (also called "synfire chains" [1]). Bednar discusses an alternate model in Chap. 7.

A more debated selectivist hypothesis involves the existence of "epigenetic cascades" [268], which refer to a series of events driven by epigenetic population-matching that affect successive interconnected regions of the brain. Evidence for phenomena of epigenetic cascades is mixed: they seem to exist in only certain regions of the brain but not in others. The selectivist viewpoint also leads to several intriguing hypotheses about brain development over the evolutionary time scale. For instance, Ebbesson's "parcellation hypothesis" [74] is an attempt to explain the emergence of specialized brain regions. As the brain becomes larger over evolutionary time, the number of inter-region connections increases but due to competition and geometric constraints, these connections will preferentially target neighbouring regions. Therefore, the increase in brain mass will tend to form "parcels" with specialized functions. Another hypothesis is Deacon's "displacement theory" [51], which tries to account for the differential enlargement and multiplication of cortical areas.

More recently, the neural constructivism of Quartz and Sejnowski [234] casts doubt on both the nativist and selectivist perspectives. First, the developing cortex appears to be free of functionally specialized structures. Second, finer measures of neural diversity, such as type-dependent synapse counts or axonal/dendritic arborization, provide a better assessment of cognitive function than total quantities of neurons and synapses. According to this view, development consists of a long period of dendritic development, which slowly generates a neural structure mediated by, and appropriately biased toward, the environment.

These three paradigms highlight principles that are clearly at play in one form or another during brain development. However, their relative merits are still a subject of debate, which could be settled through modelling and computational experiments.

## *1.3 Brain Modelling*

Computational neuroscience promotes the theoretical study of the brain, with the goal of uncovering the principles and mechanisms that guide the organization, information-processing and cognitive abilities of the nervous system [278]. A great variety of brain structures and functions have already been the topic of many modelling and simulation works, at various levels of abstraction or data-dependency. Models range from the highly detailed and generic, where as many possible phenomena are reproduced in as much detail as possible, to the highly abstract and specific, where the focus is one particular organization or behaviour, such as feed-forward neural networks. These different levels and features serve different motivations: for example, concrete simulations can try to predict the outcome of medical treatment, or demonstrate the generic power of certain neural theories, while abstract systems are the tool of choice for higher-level conceptual endeavours.

In contrast with the majority of computational neuroscience research, our main interest with this book, as exposed in this introductory chapter, resides in the potential to *use brain-inspired mechanisms for engineering challenges*.

### 1.3.1 Challenges in Large-Scale Brain Modelling

Creating a model and simulation of the brain is a daunting task. One immediate challenge is the scale involved, as billions of elements are each interacting with thousands of other elements nonlinearly. Yet, there have already been several attempts to create large-scale neural simulations (see reviews in [27, 32, 95]). Although it is a hard problem, researchers remain optimistic that it will be possible to create a system with sufficient resources to mimic all connections in the human brain within a few years [182]. A prominent example of this trend is the Blue Brain project, whose ultimate goal is to reconstruct the entire brain numerically at a molecular level. To date, it has generated a simulation of an array of cortical columns (based on data from the rat) containing approximately a million cells. Among other applications, this project allows generating and testing hypotheses about the macroscopic structures that result from the collective behaviours of instances of neural models [116, 184]. Other recent examples of large-scale simulations include a new proof-of-concept using the Japanese K computer simulating a (non-functional) collection of nearly $2 \times 10^9$ neurons connected via $10^{12}$ synapses [118], and Spaun, a more functional system consisting of $2.5 \times 10^6$ neurons and their associated connections. Interestingly, Spaun was created by top-down design, and is capable of executing several different functional behaviours [80]. With the exception of one submodule, however, Spaun does not "learn" in a classical sense.

Other important challenges of brain simulation projects, as reviewed by Cattell and Parker [32], include neural diversity and complexity, interconnectivity, plasticity mechanisms in neural and glial cells, and power consumption. Even more critically, the fast progress in computing resources able to support massive brain-like simulations is not any guarantee that such simulations will behave "intelligently". This requires a much greater understanding of neural behaviour and plasticity, at the individual and population scales, than what we currently have. After the recent announcements of two major funded programs, the EU Human Brain Project and the US Brain Initiative, it is hoped that research on large-scale brain modelling and simulation should progress rapidly.

### 1.3.2 Machine Learning and Neural Networks

Today, examples of abstract learning models are legion, and *machine learning* as a whole is a field of great importance attracting a vast community of researchers. While some learning machines bear little resemblance to the brain, many are inspired by their natural source, and a great part of current research is devoted to reverse-engineering natural intelligence.

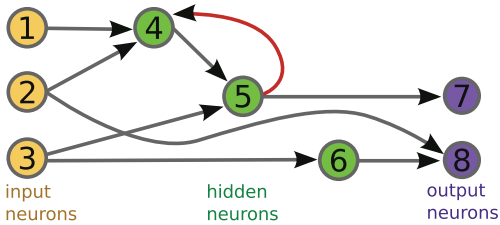


**Fig. 3** Example of neural network with three *input neurons*, three *hidden neurons*, two *output neurons*, and nine connections. One feedback connection (*5→4*) creates a cycle. Therefore, this is a recurrent NN. If that connection was removed, the network would be feed-forward only

**Chapter 2: A brief introduction to probabilistic machine learning and its relation to neuroscience.**

In Chap. 2, Trappenberg provides an overview of the most important ideas in modern machine learning, such as support vector machines and Bayesian networks. Meant as an introduction to the probabilistic formulation of machine learning, this chapter outlines a contemporary view of learning theories across three main paradigms: unsupervised learning, close to certain developmental aspects of an organism, supervised learning, and reinforcement learning viewed as an important generalization of supervised learning in the temporal domain. Beside general comments on organizational mechanisms, the author discusses the relations between these learning theories and biological analogies: unsupervised learning and the development of filters in early sensory cortical areas, synaptic plasticity as the physical basis of learning, and research that relates models of basal ganglia to reinforcement learning theories. He also argues that, while lines can be drawn between development and learning to distinguish between different scientific camps, this distinction is not as clear as it seems since, ultimately, all model implementations have to be reflected by some morphological changes in the syste [279].

In this book, we focus on neural networks (NNs). Of all the machine learning algorithms, NNs provide perhaps the most direct analogy with the nervous system. They are also highly effective as engineering systems, often achieving state-of-the-art results in computer vision, signal processing, speech recognition, and many other areas (see [113] for an introduction). In what follows, we introduce a summary of a few concepts and terminology.

For our purposes, a neural network consists of a graph of neurons indexed by $i$. A connection $i \rightarrow j$ between two neurons is directed and has a weight $w_{ij}$. Typically, input neurons are application-specific (for example, sensors), output neurons are desired responses (for example, actuators or categories), and hidden neurons are information processing units located in-between (Fig. 3).

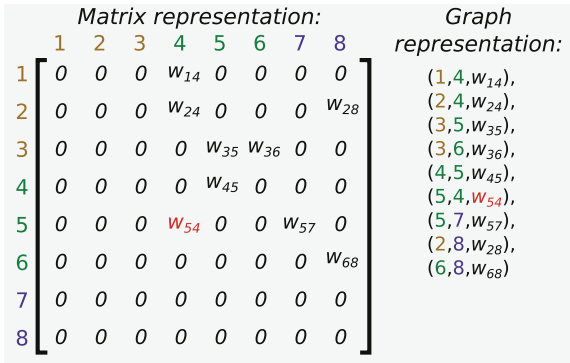


**Fig. 4** Two representations for the neural network of Fig. 3

A neural network typically processes signals propagating through its units: a vector of floating-point numbers, $s$, originates in input neurons and resulting signals are transmitted along the connections. Each neuron $j$ generates an output value $v_j$ by collecting input from its connected neighbours and computing a weighted sum via an *activation function*, $\varphi$:

$$v_j(s) = \varphi\left(\sum_{i \mid (i \to j)} w_{ij}\, v_i(s)\right)$$

where $\varphi(x)$ is often a sigmoid function, such as $\tanh(x)$, making the output nonlinear. For example, in the neural network of Fig. 3, the output of neuron 8 can be written in terms of input signals $v_1$, $v_2$, $v_3$ as follows:

$$\begin{aligned} v_8(s) &= \varphi(w_{28}\, v_2 + w_{68}\, v_6) \\ &= \varphi(w_{28}\, v_2 + w_{68}\, \varphi(w_{36}\, v_3)) \end{aligned}$$

Graph topologies without cycles are known as *feedforward* NNs, while topologies with cycles are called *recurrent* NNs. The former are necessarily stateless machines, while the latter might possess some memory capacity. With sufficient size, even simple feed-forward topologies can approximate any continuous function [44]. It is possible to build a Turing machine in a recurrent NN [260].

A critical question in this chapter concerns the *representation* format of such a network. Two common representations are adjacency matrices, which list every possible connection between nodes, and graph-based representations, typically represented as a list of nodes and edges (Fig. 4). Given sufficient space, any NN topology and set of weights can be represented in either format.

Neural networks can be used to solve a variety of problems. In classification or regression problems, when examples of input-output pairs are available to the network during the learning phase, the training is said to be *supervised*. In this scenario, the fitness function is typically a mean square error (MSE) measured between the

network outputs and the actual outputs over the known examples. With feedback available for each training signal sent, NNs can be trained through several means, most often via gradient descent (as in the "backpropagation" algorithm). Here, a error or "loss function" $E$ is defined between the desired and actual responses of the network, and each weight is updated according to the derivative of that function:

$$w_{ij}(t+1) = w_{ij}(t) - \eta \frac{\partial E}{\partial w_{ij}}$$

where $\eta$ is the learning rate. Generally, this kind of approach assumes a fixed topology and its goal is to optimize the weights.

On the other hand, *unsupervised learning* concerns cases where no output samples are available and data-driven self-organization mechanisms are at work, such as Hebbian learning. Finally, *reinforcement learning* (including neuroevolution) is concerned with delayed, sparse and possibly noisy rewards. Typical examples include robotic control problems, decision problems, and a large array of inverse problems in engineering. These various topics will be discussed later.

### 1.3.3 Brain-Like AI: What's Missing?

It is generally agreed that, at present, artificial intelligence (AI) is not "brain-like". While AI is successful at many specialized tasks, none of them shows the versatility and adaptability of animal intelligence. Several authors have compiled a list of "missing" properties, which would be necessary for brain-like AI. These include: the capacity to engage in a behavioural tasks; control via a simulated nervous system; continuously changing self-defined representations; and embodiment in the real world [165, 253, 263, 292]. Embodiment, especially, is viewed as critical because by exploiting the richness of information contained in the morphology and the dynamics of the body and the environment, intelligent behaviour could be generated with far less representational complexity [228, 291].

The hypothesis explored in this book is that *the missing feature is development*. The brain is not built from a blueprint; instead, it grows *in situ* from a complex multicellular process, and it is this adaptive growth process that leads to the adaptive intelligence of the brain. Our goal is not to account for all properties observed in nature, but rather to *identify the relevance of a developmental approach with respect to an engineering objective* driven by performance alone. In the remainder of this chapter, we review several approaches incorporating developmentally inspired strategies into artificial neural networks.

## 2 Artificial Development

There are about 1.5 million known species of multicellular organisms, representing an extraordinary diversity of body plans and shapes. Each individual grows from the division and self-assembly of a great number of cells. Yet, this developmental

process also imposes very specific constraints on the space of possible organisms, which restricts the evolutionary branches and speciation bifurcations. For instance, bilaterally symmetric cellular growth tends to generate organisms possessing pairs of limbs that are equally long, which is useful for locomotion, whereas asymmetrical organisms are much less frequent.

While the "modern synthesis" of genetics and evolution focused most of the attention on selection, it is only during the past decade that analyzing and understanding variation by comparing the developmental processes of different species, at both embryonic and genomic levels, became a major concern of evolutionary development, or "evo-devo". To what extent are organisms also the product of self-organized physicochemical developmental processes not necessarily or always controlled by complex underlying genetics? Before and during the advent of genetics, the study of developmental structures had been pioneered by the "structuralist" school of theoretical biology, which can be traced back to Goethe, D'Arcy Thompson, and Waddington. Later, it was most actively pursued and defended by Kauffman [150] and Goodwin [98] under the banner of *self-organization*, argued to be an even greater force than natural selection in the production of viable diversity.

By *artificial development* (AD), also variously referred to as artificial embryogeny, generative systems, computational ontogeny, and other equivalent expressions (see early reviews in [107, 265]), we mean the attempt to reproduce the constraints and effects of self-organization in automated design. Artificial development is about creating a growth-inspired process that will bias design outcomes toward useful forms or properties. The developmental engineer engages in a form of "meta-design" [63], where the goal is not to design a system directly but rather set a framework in which human design or automated search will specify a process that can generate a desired result. The benefits and effectiveness of development-based design, both in natural and artificial systems, became an active topic of research only recently and are still being investigated.

Assume for now that our goal is to generate a design which maximizes an objective function, $o$: $\Phi \rightarrow \mathbb{R}^n$, where $\Phi$ is the "phenotypic" space, that is, the space of potential designs, and $\mathbb{R}^n$ is a collection of performance assessments, as real values, with $n \geq 1$ ($n = 1$ denotes a single-objective problem, while $n > 1$ denotes a multiobjective problem). A practitioner of AD will seek to generate a lower-level "genetic" space $\Gamma$, a space of "environments" $E$ in which genomes will be expressed, and a dynamic process $\delta$ that transforms the genome into a phenotype:

$$\Gamma \times E \xrightarrow{\delta} \Phi \xrightarrow{o} \mathbb{R}^n$$

In many cases, only one environment is used, usually a trivial or empty instance from the phenotypic space. In these cases, we simply write:

$$\Gamma \xrightarrow{\delta} \Phi \xrightarrow{o} \mathbb{R}^n$$

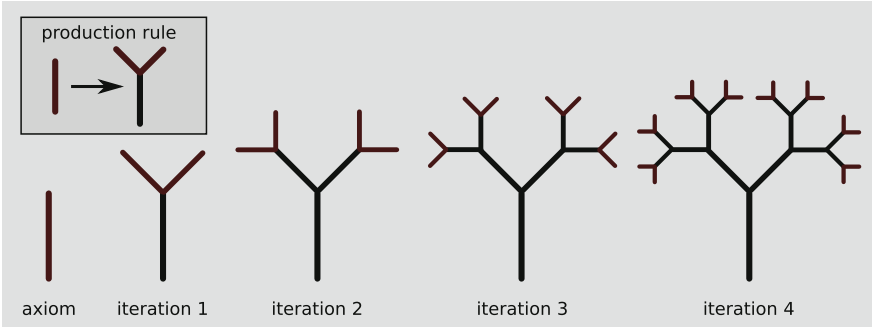


**Fig. 5** Visualization of an L-System. *Top-left* a single production rule (the "genome"). *Bottom-left* the axiom (initial "word"). Recursive application of the production rule generates a growing structure (the "phenotype"). In this case, the phenotype develops exponentially with each application of the production rule

The dynamic process $\delta$ is inspired by biological embryogenesis, but need not resemble it. Regardless, we will refer to it as growth or development, and to the quadruple $(\Gamma, E, \delta, \Phi)$ as an *AD system*.

Often, the choice of phenotypic space $\Phi$ is dictated by the problem domain. For instance, to design neural networks, one might specify $\Phi$ as the space of all adjacency matrices, or perhaps as all possible instances of some data structure corresponding to directed, weighted graphs. Or to design robots, one might define $\Phi$ as all possible lattice configurations of a collection of primitive components and actuators. Sometimes there is value in restricting $\Phi$, for example to exclude nonsensical or dangerous configurations. It is the engineer's task to choose an appropriate $\Phi$ and to "meta-design" the $\Gamma$, $E$, and $\delta$ parts that will help import the useful biases of biological growth into evolved systems.

A famous class of AD systems are the so-called L-Systems. These are formal grammars originally developed by Lindenmayer as a means of generating model plants [231]. In their simplest form, they are context-free grammars, consisting of a starting symbol, or "axiom", a collection of variables and constants, and at most one production rule per variable. By applying the production rules to the axiom, a new and generally larger string of symbols, or "word", is created. Repeated application of the production rules to the resulting word simulates a growth process, often leading to gradually more complex outputs. One such grammar is illustrated in Fig. 5, where a single variable (red stick) develops into a tree-like shape. In this case, the space of phenotypes $\Phi$ is the collection of all possible words (collections of sticks), the space of genotypes $\Gamma$ is any nonambiguous set of context-free production rules, the environment $E$ is the space in which a phenotype exists (here trivially 2D space), and the dynamic process $\delta$ is the repeated application of the rules to a given phenotype.

There are several important aspects to the meta-design of space of representations $\Gamma$ and growth process $\delta$. Perhaps the most critical requirement is that the chosen entities be "evolvable". This term has many definitions [129] but generally means that

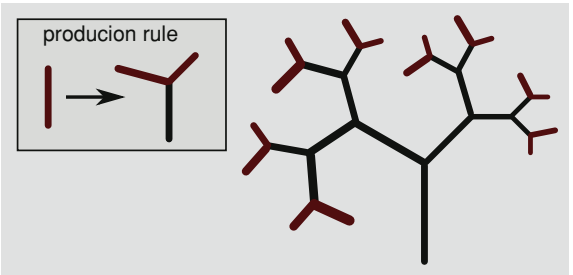


**Fig. 6** A mutation of the production rule in Fig. 5, and the output after four iterations of growth

**Fig. 7** McCormack's evolved L-Systems, inspired by, but exaggerating, Australian flora

the space of representations should be easily searchable for candidates that optimize some objective. A generally desirable trait is that small changes in a representation should lead to small changes in the phenotype—a "gentle slope" allowing for incremental search techniques. In AD systems, however, due to the nonlinear dynamic properties of the transformation process, it is not unusual for small genetic changes to have large effects on the phenotype [87].

For instance, consider in Fig. 6 a possible *mutation* of the previous L-System. Here, the original genome has undergone a small change, which has affected the resulting form. The final phenotypes from the original and the mutated version are

similar in this case: they are both trees with an identical topology. However, it is not difficult to imagine mutations that would have catastrophic effects, resulting in highly different forms, such as straight lines or self-intersections. Nonlinearity of the genotype-to-phenotype mapping $\delta$ can be at the same time a strength and a weakness in design tasks.

There is an important distinction to be made here between our motivations and those of systems biology or computational neuroscience. In AD, we seek means of creating engineered designs, not simulating or reproducing biological phenomena. Perhaps this is best illustrated via an example: McCormack, a computational artist, works with evolutionary computation and L-Systems (Fig. 7). Initially, this involved the generation of realistic models of Australian flora. Later, however, he continued to apply evolutionary methods to create exaggerations of real flora, artefacts that he termed "impossible nature" [187, 188]. McCormack's creations retain salient properties of flora, especially the ability to inspire humans, but do not model any existing organism.

## *2.1 Why Use Artificial Development?*

Artificial development is one way of approaching *complex systems engineering*, also called "emergent engineering" [282]. It has been argued that the traditional state-based approach in engineering has reached its limits, and the principles underlying complex systems—self-organization, nonlinearity, and adaptation—must be accommodated in new engineering processes [11, 203]. Incorporating complex systems into our design process is necessary to overcome our present logjam of complexity, and open new areas of productivity. Perhaps the primary reason for the interest in simulations of development is that natural embryogenesis is a practical example of complex systems engineering, one which achieves designs of scale and functionality that modern engineers aspire to. There are several concrete demonstrations of importing desirable properties from natural systems into artificial counterparts. The key property of evolvability, which we have already discussed, is linked to a notion of scalability. Other related properties include robustness via self-repair and plasticity.

### 2.1.1 Scalability

Perhaps the best studied property of AD systems is the ability to scale to several sizes. This is a consequence of a general decoupling of the complexity of the genome (what we are searching for) from the phenotype (the final product). In many models, the size of the phenotype is controlled via a single parameter, which can be the number of repetitions of a module, the number of iterations in an L-System, or a single variable

controlling the amount of available resources. In these cases, a minimal change in the size of the genome might have exponential effects on the size of the resulting phenotype.

This property—the capacity to scale—brings to mind the notion of "Kolmogorov complexity", or the measurement of the complexity of a piece of data by the shortest computer program that generates it. With the decision to use AD, we make the assumption that there exists a short computer program that can generate our desired data, i.e. that the Kolmogorov complexity of our problem is small. This implies that AD will succeed in cases where the data to be generated is sufficiently large and non-random. Unfortunately, in the general case, finding such a program for some given data is an uncomputable problem, and to date there is no good approximation other than enumerating all possible programs, a generally untenable solution [173].

In many highly relevant domains of application, the capacity for scaling has been successfully demonstrated by AD systems. Researchers will often compare their AD model to a *direct encoding* model, in which each component of the solution is specified in the genome independently. Abstract studies have confirmed our intuition that AD systems are often better for *large* phenotypes and *nonrandom* data [40, 108]. This has also been demonstrated in neural networks [86, 104, 153], virtual robotics [161]; engineering design [127], and other domains [17, 243].

### 2.1.2 Robustness and Self-repair

Another desirable property of biological systems is the capacity for robustness. By this, we mean a "canalization" or the fact that a resulting phenotype is resistant to environmental perturbations, whether they are obstacles placed in the path of a developing organism, damage inflicted, or small changes to external factors affecting cellular expression, such as temperature or sources of nutrient. In biology, this ability is hypothesized to result from a huge number of almost identical cells, a redundancy creating tolerance toward differences in cellular arrangement, cell damage, or the location of organizers [152]. Several AD systems have been shown to import robustness, which can be selected for explicitly [18]. More interestingly, robustness is often imported without the inclusion of selection pressure [86, 161, 243]. In many cases, this property seems to be a natural consequence of the use of an adaptive growth process as a design step.

An extreme example of robustness is the capacity for self-repair. Many authors have conducted experiments with AD systems in which portions of an individual are damaged (e.g. by scrambling or removing components). In these cases, organisms can often self-repair, reconfiguring themselves to reconstruct the missing or altered portions and optimize the original objective. For instance, this has been demonstrated in abstract settings [5, 42, 145, 197], digital circuits [224], and virtual robotics [275]. Interestingly, in most of these cases, the self-repair capacity is not explicitly selected for in the design stage.

### 2.1.3 Plasticity

Another property of AD systems is plasticity, also referred to as polymorphism or polyphenism (although these terms are not strictly equivalent). By this, we mean the ability of organisms to be influenced by their environment and adopt as a result any phenotype from a number of possibilities. Examples in nature are legion [94], and most striking in the tendency of plants to grow toward light or food, or the ability of nervous systems to adapt to new stimuli. While robustness means reaching the same genotype under perturbation, plasticity means reaching *different* phenotypes under perturbation. Both, however, serve to improve the ultimate fitess of the organism in a variety of environments.

In classical neural systems, plasticity is the norm and is exemplified by well-known training methods: Hebbian learning, where connections between neurons are reinforced according to their correlation under stimuli [114], and backpropagation, where connection weights are altered according to an error derivative associated with incoming stimuli [245]. These classic examples focus on synaptic structure, or the weighting of connections in some predetermined network topology. While this is certainly an element of natural self-organization, it is by no means a complete characterization of the role that plasticity plays in embryogenesis. Environmental stimuli in animal morphogenesis include other neural mechanisms, such as the constant reformation and re-connection of synapses. Both selectivist and constructivist theories of brain development posit a central role for environmental stimuli in the generation of neural morphology. Furthermore, plasticity plays a major role in other developmental processes as well. In plants, the presence or absence of nutrients, light, and other cues will all but determine the coarse morphology of the resulting form. In animals, cues such as temperature, abundance of nutrients, mechanical stress, and available space are all strong influences. Indeed, the existence of plasticity is viewed as a strong factor in the evolvability of forms: for instance, plastic mechanisms in the development of the vascular system allow for a sort of "accidental adaptation", where novel morphological structures are well served by existing genetic mechanisms for vasculogenesis, despite never being directly selected for in evolutionary history [99, 177].

Most examples of artificial neural systems exploit plasticity mechanisms to tune parameters according to some set of "training" stimuli. Despite this, the use of environmentally induced plasticity in AD systems is rare. Only a few examples have shown that environmental cues can be used to reproduce plasticity effects commonly seen in natural phenomena, such as: virtual plant growth [87, 252], circuit design [280], or other scenarios [157, 190]. In one case, Kowaliw et al. experimented with the growth of planar trusses, a model of structural engineering. They initially showed that the coarse morphology of the structures could be somewhat controlled by the choice of objective function—however, this was also a difficult method of morphology specification [163]. Instead, the authors experimented with external constraints, which consisted of growing their structures in an environment that had the shape of the desired morphology. Not only was this approach generally successful in the sense of generating usable structures of the desired overall shape, but it

also spontaneously generated results indicating evolvability. A few of the discovered genomes could grow successful trusses not only in the specific optimization environment but also in *all* the other experimental environments, thus demonstrating a capacity for accidental adaptation [162].

### 2.1.4 Other Desirable Natural Properties

Other desirable natural properties are known to occasionally result from AD systems. These include: graceful degradation, i.e. the capacity for systems performance to fail continuously with the removal of parts [18]; adaptation to previously unseen environments, thought to be the result of repetitions of phenotypic patterns capturing useful regularities (see, for instance, Chap. 9 [206]); and the existence of "scafolding", i.e. a plan for the construction of the design in question, based on the developmental growth plan [241].

## *2.2 Models of Growth*

An AD system requires a means of converting a representation into a design. This conversion typically involves a dynamic process that generates an arrangement of "cells", where these cells can stand for robotic components, structural members, neurons, and so on. Several models of multi-component growth have been investigated in detail:

- **Induced representational bias**: the designer adds a biologically inspired bias to an otherwise direct encoding. Examples include very simple cases, such as mirroring elements of the representation to generate symmetries in the phenotype [256], or enforcing a statistical property inspired by biological networks, such as the density of connections in a neural system [258].
- **Graph rewriting**: the phenotype is represented as a graph, the genome as a collection of graph-specific actions, and growth as the application of rules from the genome to some interim graph. Examples of this paradigm include L-Systems and dynamic forms of genetic programming [109, 122].
- **Cellular growth models**: the phenotype consists of a collection of cells on a lattice or in continuous space. The genome consists of logic that specifies associations between cell neighbourhoods and cell actions, where the growth of a phenotype involves the sum of the behaviours of cells. Cellular growth models are sometimes based on variants of *cellular automata*, a well-studied early model of discrete dynamics [161, 197]. This choice is informed by the success of cellular automata in the simulation of natural phenomena [56]. Other models involve more plausible physical models of cellular interactions, where cells orient themselves via intercellular physics [25, 62, 76, 144, 249]

- **Reaction-diffusion models**: due to Turing [281], they consist of two or more simulated chemical agents interacting on a lattice. The chemical interactions are modelled as nonlinear differential equations, solved numerically. Here, simple equations quickly lead to remarkable examples of self-organized patterns. Reaction-diffusion models are known to model many aspects of biological development, including overall neural organization [172, 259] and organismal behaviour [47, 298].
- Other less common but viable choices include: the **direct specification of dynamical systems**, where the genome represents geometric components such as attractors and repulsors [267]; the use of **cell sorting**, or the simulation of random cell motion among a collection of cells with various affinities for attraction, which can be used to generate a final phenotype [107].

A major concern for designers of artificial development (and nearly all complex systems) is how to find the micro-rules which will generate a desired macro-scale pattern. Indeed, this problem has seen little progress despite several decades of research, and in the case of certain generative machines such as cellular automata, it is even known to be impossible [133]. The primary way to solve this issue is using a machine learner as a search method. Evolutionary computation is the general choice for this machine learner, mostly due to the flexibility of genomic representations and objective functions, and the capacity to easily incorporate conditions and heuristics. In this case, the phenotype of the discovered design solution will be an unpredictable, emergent trait of bottom-up design choices, but one which meets the needs of the objective function. Various authors have explored several means of ameliorating this approach, in particular by controlling or predicting the evolutionary output [213, 214].

## *2.3 Why Does Artificial Development Work?*

The means by which development improves the evolvability of organisms is a critical question. In biology, the importance of developmental mechanisms in organismal organization has slowly been acknowledged. Several decades ago, Gould (controversially) characterized the role of development as that of a "constraint", or a "fruitful channelling [to] accelerate or enhance the work of natural selection" [99]. Later authors envisioned more active mechanisms, or "drives" [7, 152]. More recently, discussion has turned to "increased evolvability", partly in recognition that no simple geometric or phenotypic description can presently describe all useful phenotypic biases [115]. At the same time, mechanisms of development have gained in importance in theoretical biology, spawning the field of evo-devo [31] mentioned above, and convincing several researchers that the emergence of physical epigenetic cellular mechanisms capable of supporting robust multicellular forms was, in fact, the "hard" part of the evolution of today's diversity of life [212].

Inspired by this related biological work, practitioners of artificial development have hypothesized several mechanisms as an explanation for the success of artificial development, or as candidates for future experiments:

- **Regularities**: this term is used ambiguously in the literature. Here, we refer to the use of simple geometrically based patterns over space as a means of generating or biasing phenotypic patterns, for example relying on Wolpert's notion of gradient-based positional information [295]. This description includes many associated biological phenomena, such as various symmetries, repetition, and repetition with variations. Regularities in artificial development are well studied and present in many models; arguably the first AD model, Turing's models of chemical morphogenesis, relied implicitly on such mechanisms through chemical diffusion [281]. A recent and popular example is the Compositional Pattern Producing Network (CPPN), an attempt to reproduce the beneficial properties of development without explicit multicellular simulation [266] (see also Sect. 5.4 and Chap. 5).
- **Modularity**: this term implies genetic reuse. Structures with commonalities are routine in natural organisms, as in the repeated vertebrae of a snake, limbs of a centipede, or columns in a cortex [29]. As Lipson points out, modules need not even repeat in a particular organism or design, as perhaps they originate from a meta-processes, such as the wheel in a unicycle [174]. Despite this common conception, there is significant disagreement on how to define modularity in neural systems. In cognitive science, a module is a functional unit: a specialized and encapsulated unit of function, but not necessarily related to any particular low-level property of neural organization [89, 233]. In molecular biology, modules are measured as either information-theoretic clusters [121], or as some measure of the clustering of network nodes [147, 211, 289]. These sorts of modularity are implicated in the separation of functions within a structure, allowing for greater redundancy in functional parts, and for greater evolvability through the separation of important functions from other mutable elements [229]. Further research shows that evolution, natural and artificial, induces modularity in some form, under pressures of dynamic or compartmentalized environments [23, 24, 39, 121, 147], speciation [82], and selection for decreased wiring costs [39]. In some cases, these same measures of modularity are applied to neural networks [23, 39, 147]. Beyond modularity, hierarchy (i.e. the *recursive composition of a structure and/or function* [64, 124, 174]) is also frequently cited as a possibly relevant network property.
- **Phenotypic properties**: Perhaps the most literal interpretation of biological theory comes from Matos et al., who argue for the use of measures on phenotypic space. In this view, an AD system promotes a bias on the space of phenotypic structures that can be reached, which might or might not promote success in some particular domain. By enumerating several phenotypic properties (e.g. "the number of cells produced") they contrast several developmental techniques, showing the bias of AD systems relative to the design space [185]. While this approach is certainly capable of adapting to the problem at hand, it requires *a priori* knowledge of the interesting phenotypic properties—something not presently existing for large neural systems;

- **Adaptive feedback and learning**: Some authors posit adaptive feedback during development as a mechanism for improved evolvability. The use of an explicit developmental stage allows for the incorporation of explicit cues in the resulting phenotype, a form of structural plasticity which recalls natural growth. These cues include not only a sense of the environment, as was previously discussed, but also interim indications of the eventual success of the developing organism. This latter notion, that of a continuous measure of viability, can be explicitly included in AD system, and has been shown in simple problems to improve efficacy and efficiency [12, 157, 158, 190]. A specialized case of adaptive feedback is *learning*, by which is meant the reaction to stimuli by specialized plastic components devoted to the communication and processing of inter-cellular signals. This important mechanism is discussed in the next section.

## 3 Artificial Neurogenesis

By artificial neurogenesis, we mean *a developmentally inspired process that generates neural systems for use in a practical context*. These contexts include tasks such as supervised learning, computer vision, robotic control, and so on. The definition of developmentally inspired processes in this chapter is also kept broad on purpose: at this early stage, we do not want to exclude the possibility that aspects of our current understanding of development are spurious or replaceable.

An interesting early example of artificial neurogenesis is Gruau's *cellular encoding* [103]. Gruau works with directed graph structures: each neural network starts with one input and one output node, and a hidden "mother" cell connected between them. The representation, or "genome", is a tree encoding that lists the successive cell actions taken during development. The mother cell has a reading head pointed at the top of this tree, and executes any cellular command found there. In the case of a division, the cell is replaced with two connected children, each with reading heads pointed to the next node in the genome. Other cellular commands change registers inside cells, by adding bias or changing connections. A simple example is illustrated in Fig. 8.

Through this graph-based encoding, Gruau et al. designed and evolved networks solving several different problems. Variants of the algorithm used learning as a midstep in development and encouraged modularity in networks through the introduction of a form of genomic recursion [103, 104]. The developed networks showed strong phenotypic organization and modularity (see Fig. 9 for samples).

### *3.1 The Interplay Between Development and Learning*

A critical difference between artificial neurogenesis and AD is the emphasis on learning in the latter. Through the modelling of neural elements, a practitioner includes

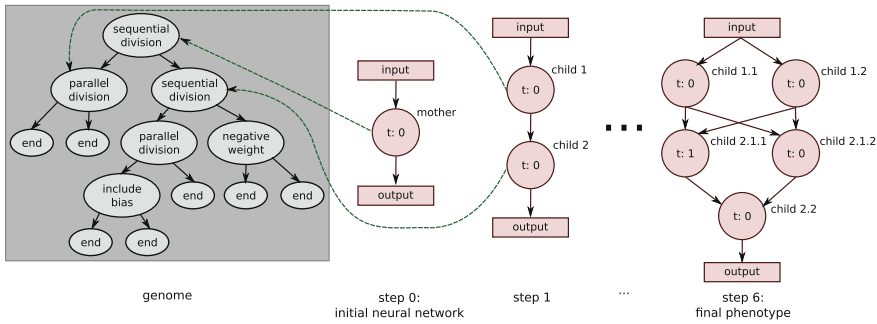


**Fig. 8** Simple example of a neural network generated via cellular encoding (adapted from [103]). On the *left*, an image of the genome of the network. On the *right*, snapshots of the growth of the neural network. The *green arrows* show the reading head of the active cells, that is, which part of the genome they will execute next. This particular network solves the XOR problem. Genomic recurrence (not shown) is possible through the addition of a recurrence node in the genomic tree

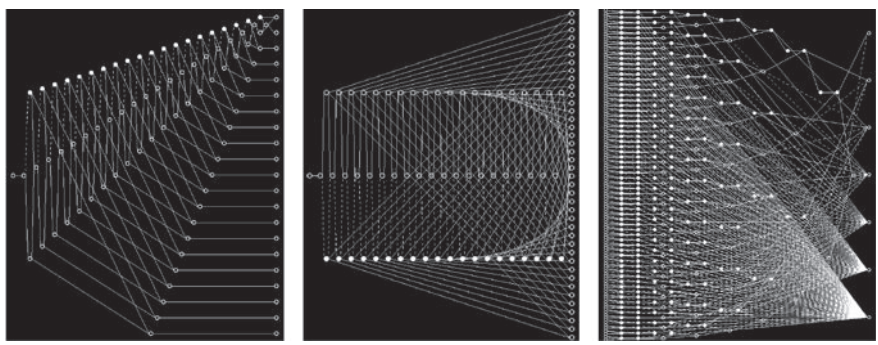

**Fig. 9** Sample neural networks generated via cellular encoding: *left* a network solving the 21-bit parity problem; *middle* a network solving the 40-bit symmetry problem; *right* a network implementing a 7-input, 128-output decoder (reproduced with permission from [103])

any number of plasticity mechanisms that can effectively incorporate environmental information.

One such hypothetical mechanism requiring the interplay between genetics and epigenetics is the *Baldwin effect* [9]. Briefly, it concerns a hypothesized process that occurs in the presence of both genetic and plastic changes and accelerates evolutionary progress. Initially, one imagines a collection of individuals distributed randomly over a fitness landscape. As expected, the learning mechanism will push some, or all, of these individuals toward local optima, leading to a population more optimally distributed for non-genetic reasons. However, such organisms are under "stress" since they must work to achieve and maintain their epigenetically induced location in the fitness landscape. If a population has converged toward a learned optimum, then in subsequent generations, evolution will operate to lower this stress, by finding genetic

means of reducing the amount of learning required. Thus, learning will identify an optimum, and evolution will gradually adapt the genetic basis of the organism to fit the discovered optimum. While this effect is purely theoretical in the natural world, it has long been known that it can be generated in simple artificial organisms [120]. Accommodating developmental processes in these artificial models is a challenge, but examples exist [72, 103]. Other theories of brain organization, such as displacement theory, have also been tentatively explored in artificial systems [70, 71].

## *3.2 Why Use Artificial Neurogenesis?*

There is danger in the assumption that all products of nature were directly selected for their contribution to fitness; this Panglossian worldview obscures the possibility that certain features of natural organisms are the result of non-adaptive forces, such as genetic drift, imperfect genetic selection, accidental survivability, side-effects of ontogeny or phylogeny, and others [100]. In this spirit, we note that while a computer simulation might show a model to be *sufficient* for the explanation of a phenomenon, it takes more work to show that it is indeed *necessary*. Given the staggering complexity of recent neural models, even a successful recreation of natural phenomena does not necessarily elucidate important principles of neural organization, especially if the reconstructed system is of size comparable to the underlying data source. A position of many practitioners working with bio-inspired neural models, as in artificial intelligence generally, is that an *alternative path to understanding neural organization is the bottom-up construction of intelligent systems*. The creation of artefacts capable of simple behaviours that we consider adaptive or intelligent gives us a second means of "understanding" intelligent systems, a second metric through which we can eliminate architectural overfitting from data-driven models, and identify redundant features of natural systems.

A second feature of many developmental neural networks is the reliance on local communication. Practitioners of AD will often purposefully avoid global information (e.g. in the form of coordinate spaces or centralized controllers) in order to generate systems capable of emergent global behaviour from purely local interactions, as is the case in nature. Regardless of historic motivations, this attitude brings potential benefits in engineered designs. First, it assumes that the absence of global control contributes to the scalability of developed networks (a special form of the robustness discussed in Sect. 2.1.1). Second, it guarantees that the resulting process can be implemented in a parallel or distributed architecture, ideally based on physically asynchronous components. Purely local controllers are key in several new engineering application domains, for instance: a uniform array of locally connected hardware components (such as neuromorphic engineering), a collection of modules with limited communication (such as a swarm of robots, or a collection of software modules over a network), or a group of real biological cells executing engineered DNA (such as synthetic biology).

### 3.3 Model Choices

A key feature in artificial neurogenesis is the level of simulation involved in the growth model. It can range from highly detailed, as is the case for models of cellular physics or metabolism, to highly abstract, when high-level descriptions of cellular groups are used as building blocks to generate form. While realism is the norm in computational neuroscience, simpler and faster models are typical in machine learning. An interesting and open question is whether or not this choice limits the capacity of machine learning models to solve certain problems. For artificial neurogenesis, relevant design decisions include: spiking versus non-spiking neurons, recurrent versus feed-forward networks, the level of detail in neural models (e.g. simple transmission of a value versus detailed models of dendrites and axons), and the sensitivity of neural firing to connection type and location.

Perhaps the most abstract models come from the field of *neuroevolution*, which relies on static feed-forward topologies and nonspiking neurons. For instance, Stanley's HyperNEAT model [49] generates a pattern of connections from another lattice of feed-forward connections based on a composition of geometric regularities. This model is a highly simplified view of neural development and organization, but can be easily evolved (see Chap. 5, [48]). A far more detailed model by Khan et al. [151] provides in each neuron several controllers that govern neural growth, the synaptogenesis of dendrites and axons, connection strength, and other factors. Yet, even these models are highly abstract compared to other works from computational neuroscience, such as the modelling language of Zubler et al. [311]. The trade-offs associated with this level of detailed modelling are discussed in depth by Miller (Chap. 8, [198]).

Assuming that connectivity between neurons depends on their geometric location, a second key question concerns the level of stochasticity in the placement of those elements. Many models from computational neuroscience assume that neural positions are at least partially random, and construct models that simply overlay preformed neurons according to some probability law. For instance, Cuntz et al. posit that synapses follow one of several empirically calculated distributions, and construct neural models based on samples from those distributions [41]. Similarly, the Blue Brain project assumes that neurons are randomly scattered: this model does, in fact, generate statistical phenomena which resemble actual brain connectivity patterns [116].

A final key decision for artificial neurogenesis is the level of detail in the simulation of neural plasticity. These include questions such as:

- Is plasticity modelled at all? In many applications of neuroevolution (Sect. 4.3), it is not: network parameters are determined purely via an evolutionary process.
- Does plasticity consist solely of the modification of connection weights or firing rates? This is the case in most classical neural networks, where a simple, almost arbitrary network topology is used, such as a multilayer perceptron. In other cases, connection-weight learning is applied to biologically motivated but static network topologies (Sects. 4.1 and 4.2, Chap.7 [13]).

- How many forms of plasticity are modelled? Recent examples in reservoir computing show the value of including several different forms (Sect. 6.1).
- Does the topology of the network change in response to stimuli? Is this change based on a constructive or destructive trigger (Sect. 6.2)? Is the change based on model cell-inspired synaptogenesis (Sect. 5)?

The plethora of forms of plasticity in the brain suggests different functional roles in cognition. For instance, artificial neural networks are prone to a phenomenon known as "catastrophic forgetting", that is, a tendency to rapidly forget all previously learned knowledge when presented with new data sources for training. Clearly, such forgetfulness will negatively impact our capacity to create multi-purpose machines [90]. Miller and Khan argue, however, that re-introducing metaphors for developmental mechanisms, such as dendritic growth, overcomes this limitation [201].

## *3.4 Issues Surrounding Developmental Neural Network Design*

The use of a developmentally inspired representation or growth routine in neural network design implies a scale of network rarely seen in other design choices. Indeed, development is associated with the generation of large structures and is not expected to be useful below a minimal number of parts. This leads to several related issues for practitioners:

- Large networks are difficult to train via conventional means. This is mainly due to computational complexity, as training procedures such as backpropagation grow with the number of connections in a network.
- A more specific issue of size, *depth*, refers to the number of steps between the input and output of the network. It is known that there are exponentially more local optima in "deep" networks than "shallow" ones, and this has important consequences for the success of a gradient-descent technique in a supervised learning task. Despite these difficulties, depth is found to be useful because certain problems can be represented in exponentially smaller formats in deep networks [16].

These issues can be ameliorated via several new and highly promising neural techniques. On such technique is *reservoir computing*, where only a small subset of a large network is trained (Sect. 4.2). A second such technique is *deep learning*, where a deep network is preconditioned to suit the data source at hand (Sect. 4.1).

In much of statistical learning, there is a drive toward finding the most parsimonious representation possible for a solution. This is usually the case in constructive and pruning networks (Sect. 6.2), in which a smaller network is an explicit metric of success. Obviously, simpler solutions are more efficient computationally and can be more easily understood. However, it is further claimed that parsimonious solutions will also perform better on previously unseen data, essentially based on the *bias/variance trade-off* argument by Geman et al. [92]. They show that for a simple, fully connected network topology, the number of hidden nodes controls the level

of bias and variance in a trained classifier. Too many nodes lead to a network with excessive variance and overfitting of the training data. They conclude that the hard part of a machine learning problem is finding a representational structure that can support a useful "bias" toward the problem at hand. It means that a heuristic architectural search must precede the exploration and optimization of network parameters. Perhaps inspired by this and similar studies on limited representations, and the hope that smaller representations will have less tendencies to overfit, parsimony is often an explicit goal in optimization frameworks. Yet, we take here a different view: for us, *certain forms of redundancy in the network might in fact be one of the architectural biases that support intelligence*. In AD, redundancy is often celebrated for increasing resilience to damage, allowing graceful degradation, and creating neutral landscapes, or genetic landscapes that encourage evolvability [239, 248, 305].

## 4 Bio-Inspired Representations

Many neural models do not explicitly simulate any developmental process, yet they are substantially informed by biology through the observation the network structure of natural neural systems (or systems from computational neuroscience), and the inclusion of an explicit "bias" containing similar properties. Several of these approaches have proven tremendously successful in recent years, contributing to the so-called "second neural renaissance" that has reinvigorated research in artificial neural networks. We summarize below some of these bio-inspired representations.

### *4.1 Deep Learning*

With the advent of deep learning, neural networks have made headlines again both in the machine learning community and publicly, to the point that "deep networks" could be seen on the cover of the New York Times. While deep learning is primarily applied to image and speech recognition [15, 46, 171], it is also mature enough today to work out of the box in a wide variety of problems, sometimes achieving state-of-the-art performance. For example, the prediction of molecular activity in the Kaggle challenge on Merck datasets (won by the Machine Learning group of the University of Toronto), or collaborative filtering and preference ranking in the Netflix movie database [246] both used deep learning.

These impressive results can be explained by the fact that deep learning very efficiently learns simple features from the data and combines them to build high-level detectors, a crucial part of the learning task. The features are learned in an unsupervised way and the learning methods are scalable: they yield the best results on the ImageNet problem [52, 166, 170], a dataset comprising 1,000 classes of common object images, after a training process that ran on a cluster of tens of thousands of CPUs and several millions of examples. Even through purely unsupervised training

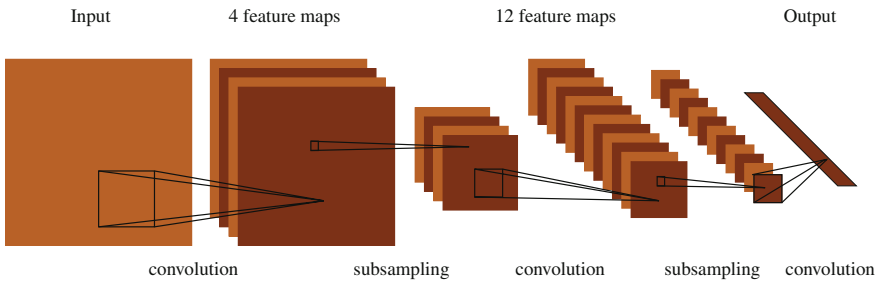


**Fig. 10** Architecture of a convolution neural network, as proposed by LeCun in [171]. The convolutional layers alternate with subsampling (or pooling) layers

on YouTube images, the features learned are specialized enough to serve as face detectors or cat detectors. A straightforward supervised tuning of these unsupervised features often leans to highly effective classifiers, typically outperforming all other techniques.

Deep networks are similar to the classical multilayer perceptrons (MLP). MLPs are organized into "hidden layers", which are rows of neurons receiving and processing signals in parallel. These hidden layers are the actual locus of the computation, while the input and output layers provide the interface with the external world. Before deep learning, most multilayered neural nets contained only one hidden layer, with the notable exception of LeCun's convolutional network [171] (see below). One reason comes from the theoretical work of Håstad [112], who showed that all boolean circuits with $\ell + 1$ layers could be simulated with $\ell$ layers, at the cost of an exponentially larger number of units in each layer. Therefore, to make the model selection phase easier, for example chosing the number of units per layer, a common practice was to consider a single hidden layer. Another reason is that networks with more than one or two hidden layers were notoriously difficult to train [274], and the very small number of studies found in the literature that involve such networks is a good indicator of this problem.

Pioneering work on deep learning was conducted by LeCun [171], who proposed a family of perceptrons with many layers called convolutional networks (Fig. 10). These neural networks combine two important ideas for solving difficult tasks: shift-invariance, and reduction of dimensionality of the data. A convolution layer implements a filtering of its input through a kernel function common to all neurons of the layer. This approach is also called weight sharing, as all neurons of a given layer always have the same weight pattern. Convolution layers alternate with "pooling layers", which implement a subsampling process. The activation level of one neuron in a pooling layer is simply the average of the activity of all neurons from the previous convolution layer. In the first layer, the network implements a filter bank whose output is subsampled then convolved by the filter implemented in the next layer.

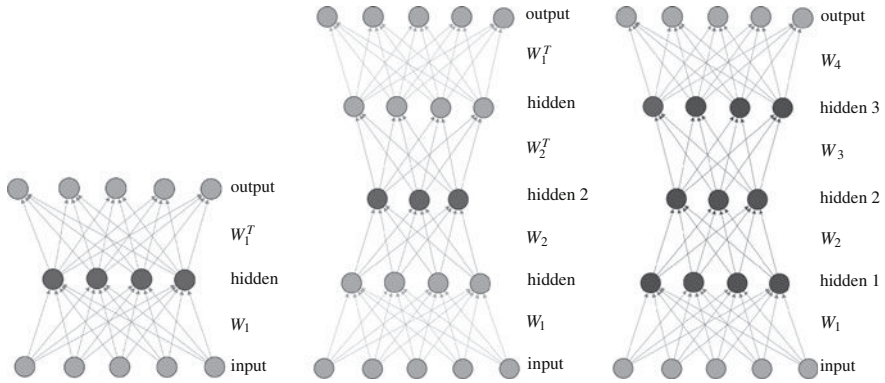


**Fig. 11** Layer-wise unsupervised training in a deep architecture: *left* training of the first hidden layer, shown in *black*; *center* training of the second hidden layer, shown in *black*. Hidden layers and associated weights that are not subject to learning are shown in *grey*

Therefore, each pair of layers extracts a set of features from the input, which in turn feed into the next pair of layers, eventually building a whole hierarchy of features. Interesting variants of convolutional networks include L2-pooling, in which the $L_2$ norm of a neuron's activation in the previous layer is used instead of the maximum or the average [141], and contrast normalization, where the activities of the pooling neurons are normalized.

Hierarchical combination of features is the key ingredient of deep networks. In convolutional networks, the weight sharing technique allows learning a specific filter for each convolution map, which drastically reduces the number of variables required, and also explains why convolutional networks converge by simple stochastic gradient descent. On the other hand, weight sharing also limits the expressivity of the network, as each filter must be associated to a feature map and too many feature maps could negatively affect the convergence of the learning algorithm.

To overcome this trade-off, the method proposed by deep learning is to build the network step by step and ensure the learning of a feature hierarchy while maintaining good expressivity [81]. This is implemented via layer-wise unsupervised training, followed by a fine tuning phase that uses a supervised learning algorithm, such as gradient descent (Fig. 11). The idea of relying on unsupervised learning to train a network for a supervised task has been advocated by Raina et al. [235] in their work about self-taught learning. It is known that adding unlabelled examples to the training patterns improves the accuracy of the classifiers, an approach called "semi-supervised" learning [217]. In self-taught learning, however, any example and any signal can be used to improve the classifier's accuracy.

The underlying hypothesis is that recurring patterns in the input signal can be learned from any of the signal classes, and these typical recurrent patterns are helpful to discriminate between different signal classes. In other words, when the signal space

is large, it is possible to learn feature detectors that lie in the region containing most of the signal's energy, and then, classifiers can focus on this relevant signal space.

The layer-wise unsupervised objective of a deep network is to minimize the reconstruction error between the signal given on the input layer of the network and the signal reconstructed on the output layer. In the autoencoder framework, this first learning step, also called generative pretraining, focuses on a pair of parameters, the weight matrix $W$ and the bias $b$ of an encoder-decoder network. The encoder layer is a mapping $f$ from the input signal $x$ to an internal representation $y$:

$$y = f(x) = s(Wx + b) \tag{1}$$

where $b$ is a bias vector and $s$ is a non-linear function, usually a sigmoidal function. The decoder is a mapping from the internal state to a reconstructed signal $z$:

$$z = g(y) = s(W^T x + b^T) \tag{2}$$

In the left part of Fig. 11, the input vector $x$ activates the neurons of the input layer, the internal state $y$ of the hidden layer is expressed by Eq. (2) and the output layer is $z = g(f(x))$. The reconstructed error to minimize is then:

$$L(x, z) \propto -\log p(x|z) \tag{3}$$

A deep network can be built by stacking networks on top of each other. The most common are the autoencoders, also called auto-associators, which are often constrained to either ensure sparsity (sparse autoencoders) [15, 169, 236], or enforce generalization by purposefully corrupting the input signals, as with denoising autoencoders [81, 285]. Another widely investigated type of network is the restricted Boltzmann machine (RBM) [119], which is based on latent variables and a probabilistic formulation. A bound on accuracy ensures that stacking RBMs could only improve the accuracy of the whole architecture. Recent developments have shown that it is possible to use many different classifiers as the building blocks of a deep architecture. Nonetheless, the neural networks and their training have been sufficiently investigated to be integrated into a toolbox and applied without prior knowledge to nearly any pattern recognition problem.

The incremental method used in deep learning can be construed as a type of simplified evolutionary process, in which a first layer is set up to process certain inputs until it is sufficiently robust, then a second layer uses as input the output of the first layer and re-processes it until convergence, and so on. In a sense, this mimics an evolutionary process based on the "modularity of the mind" hypothesis [89], which claims that cognitive functions are constructed incrementally using the output of previous modules leading to a complex system. Another evolutionary perspective on deep learning, in relation with cultural development, is proposed by Bengio [14].

**Chapter 3: Evolving culture versus local minima.**
In Chap. 3, Bengio [14] provides a global view of the main hypotheses behind the training of deep architectures. It describes both the difficulties and the benefits of deep learning, in particular the ability to capture higher-level and more abstract relations. Bengio relates this challenge to human learning, and proposes connections to culture and language. In his theory, language conveys higher-order representations from a "teacher" to a "learner" architecture, and offers the opportunity to improve learning by carefully selecting the sequence of training examples—an approach known as Curriculum Learning. Bengio's theory is divided into several distinct hypotheses, each with proposed means of empirical evaluation, suggesting avenues for future research. He further postulates cultural consequences for his theory, predicting, for instance, an increase in collective intelligence linked to better methods of memetic transmission, such as the Internet.

From a computational viewpoint, signals acquired from natural observations often reside on a low-dimension manifold embedded in a higher-dimensional space. Deep learning aims at learning local features that characterize the neighbourhood of observed manifold elements. A connection could be made with sparse coding and dictionary learning algorithms, as described in [222], since all these data-driven approaches construct over-complete bases that capture most of the signal's energy. This line of research is elaborated and developed in Chap. 4 by Rebecchi, Paugam-Moisy and Sebag [236].

**Chapter 4: Learning sparse features with an auto-associator.**
In Chap. 4, Rebecchi, Paugam-Moisy and Sebag [236] review the recent advances in sparse representations, that is, mappings of the input space to a high-dimensional feature space, known to be robust to noise and facilitate discriminant learning. After describing a dictionary-based method to build such representations, the authors propose an approach to regularize auto-associator networks, a common building block in deep architectures, by constraining the learned representations to be sparse. Their model offers a good alternative to denoising auto-associator networks, which can efficiently reinforce learning stability when the source of noise is identified.

To deal with multivariate signals and particularly complicated time-series, several deep learning systems have been proposed. A common choice is to replicate and connect deep networks to capture temporal aspect of signals, using learning rules such as backpropagation through time. However, since these networks are recurrent, the usual gradient descent search does not converge. Consequently, "vanishing" or "exploding" gradient descents have also been the subject of an intense research

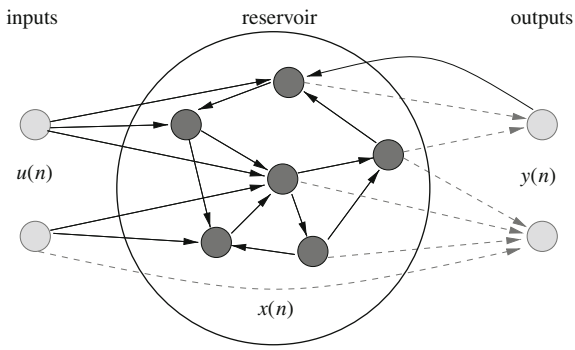


**Fig. 12** A time-series $u(n)$ is assigned to *input* neurons. These *input* neurons are connected to the "reservoir", a recurrent neural network $x(n)$. Only a subset of $x(n)$ is connected to the output neurons $y(n)$

effort and have led to the development of reservoir computing approaches, which are detailed in the next section.

## *4.2 Reservoir Computing*

Reservoir computing is an approach and family of models that rely on a recurrent neural network, called a reservoir, to generate a high-dimensional and dynamical representation of a given input, often a time series. Typically, the connections and weights in the network are randomly distributed and can produce a large number of nonlinear patterns from an input stream. Rather than modifying the reservoir with a supervised learning algorithm, however, the dynamical state of the reservoir is "read out" by a simple classifier, for example a linear regression or support vector machine, connected to a fraction of its neurons. This idea has been instantiated in different neural models, the two best known being Jaeger's Echo State Network (ESN) [137] and Maass' Liquid State Machine (LSM) [180, 210] (Fig. 12).

The ESN formulation uses a common sum of the weight / nonlinearity neurons as a neural model for the reservoir, such as McCulloch and Pitts neurons [189] or sigmoidal units. A good reservoir should produce a rich dynamics to facilitate the separability of its activity traces by the readout classifier, thus it is usually large, sparsely and randomly connected. It should also possess the *echo state property*, meaning that the effect of a previous state of the network should vanish asymptotically—in other words, the network should "forget" its previous state in a finite amount of time. To ensure this property, a common practice is to verify that the spectral radius $|\lambda_{\max}|$ of the weight matrix of the reservoir $W$ is less than 1, and close to 1 for tasks requiring long memories. Other strategies have also been explored, taking into account more specialized neural types [303].

In other setups, the reservoir relies on more realistic neuronal models, such as integrate-and-fire or spiking neurons. While such temporal models endow the network with a richer dynamics, they also come at the expense of computational

efficiency, which can be noticeably reduced even in the case of simple spiking neurons [27]. Nonetheless, this type of reservoir form the basis of LSMs [210] and "cortical microcircuits" [180], which have been employed less frequently than ESNs. In any case, both types have been applied to a great variety of tasks and are theoretically linked [138].

One of the major difficulties of the LSM paradigm is the choice of the *readout classifier*. A simple linear regression achieves correct results and has been used to demonstrate theoretical characterizations [180], yet it ignores the fact that spiking neurons convey information through precise spike timings. Several propositions have been made to exploit this temporal information: encoding patterns with transient synchrony, as shown by Hopfield and Brody [123], or applying a margin classifier based on connection delays [226].

Whether LSMs or ESNs, another key element is the *topology* of the reservoir. The spectral radius $|\lambda_{\max}|$ of the weight matrix $W$ plays a crucial role in determining the dynamics that will take place in the recurrent network. Other factors, such as small-world degree, scale-free regimes, and bio-inspired axonal growth patterns, have also been shown to positively influence the capabilities of the reservoir [242]. On the other hand, a recent theoretical analysis by Zhang et al. argues that all random reservoir topologies asymptotically converge to the same distribution of eigenvalues, implying that the topology is relatively indifferent after all [307]. Finer investigations of the dynamics are also possible [226] but they have not yet been applied in this context.

Beyond fixed topologies, a topic of great relevance to this chapter concerns endowing the reservoir with *plasticity*. Applying an unsupervised learning procedure to the weights allows the reservoir to adapt to very constrained topologies, although a theoretical analysis in this case becomes problematic [250]. The use of plasticity in reservoir computing will be discussed further in Sect. 6. Another widely investigated aspect is the influence of an *external loop*, by which the readout classification results are reinjected in the reservoir. This feedback adds another level of cognition to the network, as the system can now utilize its own capacity for prediction in input. An in-depth review of reservoir computing challenges and common practice can be found in a special issue of Neural Network [139], and a comprehensive explanation of the ongoing approaches is proposed in [179].

## *4.3 Neuroevolution*

In evolutionary computation, there has been an long-standing interest in artificial neural networks for classification and regression, as well as control problems. The term "neuroevolution" is now well established and covers a large range of approaches (evolving weights, evolving topologies, learning rules, developmental processes) and applications. In this framework, the design of a particular neural network for solving a task is driven by its performance with respect to the defined task. This performance is itself described in terms of fitness value(s), and the evolutionary algorithm targets incremental improvements of fitness evaluations. To this aim, it

produces new candidate solutions (i.e. particular configurations of neural networks) from previously tried ones through the action of various mutation and recombination operators [79].

For example, neural networks have long been the method of choice of evolutionary robotics when performing "policy search" in reinforcement learning problems [219, 271, 273]. As a formalism for controller representation, they exhibit interesting features such as robustness (with respect to noisy sensory inputs), evolvability (in the sense that small weight changes give rise to small behavioral changes), and ease of implementation (since update time varies only linearly with the number of links). This is also true in situations with limited hardware specifications, such as onboard robotic systems.

Three important decisions may impact the choice of a learning method to train the network: (a) the definition of the neural network to be considered (with/without recurrent connections), (b) the choice of variables to learn (the weights of a fixed topology, and/or the topology), and (c) how these variables will be learnt (how to encode such a network, how to navigate through the search space). While there exist several methods in the literature for evolving weights only, such as classic multi-layered perceptrons or echo state networks [110], things become more challenging when evolving entire topologies. On the one hand, the choice of a particular search space relies for a great part on the programmer's expertise, and a poor guess may hinder the whole process. On the other hand, learning both the weights and the topology opens up a much larger search space and may well lead to performance normally unreachable through pure synaptic modification. Due to the versatility and robustness of evolutionary algorithms, they are considered promising candidates in the exploration of configuration spaces.

Evolutionary algorithms (EAs) quickly appeared as a relevant approach toward NN learning by the end of the 1990s (see [302] for a detailed survey). Although EAs can be useful for the optimization of the weights of a feedforward NN, they have instead been mainly used for their flexibility in handling complex search spaces. Many algorithms modifying the structure of neural networks through dedicated variation operators have been proposed.

Notable works and models in this field include: GNARL [6], which uses a direct encoding of the neural network to build a robot controller; EANT [148], which evolves the structure and weights via distinct processes; SANE (Symbiotic Adaptive Neuro-Evolution) [204] and ESP (Enforced Sub-Population) [96, 97], which evolve a population of neurons (rather than a network) and combine these neurons to form effective neural networks; and GASNET [132], which combines the optimization of the position of neurons in an Euclidean space through diffusion of chemicals. More recently, NEAT (Neuro Evolution of Augmenting Topologies) [264] has set new standards for neuroevolution algorithms in pure performance and speed of convergence based on classical benchmarks from evolutionary robotics.

It has been known for a long time [194] that the choice of a representation, i.e. search space, is crucial for the success of any evolutionary algorithm. This led to the exploration of genotype-to-phenotype maps using a more compact representation, which should theoretically enable the evolution of more complex neural networks.

One approach is the inclusion of an induced representational bias. In these cases, forms of structural organization known to be employed by natural neural systems are added to the otherwise directly encoded network. An illuminating study by Seys and Beer considers the value of several forms of induced symmetry on the generation of NNs [256]. Evolved genomes are "unpacked" by creating symmetric copies of the evolved network substructure, and evaluated via the whole unpacked network. The authors contrast their results against nonsymmetric networks, showing that the inclusion of symmetry makes NNs more evolvable, even compared to nonsymmetric networks of smaller size. A more practical example inspired by computational neuroscience models comes from Doncieux et al. [59]. Finally, recent works have explored the benefits of particular topological properties, such as regularity and modularity, whether a priori designed or evolved [33, 37, 284]. Another approach to the genotype-to-phenotype map strategy consists of including a simulation of development, a process that serves to construct the phenotype in a time-based fashion. The next section covers these strategies in greater detail.

## 5 Developmental Systems

This section gives an overview of neural *developmental systems*, which are about the abstraction of a developmental process to obtain artificial neural networks from simpler representations. Since, in the vast majority of cases, adequate representations (genomes) are optimized via evolutionary computation, we will use terminology from that field. Due to their metaphorical inspiration from developmental biology, developmental systems have received several names, including computational embryogeny [17], artificial ontogeny [25] and artificial embryogeny [265]. While all these terms emphasize the biological metaphor, we think that a broader phrasing of "evolution of developmental neural networks", or *evo-devo-NN* for short, would be more appropriate for this section.

The idea of combining evolution and development for designing artificial neural networks was first put to the test in 1990. Kitano [153] criticized direct encoding methods and proposed exploring indirect encodings as a promising solution to address the challenge of scalability. In this setup, the evolved genotypic description of a solution undergoes a reformulation process (or *mapping*) in order to obtain a usable phenotype. As noted in later works, this would enable it to evolve compact representations and possibly exploit properties such as modularity and hierarchy.

The debate about the relevance of evolving developmental neural networks has been, and still is, very much alive. In the first decade after Kitano's original claim regarding scalability, his results were first confirmed [75] (in a different context) then challenged [258] (in the original context). We now review the various approaches and works conducted in this area.

## 5.1 Grammar-Based Encoding

In his seminal work, Kitano [153] described the first approach with indirect encoding for generating artificial neural networks. Kitano used a genetic algorithm to evolve L-System grammar rules, a work which was later extended with lifelong neurogenesis [154]. His original contribution also fostered the emergence of a whole new field, which explored various approaches to developmental neural networks and addressed various challenges, some of them still open. Since then, evolutionary L-Systems have been further applied to the generation of neural networks [22, 225] or the co-evolution of artificial creatures [126]. While similar methods have been proposed to evolve morphologies, Hornby's GenRe system [126] relied on L-Systems for generating both body *and* brain (i.e. neural network) of virtual and real robots.

As previously discussed, in 1992 Gruau designed an original approach to the evolution of graph-rewriting rules called Cellular Encoding [101, 102]. His model was based on genetic programming to evolve a list of instructions that an original cell could follow to determine its fate. This cell would undergo several transformations (such as cell division) until a graph was built. A major contribution of this work was to provide the first-ever neural controller of hexapodal robot gait [103]. It was further extended [155, 178] and reused [25, 164] by several authors.

Other studies have explored the evolution of rewriting rules to generate neural networks. In the early 1990s, Nolfi and Parisi evolved direct encodings of neuron locations on a 2D substrate, then applied a heuristic for the simulation of axon growth (using previously evolved parameters) to obtain full-grown networks that were executing a robot navigation task. This work was later extended with cell division and migration [30], and lifetime adaptation through environment-triggered axon growth [218]. In 1994, Sims' "virtual creatures" also relied on evolved graph-rewriting rules both in the neural networks *and* in the morphologies [261]. Most recently, Mouret and Doncieux proposed Modular Encoding for Neural Networks based on Attribute Grammars (MENNAG), a general approach to evo-devo-NN based on the definition of grammar-based constraints [207], and the EvoNeuro method [205], which takes inspiration from computational neuroscience in order to generate large-scale and highly regular neural networks. Other applications exist as well [3].

## 5.2 Genetic Regulatory Networks

A major topic of interest in theoretical biology todayf is the modelling of *gene regulatory networks* (GRNs), which represent the interactions among genes and transcription products (mainly DNA-binding proteins) governing cell behaviour and maintenance. Generally, theoretical models are chosen based on their ability to

replicate specific patterns of expression found in natural systems [4, 150], or based on approximations of the molecular mechanisms of gene regulation [10]. In all cases, GRN simulations comprise a set of differential equations describing the dynamics of a various product concentrations. These models have been explored for their computational properties [144, 176, 215].

Offering a different approach to developmental systems, GRNs became a strong source of inspiration for researchers in computer science for obvious reasons. Starting from a relatively compact description such as a string of symbols, GRNs made it possible to build an entire network topology and function by defining interaction patterns among parts of the original representation. The possible benefits for control systems were first explored in 1994 by Dellaert and Beer [55] and quickly used to generate neural networks [54]. Their work combined a boolean GRN and a cell division process, alternating regulation and division over $N$ iterations, in order to iteratively grow a full neural network. Although evolution was only discussed, some of the resulting networks were tested on a simplified robot navigation task as a proof of concept.

This first attempt was soon followed by others that had the same dual objective of taking inspiration from biology to achieve compact representations and, at the same time, addressing evolutionary robotics challenges. Jakobi [140] described a method to evolve a bit-string, acting as a GRN to grow a neural network for robot control. Eggenberger proposed a similar approach, stressing scalability as the main motivation, and applied it to robot morphogenesis [76] and pattern recognition in neural networks [78].

An interesting alternative came from Reisinger and Miikkulainen [238], who used an evolved GRN structure directly as a neural network architecture. Applying their system to game playing, the authors contrasted their GRN-based NN against several non-developmental alternatives, and found favourable results. Their analysis mentions several reasons for this success: their representation was significantly more compact, more evolvable under a simple mutation operator, and pushed phenotypes toward larger, more recurrent network motifs, typical of networks in nature.[3]

A more recent instance of GRN-inspired neural model is the AGE (Analog Genetic Encoding) model by Mattiussi and Floreano [73, 186], in which a string of symbols (rather than bits) represents a genotypic description, while the "coding" parts of the genome (i.e. syntactically correct with respect to the gene definitions) build a neural network. As with other GRN abstractions, the AGE process is a one-step transformation from the representation to the network, i.e. self-regulation is abstracted as a one-pass process. Wróbel et al. later proposed a system called GReaNs (Genetic Regulatory evolving artificial Networks), which shares many similarities with AGE. Among several applications, GReaNs has been used to evolve spiking neural networks [296]. In this scope, each gene stands for a node, and the connection between

[3] The authors point out a similarity between their developed NNs and natural networks, specifically the existence of higher-order network triads. However, Milo et al. [202] attribute the existence of such triads in natural networks to the minimization of information processing time, a factor which was not relevant to the NNs. Hence, we consider this similarity unexplained.

nodes is determined by their relative euclidian distance to one another—as is the case with AGE. The sign of this connection, however, is evolved seperately.

**Chapter 6: Using the Genetic Regulatory evolving artificial Networks (GReaNs) platform for signal processing, animat control, and artificial multicellular development.**

In Chap. 6, Wróbel and Joachimczak present their bio-inspired model of pattern formation and morphogenesis, GReaNs, and show that it can support an evo-devo approach to complex neural networks. The topology of a GReaN is encoded in a linear genome composed of genetic modules, including regulatory factors, regions of promoting or repressing elements, and inputs and outputs. The resulting genetic network is evolved toward the control of the behaviour of a cell, coupling the chemical simulation with mechanical outputs. The authors review the results of previous experiments in which GReaNs have been used to design single-celled animats, 2D soft-bodied animats, and 3D morphologies, as well as more recent work where spiking neuron models emerge from the GRNs. They conclude by laying out their vision for the evolution of plausible neural control mechanisms from genetic origins [297].

## *5.3 Cellular Automata Models*

Also inspired from biology, several authors have explored models of multicellularity. Defined as a particular kind of cellular automaton (CA), each cell is capable of processing information, either by triggering further growth of the network or by relaying information as a neuronal cell. The seminal work from De Garis followed this metaphor to design the CAM-brain (Cellular Automata Machine), a two-dimensional CA in which a source cell could develop into a full organism capable of transmitting and manipulating information like a regular neural network [50]. This kind of approach raises the question of the halting problem, i.e. when and how development should stop [57]. Astor and Adami applied a similar approach based on a hexagonal grid, which addressed the halting problem by setting boundary cells to limit the total number of possible neurons. Adding a self-limiting mechanism allowed development to terminate before the environment was saturated with cells.

Early CA-based works were mostly limited to proof-of-concept experiments where evolution was merely discussed but not exploited. By contrast, Federici et al. designed a continuous CA implementation, which they termed *cell chemistry*, to generate spiking neural networks for solving a robotic navigation task in a discrete environment [84, 85]. In a later work, this approach was shown to outperform a direct encoding approach on a pattern recognition task [244].

### *5.4 HyperNEAT*

In 2007, Stanley et al. presented the first version of HyperNEAT (Hybercube-based NeuroEvolution of Augmenting Topologies [49, 266]. Since then, it has become very popular and has been applied in many domains, including the evolution of neural networks. One of the key principles behind HyperNEAT is a high level of abstraction that emphasizes the expressivity of the genotype-phenotype mapping in terms of composed transformation functions, instead of a temporal development process.

**Chapter 5: HyperNEAT: the first five years.**
In Chap. 5, D'Ambrosio, Gauci, and Stanley summarize recent work on generating patterns with properties such as regularity, symmetry, and repetition with variations. This chapter successively considers spatial pattern generation using CPPNs (compositional pattern-producing networks), NEAT (NeuroEvolution of Augmenting Topologies), and neural connectivity pattern generation using the many flavours of HyperNEAT (Hybercube-based NEAT). The basic idea behind this work is to define the search space as a set of compositions of simple functions, each function with particular behaviours (e.g. favouring symmetries, repetitions, etc.). To some extent, HyperNEAT is an abstraction of the developmental process, mapping a compact representation to a possibly large phenotype, but removing the temporal aspects of such a process. The benefits of the HyperNEAT approach are presented and discussed: the compact encoding of large networks which posses relevant structural properties, the ability to generate solutions in various sizes and resolutions, and the exploitation of the geometric properties of the problem at hand. Finally, a short review of existing applications across several fields is given, from image generation to robotic control, from visual discrimination to playing chess and Go [48].

HyperNEAT has also inspired other works in various ways. The HybridID ("Hybridization of Indirect and Direct Encodings") algorithm [38, 40] tries to integrate the best of indirect encodings and direct encodings by successively applying HyperNEAT and FT-NEAT to refine the last steps of evolution. The DSE ("Developmental Symbolic Encoding") model [270] takes inspiration both from HyperNEAT and Cellular Encoding, retaining interesting properties such as the ability to create neural networks with regularity, modularity and scalability. DSE also provides an interesting complement to existing approaches as it focuses on specific problems for which scale-free network topologies are relevant. Alternatively, the NEON ("NeuroEvolution with ONtogeny") algorithm [134] extends the traditional NEAT algorithm with a developmental process.

### 5.5 *Beyond Artificial Neural Networks*

From the start, there have been strong interactions among subfields of artificial development and evolution: from evolvable hardware [175, 199, 200] to simulated dynamics of genetic regulatory networks [10, 64, 237]; from artificial models of morphogenesis [57, 63, 158, 161] to agent control [191, 192]. In robotics (virtual or real), the integration of development with morphofunctional machines shows great promise toward faster and more innovative methodologies of automated design. Under the name of *brain-body co-evolution*, an emerging trend of evolutionary computation argues that structure and function should not be predefined and optimized separately, but simultaneously as a whole, and based on the same genome. The work of Sims offered the first results with simulated robots [261], and was soon followed by researchers exploring various grammar-based encoding for a similar purpose, such as the works lead by Eggenberger [77], Hornby [126] and Bongard [23].

Recent research in this domain has also seen a division between works targeting engineering and more fundamental research. On the one hand, developmental systems for morphogenesis is illustrated by physical systems from Hornby [125], Hiller [117] and Rieffel [240], where more recent works benefit from the advent of versatile 3D printing machines. On the other hand, several authors have either explored virtual creatures [35, 66, 142, 143, 156, 193, 249], or considered a less robot-oriented interpretation of simulated morphofunctional machines [43, 62, 63, 247]. Doursat et al. [67, 68] propose a new approach encompassing these trends: "Morphogenetic Engineering" aims to reconcile engineering with decentralized complex systems. It explores new methodologies to model and create precise architectures that self-organize from a swarm of heterogeneous agents, in particular by development. It can also describe brain representations based on dynamic "neural shapes" in phase space, formed by myriads of correlated spikes [60].

From artificial neural networks to robotics, this shared interest in the developmental paradigm can be explained by the need for features (such as modularity, regularity, or hierarchy) that are considered relevant for functional or morphological reasons. Moreover, scalability stands as a critical issue in all these domains, and the combination of a compact genotype with a dedicated developmental process remains a promising track to achieve large phenotypes, as long as evolvability as a property is successfully retained.

## 6 Epigenetic Simulation

In this section, we consider algorithms that rely primarily on the simulation of *epigenetic mechanisms*, in the sense that they build neural networks from transient information provided by stimuli. From a certain perspective, this is already the norm

in artificial neural nets, where "classic" techniques involve simple and often fixed network topologies trained via stimulus-based methods such as backpropagation. Here, by contrast, we consider cases in which the structural design of the network is strongly influenced by the environment, or where a more biologically motivated synaptic plasticity mechanism has a significant effect on the topology.

Perhaps the best argumentation that development and epigenetics are both necessary in artificial networks comes from a study by Valsalam et al. [283]. In this work, the authors were concerned with exploring the role of prenatal and postnatal learning on the generation of a network. Under this viewpoint, development is modelled by non-environmentally induced stimuli, that is, patterns produced genetically rather than coming from the task at hand. Valsalam et al. explored three groups of models, all applied to hand-written character recognition and relying on a simple static network *(terminology ours)*:

- **learn**: in the first group, networks were trained by competitive Hebbian learning using input samples
- **evo**: in the second group, networks evolved through a simple neuro-evolutionary technique
- **pre-learn-evo**: in the third group, networks were trained by competitive Hebbian learning, first using genetically defined pretraining samples, then using input samples.

In summary, these three groups represented pure learning, pure genetic control, and a technique combining prenatal development and learning. The authors found that the two evolutionary models, "evo" and "pre-learn-evo", were far superior to "learn" in classifying hand-written characters. Furthermore, the "pre-learn-evo" type completed the task in a fraction of the time taken by "evo". They argued that the prenatal learning stage could be replaced with alternative forms of development for similar results. Valsalam et al. concluded that their prenatal stage implemented a form of bias on the space of neural models, which could be adjusted by evolution to adapt the particular network to the problem at hand. A more recent study by Tonelli and Mouret also shows that a combination of development (via map-based and HyperNEAT-like encodings) and plasticity can lead to improved learning efficacy, which they attribute to the increased propensity toward the generation of symmetric networks [277] (see also Chap. 9).

These studies are perfectly in line with the view of *development as a means of achieving useful phenotypic biases*, in this case via Hebbian learning. In a sense, some of these algorithms pose a challenge to the existence of developmental modelling in general, with the suggestion that very simple static topologies might be sufficient for intelligent behaviour when subjected to proper epigenetic mechanisms. Perhaps one of the most striking examples is given by the work of Bednar and colleagues:

**Chapter 7: Constructing complex systems via activity-driven unsupervised Hebbian self-organization.**

In Chap. 7, Bednar summarizes his recent work on exploring the use of Hebbian learning as a mechanism for the recreation of phenomena associated with the visual cortex. Starting from highly regular topologies, a simple form of Hebbian learning is applied. Through learning and the appropriate design of simple and complex cell layers, the major functional properties of the primary visual cortex emerge: receptive fields, selective topographic maps, surround modulation, visual contrast, and temporal responses. This impressive array of functional responses is notable for emerging simultaneously from a highly simple neural model, a result which suggest that most of the development and function of the first layer of the primary visual cortex can be viewed as an instance of unsupervised learning. Bednar goes on to discuss the lessons available from his model for the design of complex data-processing systems [13].

As in biology, it is difficult to determine whether certain phenomena should considered "strictly" developmental (in the sense of genetic control), or whether they depend on epigenetic processes. In reality, almost all scenarios integrate both mechanisms in a tight feedback loop. No amount of genetic information can control the fate and behavior of each cell, therefore a great many details have to depend on their interactions with one another and with environmental stimuli (which, for the most part, arise from the cell assembly itself). This is why a combination of developmental and epigenetic mechanisms will also be necessary in the simulation of intelligent networks. We summarize below three active areas of research that we characterize as epigenetic models: Hebbian pretraining, constructive and pruning algorithms, and epigenetic neuroevolution.

## *6.1 Hebbian Pretraining*

Several recent models have explored the addition of Hebbian learning to bio-inspired representations. These have used reservoir computing instead of simpler feed-forward networks, and have concentrated on how to initialize and pretrain the reservoir.

Self-Organizing Recurrent Neural Network (SORN) is a model by Lazar et al. [168], which develops a recurrent neural network reservoir with a particular connectivity for inhibitory neurons. It includes three plasticity mechanisms: intrinsic plasticity, STDP, and synaptic normalization. SORN is trained via an echo state approach, and contrasted against static reservoirs and more limited forms of the model. The authors show that the conjunction of the three forms of plasticity outperform other configurations on simple learning tasks: counting and occluding. There is further suggestion that the organization of neural systems might be predictable from

the model. Zheng et al. [309] have constructed a version of SORN in which structure was a result only of internal plasticity (i.e. no external inputs), and tested over a range of parameters. They discovered that an emergent consequence of the model was a log-normal weight distribution, which resembles the organization found in nature, and has been implicated in the computational capacities of network in general. This suggests that the plasticity mechanisms alone in the absence of environmental stimuli are capable of generating useful organizational principles in a model cortex.

Yin et al. consider the addition of Hebbian learning mechanisms to a recurrent reservoir approach [304]. In this model, a genetic regulatory network specifies Hebbian and anti-Hebbian learning to generate plasticity parameters. The role of the genome here is to create a particular form of plasticity suitable for the problem at hand. An initially complete reservoir is then pruned according to the interplay between input and the GRN, leading to a sparse and pretrained reservoir. The networks are trained via "backpropagation through time" (BPTT), and evaluated on a collection of vision-based tasks with favourable results. Similar work has also been shown to have value in Liquid State Machines [220, 221].

## *6.2 Constructive and Pruning Algorithms*

Closely related to the notion of using simulations of neural development are domains such as constructive neural networks (CoNNs) and pruning networks. Both are families of network design algorithms that operate by gradually changing a network structure in response to training data. They are designed to explore artificial versions of neural organization starting from two opposite viewpoints: CoNNs instantiate a form of *constructivist* process, whereas pruning networks illustrate a *selectivist* process.

In constructive algorithms, a small initial network (sometimes a single hidden neuron) is gradually transformed into a large network in a series of iterations. The network is trained until convergence or until some other stopping criterion has been met. Based on output from this training, the algorithm either terminates or adds more neurons or connections to the network. Once a global termination criterion is reached, the final, larger network is returned, possibly for additional training.

Perhaps the most popular CoNN algorithm is the cascade-correlation architecture [83], which has spawned numerous variants. In a recent review, Nicoletti et al. [58] have compiled a list of models and design decisions which characterize different CoNN approaches. More recent work has concentrated on network growth based on sensitivity analysis [106], adaptive neural activation functions [257], extreme learning machines [308], and extending CoNN to reinforcement learning [131].

In contrast, pruning algorithms start with a large network and gradually remove nodes. Initially, some large network is generated and trained. Next, particular neurons are selected as unimportant, and those neurons are deleted or merged. This process iterates until some global stopping criterion is reached, and finally, a smaller network is returned. Pruning algorithms are less restrained than CoNN algorithms, as

pruning plays a role in many different forms of neural networks and at different times. Deciding which neurons to prune can be made in many ways: for example, neurons that are highly correlated, neurons connected via weak weights, neurons with little influence over outputs, or neurons identified by more complex procedures, such as Optimal Brain Damage [300] and the Optimal Brain Surgeon [111]. A general disadvantage to pruning is that the use of large networks as a starting point tends to require significant computational effort. Recent work on pruning algorithms has included decisions to prune based on sensitivity analysis [167], component analysis [216], and competitive pressures [269]. Extensions to extreme learning machines [195] and other applications [306] have also been tried.

In their simplest forms, both types of algorithms are greedy and can fall prey to "architectural local optima" [6]. However, modern variants are more complex and less easily characterized. One such example, AMGA (adaptive merging and growing algorithm) comes from Islam et al. [136]. AMGA generates a network by both construction and pruning, using adaptive rules as triggers. Between iterations, a given network is trained via backpropagation. Construction occurs by splitting an existing hidden node, which results in the preservation of the behavioural linkages among neurons. Pruning occurs by merging highly correlated hidden nodes. AMGA is highly effective at supervised learning, outperforming several other neural and SVM techniques. The authors hypothesize that constructive-pruning hybrid techniques successfully avoid the local optima that hindered previous algorithms. Many such hybrid techniques have been explored [26, 105, 128, 130, 209, 232, 299].

CoNN and pruning algorithms are inspired by development, although motivations differ somewhat from those of developmental systems. They generally target the most parsimonious network possible, and show little interest for a parallel implementation of the algorithms, since they often rely on global data structures such as inter-neural correlations or Hessian matrices. Regardless, these techniques provide insight into how to execute ontogenic and epigenetic processes simultaneously.

## *6.3 Epigenetic Neuroevolution*

Other authors have explored techniques that could be characterized as *epigenetic neuroevolution* as they offer a combination of evolutionary algorithms and learning techniques operating in tandem. Researchers in this category hope that such combination might return the best of both worlds: the high accuracy associated with epigenetic training and the capacity to explore a wide space of possible networks associated with neuroevolution, leading together to the ability to generalize to new environmental stimuli. Some authors also hope to avoid the architectural local minima generated by other non-evolutionary techniques[4]

[4] Caveat: while neuroevolution is known to be more versatile than, for instance, classic CoNN algorithms, it is also known that evolutionary computation will be often hindered by local optima in the fitness landscape, suggesting a possibly different sort of suboptimality.

A simple early example of epigenetic neuroevolution is outlined in Yao and Liu [301]. It is based on an evolutionary algorithm that controls the topology and weights of a network (via a form of genetic programming), while a learning routine is applied at mid-step to trigger the addition or deletion of neurons in the network. Training accuracy was used as the fitness of an individual during evolution. As the authors later argue, the issue with such a naive approach is that learning techniques based on gradient descent, when initialized with random weights, tend to be very noisy to the point of negatively affecting the evolutionary process. Using an averaged success over several independent learning sessions may provide a solution, but also tends to be too computationally expensive to serve as a fitness function [302].

To alleviate these problems, several authors including Yao and Liu have explored hybrid algorithms where the evolutionary search is global while the learning methods work on a local level. A recent example comes from Oong and Isa [223], who evolved a direct representation of the network via an adjacency matrix. This matrix, however, is augmented with a secondary genetic representation, a "node vector", which applies structural changes to the network topology. The interim success of the network is used to compute a measure of generalization loss, which in turn serves to control the weight of the evolutionary mutation. Thus, for Oong and Isa, instead of letting epigenetic information directly control the change of a network, it is a cue for the meta-process (evolution) to adjust the degree of exploration versus exploitation. Other forms of hybrid evolutionary-epigenetic algorithms, including the use of constructive techniques, have been explored [91].

**Chapter 8: Neuro-centric and holocentric approaches to the evolution of developmental neural networks.**

In Chap. 8, Miller explores two strategies of generating neural networks via neuroevolution. The first, a *neurocentric* approach, involves the detailed modelling of cells in a dynamic, time-based process. In this case, several independent control mechanisms are created, ones which emulate detailed sub-cellular behaviours. The second strategy, a *holocentric* approach, operates on a whole neural network. Here, network-specific operations make changes to sub-graphs of neurons and connections. By contrasting these two approaches, the author explores the value of the inclusion of a detailed and more plausible model of growth and plasticity relative to the additional computational costs involved. The chapter closes with design advice for practitioners [198].

In some cases, an explicit developmental process and a later epigenetic process are both included, which can make development occur twice: as a genotype-phenotype mapping process, and as a plastic property during operation, closely related to learning. Autonomous robotics is one prominent area of application, where neural network controllers are grown from compact genotypes (Sect. 4), and modification to the actual controller may occur during the robot's lifetime, whether the objective is long-term adaptation to the environment [151, 218], memorizing events [88], or learning new capabilities [262, 276].

**Chapter 9: Artificial evolution of plastic neural networks: a few key concepts.**

In Chap. 9, Mouret and Tonelli consider the use of neuroevolution to find plastic neural networks for reinforcement learning. A neuroevolutionary process produces a network topology, which is then trained via Hebbian learning in a (possibly reward-based) environment. Two key motivations for this type of approach are the promotion of behavioural robustness and reward-based behavioural change, concepts which suffer from inconsistent terminology in the literature. The authors provide new definitions of both concepts, and turn their attention to a key issue: the response of a neural network to previously unseen scenarios. To promote research on the topic, they define and discuss relevant concepts, such as the capacity for general and transitive learning, then theorize about the benefits of a developmental stage in terms of general learning [206].

## 7 Summary

In this introduction, we have explored the central hypothesis of this book that *adaptive growth is a means of producing brain-like machines*. The emulation of neural development can incorporate desirable characteristics of natural neural systems into engineered designs. We have reviewed several strategies for performing this "meta-design", which also involves identifying specific network biases and their benefits. In particular, we have seen that several recent studies show a strong synergy, sometimes interchangeability, between developmental and epigenetic processes—a topic that has remained largely under-explored in the literature. The chapters that follow in this book describe some of the most important works in this area, offering a state-of-the-art review of intelligent machine design.

Recent accelerating progress in observation and modelling techniques in neuroscience, and systems biology in general, ensures the continued generation of novel insights into brain organization. This new collection of "biases" should be further explored and exploited in neural networks over the coming years, suggesting that artificial neurogenesis is a promising avenue of research.

## References

1. M. Abeles, *Local Cortical Circuits: An Electrophysiological Study*, vol. 6 (Springer, New York, 1982)
2. W.C. Abraham, M.F. Bear, Metaplasticity: the plasticity of synaptic plasticity. Trends Neurosci. **19**(4), 126–130 (1996)

3. I. Aho, H. Kemppainen, K. Koskimies, E. Makinen, T. Niemi, Searching neural network structures with l systems and genetic algorithms. Int. J. Comput. Math. **73**(1), 55–75 (1999)
4. U. Alon, *An Introduction to Systems Biology: Design Principles of Biological Circuits* (CRC Press, Boca Raton, 2007)
5. T. Andersen, R. Newman, T. Otter, Development of virtual embryos with emergent self-repair. in *AAAI Fall Symposium* (2006)
6. P.J. Angeline, G.M. Saunders, J.B. Pollack, An evolutionary algorithm that constructs recurrent neural networks. IEEE Trans. Neural Netw. **5**(1), 54–65 (1994)
7. W. Arthur, The effect of development on the direction of evolution: toward a twenty-first century consensus. Evol. Dev. **6**(4), 282–288 (2004)
8. F.A.C. Azevedo, L.R.B. Carvalho, L.T. Grinberg, J.M. Farfel, R.E.L. Ferretti, R.E.P. Leite, W.J. Filho, R. Lent, S. Herculano-Houzel, Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain. J. Comp. Neurol. **513**(5), 532–541 (2009)
9. J.M. Baldwin, A new factor in evolution. Am. Nat. **30**, 441–451 (1896)
10. W. Banzhaf, On the dynamics of an artificial regulatory network. in *European Conference on Artificial Life (ECAL 2003)* (Springer, Berlin, 2003), pp. 217–227
11. W. Banzhaf, N. Pillay, Why complex systems engineering needs biological development. Complexity **13**(2), 12–21 (2007)
12. J. Beal, Functional blueprints: an approach to modularity in grown systems. Swarm Intell. **5**(3–4), 257–281 (2011)
13. J.A. Bednar, *Constructing Complex Systems Via Activity-Driven Unsupervised Hebbian Self-organization*. in ed. by Kowaliw et al. [160], pp. 216–241
14. Y. Bengio, *Evolving Culture Versus Local Minima*. in ed. by Kowaliw et al. [160], pp. 112–143
15. Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, Greedy layer-wise training of deep networks. in *Advances in Neural Information Processing Systems* (2007)
16. Y. Bengio, Y. LeCun, Scaling learning algorithms towards AI. in *Large Scale Kernel Machines* (MIT Press, Cambridge, 2007)
17. P. Bentley, Three ways to grow designs: a comparison of embryogenies for an evolutionary design problem. in *Conference on Genetic and Evolutionary Computation* (1999), pp. 35–43
18. P. Bentley, Investigations into graceful degradation of evolutionary developmental software. Nat. Comput. **4**(4), 417–437 (2005)
19. E. Bienenstock, R. Doursat, Spatio-temporal coding and the compositionality of cognition. in *Temporal Correlations and Temporal Coding in the Brain* (1990), pp. 42–47
20. E. Bienenstock, C. von der Malsburg, A neural network for invariant pattern recognition. Europhys. Lett. **4**(1), 121–126 (1987)
21. E. Bienenstock, R. Doursat, A shape-recognition model using dynamical links. Netw. Comput. Neural Syst. **5**(2), 241–258 (1994)
22. E.J.W. Boers, H. Kuiper, *Biological Metaphors and the Design of Modular Artificial Neural Networks* Technical report (Leiden University, 1992)
23. J.C. Bongard, Evolving modular genetic regulatory networks. in *IEEE Congress on Evolutionary Computation (CEC)* (2002), pp. 1872–1877
24. J.C. Bongard, Spontaneous evolution of structural modularity in robot neural network controllers. in *Conference on Genetic and Evolutionary Computation (GECCO)* (Springer, 2011)
25. J.C. Bongard, R. Pfeifer, Evolving complete agents using artificial ontogeny. in ed. by F. Hara, R. Pfeifer *Morpho-functional Machines: The New Species (Designing Embodied Intelligence)* (Springer, 2003), pp. 237–258
26. M. Bortman, M. Aladjem, A growing and pruning method for radial basis function networks. IEEE Trans. Neural Netw. **20**(6), 1039–1045 (2009)
27. R. Brette, M. Rudolph, N.T. Carnevale, M.L. Hines, D. Beeman, J. Bower, M. Diesmann, A. Morrison, P. Goodman, F. Harris Jr, M. Zirpe, T. Natschläger, D. Pecevski, G.B. Ermentrout, M. Djurfeldt, A. Lansner, O. Rochel, T. Viéville, E. Muller, A.P. Davison, S. El Boustani, A. Destexhe, Simulation of networks of spiking neurons: A review of tools and strategies. J. Comput. Neurosci. **23**(3), 349–398 (2007)

28. D.J. Cahalane, B. Clancy, M.A. Kingsbury, E. Graf, O. Sporns, B.L. Finlay, Network structure implied by initial axon outgrowth in rodent cortex: empirical measurement and models. PLoS ONE **6**(1), 01 (2011)
29. W. Callebaut, D. Rasskin-Gutman, *Modularity: Understanding the Development and Evolution of Natural Complex Systems* (MIT Press, 2005)
30. A. Cangelosi, D. Parisi, S. Nolfi. Cell division and migration in a genotype for neural networks. Conf. Comput. Netw. 497–515 (1994)
31. S. Carroll, J. Grenier, S. Weatherbee, *From DNA to Diversity: Molecular Genetics and the Evolution of Animal Design* 2nd edn (Blackwell Publishing, 2005)
32. R. Cattell, A. Parker, Challenges for brain emulation: why is it so difficult? Nat. Intell. INNS Mag. **1**(3), 17–31 (2012)
33. L. Cazenille, N. Bredeche, H. Hamann, J. Stradner, Impact of neuron models and network structure on evolving modular robot neural network controllers. in *Conference on Genetic and evolutionary computation (GECCO)*, (ACM Press, New York, 2012), p. 89
34. J.P. Changeux, A. Danchin, Nature **264**, 705–712 (1976)
35. N. Cheney, R. Maccurdy, J. Clune, H. Lipson, Unshackling evolution: evolving soft robots with multiple materials and a powerful generative encoding. in *Genetic and Evolutionary Computation Conference (GECCO)* (2013), pp. 167–174
36. N. Chomsky, *Aspects of the Theory of Syntax* (MIT Press, 1965)
37. J. Clune, B.E. Beckmann, P.K. McKinley, C. Ofria, Investigating whether hyperNEAT produces modular neural networks. in *Conference on Genetic and Evolutionary Computation (GECCO)*, (ACM Press, New York, 2010), pp. 1523–1530
38. J. Clune, B.E. Beckmann, R.T. Pennock, C. Ofria, HybrID: A hybridization of indirect and direct encodings for evolutionary computation. in *European Conference on Artificial Life (ECAL)* (2009), pp. 134–141
39. J. Clune, J.B. Mouret, H. Lipson, The evolutionary origins of modularity. Proc. Roy. Soc. B Biol. Sci. **280**(1755), 20122863 (2013)
40. J. Clune, K.O. Stanley, R.T. Pennock, C. Ofria, On the performance of indirect encoding across the continuum of regularity. IEEE Trans. Evol. Comput. **15**(3), 346–367 (2011)
41. H. Cuntz, F. Forstner, A. Borst, M. Häusser, One rule to grow them all: a general theory of neuronal branching and its practical application. PLoS Comput. Biol. **6**(e1000877), 08 (2010)
42. S. Cussat-Blanc, H. Luga, Y. Duthen, Cell 2Organ: Self-repairing artificial creatures thanks to a healthy metabolism. in *IEEE Congress on Evolutionary Computation (CEC)* (2009), pp. 2708–2715
43. S. Cussat-Blanc, J. Pascalie, S. Mazac, H. Luga, Y. Duthen, A synthesis of the cell2organ developmental model. in Doursat et al. [67], pp. 353–381
44. G. Cybenko, Approximations by superpositions of sigmoidal functions. Math. Control Sig. Syst. **4**(2), 303–314 (1989)
45. N.M. da Costa, K.A.C. Martin, Whose cortical column would that be? Front. Neuroanat. **4**(16), (2010)
46. G. Dahl, M. Ranzato, A. Mohamed, G.E. Hinton, Phone Recognition with the mean-covariance restricted Boltzmann machine. in *Advances in Neural Information Processing Systems* (2010), pp. 469–477
47. K. Dale, P. Husbands, The evolution of reaction-diffusion controllers for minimally cognitive agents. Artif. Life **16**, 1–19 (2010)
48. D.B. DAmbrosio, J. Gauci, K.O. Stanley, HyperNEAT: the first five years. in ed. by Kowaliw et al. [160], pp. 167–197
49. D.B. D'Ambrosio, K.O. Stanley, A novel generative encoding for exploiting neural network sensor and output geometry. in *Conference on Genetic and Evolutionary Computation (GECCO)* (ACM Press, New York, 2007), pp. 974–982
50. H. De Garis, Growing an artificial brain: the genetic programming of million-neural-net-module artificial brains within trillion cell cellular automata machines. in *Proceedings of the Third Annual Conference on Evolutionary Programming* (1994), pp. 335–343
51. T.W. Deacon, Rethinking mammalian brain evolution. Am. Zool. **30**(3), 629–705 (1990)

52. J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, Q. Le, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, A. Ng, Large scale distributed deep networks. in ed. by P. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, K.Q. Weinberger. *Advances in Neural Information Processing Systems 25* (2012), pp. 1232–1240
53. A.S. Dekaban, D. Sadowsky, Changes in brain weights during the span of human life: Relation of brain weights to body heights and body weights. Ann. Neurol. **4**(4), 345–356 (1978)
54. F. Dellaert, R.D. Beer, *Co-evolving Body and Brain in Autonomous Agents Using a Developmental Model*. Technical report, Department of Computer Engineering and Science (Case Western Reserve University, Cleveland, 1994)
55. F. Dellaert, R.D. Beer, Toward an evolvable model of development for autonomous agent synthesis. in *Proceedings of the fourth International Workshop on the Synthesis and Simulation of Living Systems (ALIFE Workshop)* (1994)
56. A. Deutsch, S. Dormann, *Cellular Automaton Modelling of Biological Pattern Formation: Characterization, Applications and Analysis* (Birkhauser, 2005)
57. A. Devert, N. Bredeche, M. Schoenauer, Robustness and the halting problem for multicellular artificial ontogeny. IEEE Trans. Evol. Comput. **15**(3), 387–404 (2011)
58. M. do Carmo Nicoletti, J. Bertini, D. Elizondo, L. Franco, J. Jerez, Constructive neural network algorithms for feedforward architectures suitable for classification tasks. in ed. by L. Franco, D. Elizondo, J. Jerez. *Constructive Neural Networks, Studies in Computational Intelligence*, vol. 258 (Springer, Heidelberg, 2009), pp. 1–23
59. S. Doncieux, J.-B. Mouret, T. Pinville, P. Tonelli, B. Girard, The evoneuro approach to neuroevolution. in Kowaliw et al. [159], pp. 10–14
60. R. Doursat, Bridging the mind-brain gap by morphogenetic neuron flocking: The dynamic self-organization of neural activity into mental shapes. in *2013 AAAI Fall Symposium Series* (2013)
61. R. Doursat, Contribution à l'étude des représentations dans le système nerveux et dans les réseaux de neurones formels. PhD thesis, Université Pierre et Marie Curie (Paris 6), 1991
62. R. Doursat, Facilitating evolutionary innovation by developmental modularity and variability. in *Conference on Genetic and Evolutionary Computation (GECCO)* (ACM, 2009), pp. 683–690
63. R. Doursat, Organically grown architectures: creating decentralized, autonomous systems by embryomorphic engineering. in ed. by R.P. Würtz. *Organic computing, Understanding Complex Systems* (Springer, 2008), pp. 167–199
64. R. Doursat, The growing canvas of biological development: multiscale pattern generation on an expanding lattice of gene regulatory networks. InterJournal Complex Syst. 1809 (2006)
65. R. Doursat, E. Bienenstock, Neocortical self-structuration as a basis for learning. in *5th International Conference on Development and Learning (ICDL 2006)* (2006), pp. 1–6
66. R. Doursat, C. Sánchez, R. Dordea, D. Fourquet, T. Kowaliw, Embryomorphic engineering: emergent innovation through evolutionary development. in ed. by Doursat et al. [67], pp. 275–311
67. R. Doursat, H. Sayama, O. Michel (eds.), *Morphogenetic Engineering: Toward Programmable Complex Systems. Understanding Complex Systems* (Springer, 2012)
68. R. Doursat, H. Sayama, O. Michel, A review of morphogenetic engineering. *Nat. Comput.* 1–19 (2013)
69. J.E. Dowling, *The Great Brain Debate: Nature or Nurture?* (Princeton University Press, 2007)
70. K. Downing, A neural-group basis for evolving and developing neural networks. in *AAAI-Devp* (2006)
71. K. Downing, Supplementing evolutionary developmental systems with abstract models of neurogenesis. in *9th Genetic and Evolutionary Computation Conference (GECCO)* (2007), pp. 990–996
72. K. Downing, The Baldwin effect in developing neural networks. in *Genetic and Evolutionary Computation Conference (GECCO)* (2010), pp. 555–562
73. P. Durr, C. Mattiussi, D. Floreano, Neuroevolution with analog genetic encoding. in *Parallel Problem Solving from Nature (PPSN)* (2006), pp. 671–680

74. S.O.E. Ebbesson, The parcellation theory and its relation to interspecific variability in brain organization, evolutionary and ontogenetic development, and neuronal plasticity. Cell Tissue Res. **213**(2), 179–212 (1980)
75. P. Eggenberger Hotz, Creation of neural networks based on developmental and evolutionary principles. in *International Conference on Artificial, Neural Networks* (1997), pp. 337–342
76. P. Eggenberger Hotz, Evolving morphologies of simulated 3D organisms based on differential gene expression. in *European Conference on Artificial Life (ECAL)* (MIT Press, 1997), pp. 205–213
77. P. Eggenberger Hotz, Evolving morphologies of simulated 3D organisms based on differential gene expression. in *European Conference on Artificial Life (ECAL)* (1997), pp. 205–213
78. P. Eggenberger Hotz, G. Gomez, R. Pfeiffer, Evolving the morphology of a neural network for controlling a foveating retina and its test on a real robot. in *Artificial Life 8* (2002), pp. 243–251
79. A.E. Eiben, J.E. Smith, *Introduction to Evolutionary Computing* (Springer, 2003)
80. C. Eliasmith, T.C. Stewart, X. Choo, T. Bekolay, T. DeWolf, Y. Tang, D. Rasmussen, A large-scale model of the functioning brain. Science **338(20)**, 1202-1205 (2012)
81. D. Erhan, Y. Bengio, A. Courville, P.A. Manzagol, P. Vincent, S. Bengio, Why does unsupervised pre-training help dDeep learning? J. Mach. Learn. Res. **11**, 625–660 (2010)
82. C. Espinosa-Soto, A. Wagner, Specialization can drive the evolution of modularity. PLoS Comput. Biol. **6**(3), e1000719 (2010)
83. S.E. Fahlman, C. Lebiere, The cascade-correlation learning architecture. in ed. by D.S. Touretzky. *Advances in Neural Information Processing Systems 2*, (Morgan Kaufmann, 1990), pp. 524–532
84. D. Federici, Evolving a neurocontroller through a process of embryogeny. in *Proceeding of Simulation of Adaptive Behavior (SAB)* (2004), pp. 373–384
85. D. Federici, Evolving developing spiking neural networks. in *IEEE Congress on Evolutionary Computation* (2005), pp. 43–550
86. D. Federici, K. Downing, Evolution and development of a multicellular organism: Scalability, resilience, and neutral complexification. Artif. Life **12**(3), 381–409 (2006)
87. J.D. Fernández, D. Lobo, G.M. Martín, R. Doursat, F.J. Vico, Emergent diversity in an open-ended evolving virtual community. Artif. Life **18**(2), 199–222 (2012)
88. D. Floreano, J. Urzelai, Neural morphogenesis, synaptic plasticity, and evolution. Theory Biosci. **120**(3–4), 225–240 (2001)
89. J.A. Fodor, *Modularity of Mind: An Essay on Faculty Psychology* (MIT Press, 1983)
90. R.M. French, Catastrophic forgetting in connectionist networks. Trends Cogn. Sci. **3**(4), 128–135 (1999)
91. N. García-Pedrajas, D. Ortiz-Boyer, A cooperative constructive method for neural networks for pattern recognition. Pattern Recogn. **40**(1), 80–98 (2007)
92. S. Geman, E. Bienenstock, R. Doursat, Neural networks and the bias/variance dilemma. Neural Comput. **4**(1), 1–58 (1992)
93. S.F. Gilbert, *Developmental Biology* 8 edn. (Sinauer Associates, 2008)
94. S.F. Gilbert, D. Epel, *Ecological Developmental Biology* 1 edn. (Sinauer Associates, 2008)
95. B. Goertzel, R. Lian, I. Arel, H. de Garis, S. Chen, A world survey of artificial brain projects, part II: biologically inspired cognitive architectures. Neurocomputing **74**(1–3), 30–49 (2010)
96. F. Gomez, R. Miikkulainen, Solving non-markovian control tasks with neuro-evolution. in *IJCAI* (1999), pp. 1356–1361
97. F. Gomez, R. Miikkulainen, Incremental evolution of complex general behavior. Adapt. Behav. **5**, 317–342 (1997)
98. B.C. Goodwin, *How the Leopard Changed Its Spots: The Evolution of Complexity* (Scribner, 1994)
99. S.J. Gould, *The Structure of Evolutionary Theory* (The Belknap Press of Harvard University Press, 2002)
100. S.J. Gould, R. Lewontin, The spandrels of san marco and the panglossian paradigm: a critique of the adaptationist programme. Proc. Roy. Soc. London Ser. B Biol. Sci. **205**(1161), 581–598 (1979)

101. F. Gruau, Cellular encoding as a graph grammar. in *Grammatical Inference: IEE Colloquium on Theory, Applications and Alternatives* (1993), pp. 1–17
102. F. Gruau, Genetic synthesis of Boolean neural networks with a cell rewriting developmental process. in *Proceedings of COGANN-92: International Workshop on Combinations of Genetic Algorithms and Neural Networks* (IEEE Computer Society Press, 1992), pp. 55–74
103. F. Gruau, *Neural Network Synthesis Using Cellular Encoding And The Genetic Algorithm* (PhD thesis, Université Claude Bernard-Lyon, 1994)
104. F. Gruau, D. Whitley, L. Pyeatt, A comparison between cellular encoding and direct encoding for genetic neural networks. in *Conference on Genetic Programming* (1996), pp. 81–89
105. H.-G. Han, J.-F. Qiao, A structure optimization algorithm for feedforward neural network construction. Neurocomputing **99**, 347–357 (2012)
106. H.-G. Han, J.-F. Qiao, A repair algorithm for radial basis function neural network and its application to chemical oxygen demand modeling. Int. J. Neural Syst. **20**(01), 63–74 (2010)
107. S. Harding, W. Banzhaf, Artificial development. in *Organic Computing, Understanding Complex Systems* (Springer, Heidelberg, 2008), pp. 201–219
108. S.L. Harding, J.F. Miller, The dead state: A comparison between developmental and direct encodings (updated version). in *Workshop on Complexity through Development and Self-Organizing Representations (CODESOAR), Genetic and Evolutionary Computation Conference (GECCO)* (2006)
109. S.L. Harding, J.F. Miller, W. Banzhaf, Self-modifying cartesian genetic programming, in ed. by J.F. Miller *Cartesian Genetic Programming*, Natural Computing Series (Springer, Berlin, 2011), pp. 101–124
110. C. Hartland, N. Bredeche, M. Sebag, Memory-enhanced evolutionary robotics: the echo state network approach. in *IEEE Congress on Evolutionary Computation, 2009 (CEC)* (2009), pp. 2788–2795
111. B. Hassibi, D.G. Stork, Second order derivatives for network pruning: Optimal brain surgeon. in *Advances in Neural Information Processing Systems* (1993), pp. 164–164
112. J. Hastad, Almost optimal lower bounds for small depth circuits. in *Proceedings of the Eighteenth Annual ACM Aymposium on Theory of Computing, STOC '86* (ACM, New York, 1986), pp. 6–20
113. S. Haykin, *Neural Networks and Learning Machines* 3 edn. (Pearson Inc., 2009)
114. D.O. Hebb, *The Organization of Behavior* (Wiley, New York, 1949)
115. J.L. Hendrikse, T.E. Parsons, B. Hallgrímsson, Evolvability as the proper focus of evolutionary developmental biology. Evol. Dev. **9**(4), 393–401 (2007)
116. S.L. Hill, Y. Wang, I. Riachi, F. Schürman, H. Markram, Statistical connectivity provides a sufficient foundation for specific functional connectivity in neocortical neural microcircuits. vol. 18 *Proceedings of the National Academy of Sciences* (2012)
117. J. Hiller, H. Lipson, Automatic design and manufacture of soft robots. IEEE Trans. Robot. **28**, 457–466 (2012)
118. R. Himeno, J. Savin, Largest neuronal network simulation achieved using K computer @ONLINE. http://www.riken.jp/en/pr/press/2013/20130802_1/. Accessed: 09/2013
119. G.E. Hinton, S. Osindero, Y.W. Teh, A fast learning algorithm for deep belief nets. Neural Comput. **18**(7), 1527–1554 (2006)
120. G.E. Hinton, S.J. Nowlan, How learning can guide evolution. Complex Syst. **1**, 495–502 (1987)
121. A. Hintze, C. Adami, Evolution of complex modular biological networks. PLoS Comput. Biol. **4**(2), 1–12 (2008)
122. T.-H. Hoang, R.I. McKay, D. Essam, N.X. Hoai, On synergistic interactions between evolution, development and layered learning. IEEE Trans. Evol. Comput. **15**(3), 287–312 (2011)
123. J.J. Hopfield, C.D. Brody, What is a moment? transient synchrony as a collective mechanism for spatiotemporal integration. Proc. Natl. Acad. Sci. **98**(3), 1282–1287 (2001)
124. G.S. Hornby, Measuring, enabling and comparing modularity, regularity and hierarchy in evolutionary design. in *Conference on Genetic and Evolutionary Computation (GECCO)* (2007), pp. 1729–1736

125. G.S. Hornby, H. Lipson, J.B. Pollack, Generative representations for the automated design of modular physical robots. IEEE Trans. Robot. Autom. **19**(4), 703–719 (2003)
126. G.S. Hornby, J.B. Pollack, Creating high-level components with a generative representation for body-brain evolution. Artif. Life **8**(3), 223–246 (2002)
127. P.E. Hotz, Comparing direct and developmental encoding schemes in artificial evolution: a case study in evolving lens shapes. in *Congress on Evolutionary Computation (CEC)* (2004), pp. 752–757
128. C.-F. Hsu, Adaptive growing-and-pruning neural network control for a linear piezoelectric ceramic motor. Eng. Appl. Artif. Intell. **21**(8), 1153–1163 (2008)
129. T. Hu, W. Banzhaf, Evolvability and speed of evolutionary algorithms in light of recent developments in biology. J. Artif. Evol. Appl. **1–28**, 2010 (2010)
130. D.-S. Huang, J.-X. Du, A constructive hybrid structure optimization methodology for radial basis probabilistic neural networks. IEEE Trans. Neural Netw. **19**(12), 2099–2115 (2008)
131. A. Huemer, M. Gongora, D. Elizondo, A robust reinforcement based self constructing neural network. in *International Joint Conference on Neural Networks (IJCNN)* (2010), pp. 1–7
132. P. Husbands, T. Smith, N. Jakobi, M. O'Shea, Better living through chemistry: evolving GasNets for robot control. Connection Sci. **10**(3–4), 185–210 (1998)
133. A. Ilachinski, *Cellular Automata: A Discrete Universe* (World Scientific, 2001)
134. B. Inden, Neuroevolution and complexifying genetic architectures for memory and control tasks. Theory Biosci. **127**(2), 187–194 (2008)
135. T. Ishibashi, K. Dakin, B. Stevens, P. Lee, S. Kozlov, C. Stewart, R. Fields, Astrocytes promote myelination in response to electrical impulses. Neuron **49**(6), 823–832 (2006)
136. M.M. Islam, A. Sattar, F. Amin, Xin Yao, K. Murase, A new adaptive merging and growing algorithm for designing artificial neural networks. IEEE Trans. Syst. Man Cyber. Part B Cybern. **39**(3), 705–722 (2009)
137. H. Jäeger, H. Haas, Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. Science **304**(5667), 78–80 (2004)
138. H. Jäeger, M. Lukoševičius, D. Popovici, U. Siewert, Optimization and applications of echo state networks with leaky- integrator neurons. Neural Netw. **20**(3), 335–352 (2007)
139. H. Jäeger, W. Maass, J. Principe, Introduction to the special issue on echo state networks and liquid state machines. Neural Netw. **20**(3), 287–289 (2007)
140. N. Jakobi, Harnessing morphogenesis. in *International Conference on Information Processing in Cells and Tissues* (1995), pp. 29–41
141. K. Jarrett, K. Kavukcuoglu, M. Ranzato, Y. LeCun, What is the best multi-stage architecture for object recognition? in *Proceedings of International Conference on Computer Vision (ICCV)* (2009), pp. 2146–2153
142. M. Joachimczak, T. Kowaliw, R. Doursat, B. Wróbel, Brainless bodies: controlling the development and behavior of multicellular animats by gene regulation and diffusive signals. in *Conference on the Simulation and Synthesis of Living Systems (ALife)*, (2012), pp. 349–356
143. M. Joachimczak, T. Kowaliw, R. Doursat, B. Wróbel, Controlling development and chemotaxis of soft-bodied multicellular animats with the same gene regulatory network. in *Advances in Artificial Life (ECAL)* (MIT Press, 2013), pp. 454–461
144. M. Joachimczak, B. Wróbel, Processing signals with evolving artificial gene regulatory networks. in *Conference on the Simulation and Synthesis of Living Systems (ALife)* (MIT Press, 2010), pp. 203–210
145. M. Joachimczak, B. Wróbel, Evolution of robustness to damage in artificial 3-dimensional development. Biosystems **109**(3), 498–505 (2012)
146. M. Kaiser, C.C. Hilgetag, A. von Ooyen, A simple rule for axon outgrowth and synaptic competition generates realistic connection lengths and filling fractions. Cereb. Cortex **19**(12), 3001–3010 (2009)
147. N. Kashtan, U. Alon, Spontaneous evolution of modularity and network motifs. Proc. Natl. Acad. Sci. **102**(39), 13773 (2005)
148. Y. Kassahun, G. Sommer, Evolution of neural networks through incremental acquisition of neural structures. Technical Report Number 0508, Christian-Albrechts-Universität zu Kiel, Institut für Informatik und Praktische Mathematik, Juni 2005

149. M.J. Katz, R.J. Lasek, Evolution of the nervous system: Role of ontogenetic mechanisms in the evolution of matching populations. Proc. Natl. Acad. Sci. **75**(3), 1349–1352 (1978)
150. S.A. Kauffman, *The Origins of Order: Self Organization and Selection in Evolution* (Oxford University Press, Oxford, 1993)
151. G.M. Khan, J.F. Miller, D.M. Halliday, Evolution of cartesian genetic programs for development of learning neural architecture. Evol. Comput. **19**(3), 469–523 (2011)
152. M.W. Kirschner, J.C. Gerhart, *The Plausibility of Life: Resolving Darwin's Dilemma* (Yale University Press, 2005)
153. H. Kitano, Designing neural networks using genetic algorithms with graph generation system. Complex Syst. **4**, 461–476 (1990)
154. H. Kitano, A Simple Model of Neurogenesis and Cell Differentiation based on Evolutionary Large-Scale Chaos. Artif. Life **2**, 79–99 (1995)
155. J. Kodjabachian, J.-A. Meyer, Evolution and development of neural networks controlling locomotion, gradient-following and obstacle avoidance in artificial insects. IEEE Trans. Neural Netw. **9**(5), 796–812 (1998)
156. M. Komosinski, The world of framsticks: simulation, evolution, interaction. in *Virtual Worlds* (2000), pp. 214–224
157. T. Kowaliw, W. Banzhaf, Augmenting artificial development with local fitness. in ed. by A. Tyrrell *IEEE Congress on Evolutionary Computation (CEC)* (2009), pp. 316–323
158. T. Kowaliw, W. Banzhaf, Mechanisms for complex systems engineering through artificial development. in ed. by Doursat et al. [67], pp. 331–351
159. T. Kowaliw, N. Bredeche, R. Doursat (eds.), *Growing Adaptive Machines: Combining Development and Learning in Artificial Neural Networks* (Springer, 2014)
160. T. Kowaliw, N. Bredeche, R. Doursat (eds.), *Proceedings of DevLeaNN: A Workshop on Development and Learning in Artificial Neural Networks* (Paris, France, 2011)
161. T. Kowaliw, P. Grogono, N. Kharma, Bluenome: A novel developmental model of artificial morphogenesis. in *Conference on Genetic and Evolutionary Computation (GECCO)* (2004), pp. 93–104
162. T. Kowaliw, P. Grogono, N. Kharma, Environment as a spatial constraint on the growth of structural form. in *Conference on Genetic and Evolutionary Computation (GECCO)* (2007), pp. 1037–1044
163. T. Kowaliw, P. Grogono, N. Kharma, The evolution of structural form through artificial embryogeny. in *IEEE Symposium on Artificial Life (ALIFE)* (2007), pp. 425–432
164. J.R. Koza, D. Andre, F.H Bennett III, M. Keane, *Genetic Programming 3: Darwinian Invention and Problem Solving* (Morgan Kaufman, 1999)
165. J.L. Krichmar, G.M. Edelman, Brain-based devices for the study of nervous systems and the development of intelligent machines. Artif. Life **11**(1–2), 63–77 (2005)
166. A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks. in ed. by P. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, K.Q. Weinberger *Advances in Neural Information Processing Systems 25* (2012), pp. 1106–1114
167. P. Lauret, E. Fock, T.A. Mara, A node pruning algorithm based on a fourier amplitude sensitivity test method. IEEE Trans. Neural Netw. **17**(2), 273–293 (2006)
168. A. Lazar, G. Pipa, J. Triesch, SORN: a self-organizing recurrent neural network. Front. Comput. Neurosci. **3**(23), 1–9 (2009)
169. Q. Le, A. Karpenko, J. Ngiam, A.Y. Ng, Ica with reconstruction cost for efficient overcomplete feature learning. in ed. by J. Shawe-Taylor, R.S. Zemel, P. Bartlett, F.C.N. Pereira, K.Q. Weinberger *Advances in Neural Information Processing Systems 24* (2011), pp. 1017–1025
170. Q. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. Corrado, J. Dean, A. Ng, Building high-level features using large scale unsupervised learning, in ed. by J. Langford, J. Pineau *Proceedings of the 29th International Conference on Machine Learning (ICML-12), ICML '12* (Omnipress, New York, 2012), pp. 81–88
171. Y. LeCun, Y. Bengio, Convolutional networks for images, speech, and time series. in *The Handbook of Brain Theory and Neural Networks* (MIT Press, 1998)

172. J. Lefèvre, J.-F. Mangin, A reaction-diffusion model of human brain development. PLoS Comput. Biol. **6**(4) e1000749 (2010)
173. M. Li, P.M.B. Vitányi, *An Introduction to Kolmogorov Complexity and Its Applications* 3rd edn. (Springer, 2008)
174. H. Lipson, Principles of modularity, regularity, and hierarchy for scalable systems. J. Biol. Phys. Chem. **7**, 125–128 (2007)
175. J. Lohn, G. Hornby, D. Linden, Evolutionary antenna design for a NASA spacecraft. in *Genetic Programming Theory and Practice II* Chap. 18 (Springer, Ann Arbor, 2004), pp. 301–315
176. R.L. Lopes, E. Costa, The regulatory network computational device. Genetic Program. Evolvable Mach. **13**, 339–375 (2012)
177. C.J. Lowe, G.A. Wray, Radical alterations in the roles of homeobox genes during echinoderm evolution. Nature **389**, 718–721 (1997)
178. S. Luke, L. Spector, Evolving graphs and networks with edge encoding : preliminary report. in *Late Breaking Papers at the Genetic Programming 1996 Conference* (1996), pp. 117–124
179. M. Lukoševičius, H. Jaeger, Reservoir computing approaches to recurrent neural network training. Comput. Sci. Rev. **3**(3), 127–149 (2009)
180. W. Maass, T. Natschläger, H. Markram, Real-time computing without stable states: a new framework for neural computation based on perturbations. Neural Comput. **14**(11), 2531–2560 (2002)
181. J.M. Mandler, *The Foundations of Mind: Origins of Conceptual Thought* (Oxford University Press, Oxford, 2004)
182. H. Markram, A brain in a supercomputer. www.ted.com. Accessed: 27/12/2012
183. H. Markram, J. Lübke, M. Frotscher, B. Sakmann, Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs. Science **275**(5297), 213–215 (1997)
184. H. Markram, The blue brain project. Nat. Rev. Neurosci. **7**(2), 153–160 (2006)
185. A. Matos, R. Suzuki, T. Arita, Heterochrony and artificial embryogeny: A method for analyzing artificial embryogenies based on developmental dynamics. Artif. Life **15**(2), 131–160 (2009)
186. C. Mattiussi, D. Floreano, Analog genetic encoding for the evolution of circuits and networks. IEEE Trans. Evol. Comput. **11**(5), 596–607 (2007)
187. J. McCormack, Aesthetic evolution of L-Systems revisited. in *Applications of Evolutionary Computing (Evoworkshops)* (2004), pp. 477–488
188. J. McCormack, *Impossible Nature: the Art of Jon McCormack*, Australian Centre for the Moving Image (2004)
189. W.S. McCulloch, W. Pitts, A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biophys. **5**(4), 114–133 (1943)
190. N.F. McPhee, E. Crane, S.E. Lahr, R. Poli, Developmental plasticity in linear genetic programming. in *conference on Genetic and Evolutionary Computation (GECCO)* (2009), pp. 1019–1026
191. T. Menezes, E. Costa, Artificial brains as networks of computational building blocks. in *European Conference on Complex Systems* (2008)
192. T. Menezes, E. Costa, The gridbrain: an heterogeneous network for open evolution in 3d environments. in *IEEE Symposium on Artificial Life* (2007), pp. 155–162
193. Y. Meng, Y. Zhang, Y. Jin, Autonomous self-reconfiguration of modular robots by evolving a hierarchical mechanochemical model. IEEE Comput. Intell. Mag. **6**(1), 43–54 (2011)
194. Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs* 1st–3rd edn. (Springer, New-York, 1992–1996)
195. Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, A. Lendasse, Op-elm: Optimally pruned extreme learning machine. IEEE Trans. Neural Netw. **21**(1), 158–162 (2010)
196. K.D. Micheva, B. Busse, N.C. Weiler, N. O'Rourke, S.J. Smith, Single-synapse analysis of a diverse synapse population: Proteomic imaging methods and markers. Neuron **68**(4), 639–653 (2004)
197. J.F. Miller, Evolving a self-repairing, self-regulating, french flag organism. in *Conference on Genetic and Evolutionary Computation (GECCO)* (Springer, 2004), pp. 129–139

198. J.F. Miller, Neuro-centric and holocentric approaches to the evolution of developmental neural networks. in ed. by Kowaliw et al. [160], pp. 242–268
199. J.F. Miller, W. Banzhaf, Evolving the program for a cell:fFrom french flags to boolean circuits. in *On Growth, Form and Computers* (2003), pp. 278–301
200. J.F. Miller, P. Thomson, A developmental method for growing graphs and circuits. in *Evolvable Systems: From Biology to Hardware* (2003), pp. 93–104
201. J.F. Miller, G.M. Khan, Where is the brain inside the brain? Memetic Comput. **3**, 217–228 (2011)
202. R. Milo, S. Itzkovitz, N. Kashtan, R. Levitt, S. Shen-Orr, I. Ayzenshtat, M. Sheffer, U. Alon, Superfamilies of evolved and designed networks. Science **303**(5663), 1538–1542 (2004)
203. A.A. Minai, D. Braha, Y. Bar-Yam, Complex engineered systems: Science meets technology. in ed. by D. Braha, Y. Bar-Yam, A.A. Minai *Complex Engineered Systems: Science Meets Technology, Chapter Complex Engineered Systems: A New Paradigm* (Springer, 2006), pp. 1–21
204. D.E. Moriarty, *Symbiotic Evolution of Neural Networks in Sequential Decision Tasks*, Ph.D. Thesis (University of Texas at Austin, USA, 1998)
205. J.-B. Mouret, S. Doncieux, B. Girard, Importing the computational neuroscience toolbox into neuro-evolution-application to basal ganglia. in *Conference on Genetic and Evolutionary Computation (GECCO)* (2010), pp. 587–595
206. J.-B. Mouret, P. Tonelli, Artificial evolution of plastic neural networks: a few key concepts. en ed. by Kowaliw et al. [160], pp. 269–280
207. J.-B. Mouret, S. Doncieux, MENNAG: a modular, regular and hierarchical encoding for neural-networks based on attribute grammars. Evol. Intell. **1**(3), 187–207 (2008)
208. T.D. Mrsic-Flogel, S.B. Hofer, K. Ohki, R.C. Reid, T. Bonhoeffer, M. Hbener, Homeostatic regulation of eye-specific responses in visual cortex during ocular dominance plasticity. Neuron **54**, 961–972 (2007)
209. P.L. Narasimha, W.H. Delashmit, M.T. Manry, J. Li, F. Maldonado, An integrated growing-pruning method for feedforward network training. Neurocomputing **71**(13–15), 2831–2847 (2008)
210. T. Natschläger, W. Maass, H. Markram, The "liquid computer": A novel strategy for real-time computing on time series. Spec Issue Found. Inf. Proc. TELEMATIK **8**, 39–43 (2002)
211. M.E.J. Newman, Modularity and community structure in networks. Proc. Natl. Acad. Sci. **103**(23), 8577–8582 (2006)
212. S.A. Newman, G. Forgacs, G.B. Müller, Before programs: the physical origination of multicellular forms. Int. J. Dev. Biol. **50**, 289–299 (2006)
213. S. Nichele, G. Tufte, Genome parameters as information to forecast emergent developmental behaviors. in ed. by J. Durand-Lose, N. Jonoska *Unconventional Computation and Natural Computation (UCNC)* (Springer, 2012), pp. 186–197
214. S. Nichele, G. Tufte, Trajectories and attractors as specification for the evolution of behaviour in cellular automata. in *IEEE Congress on Evolutionary Computation (CEC)* (2010), pp. 1–8
215. M. Nicolau, M. Schoenauer, W. Banzhaf, Evolving genes to balance a pole. in ed. by A. Esparcia-Alczar, A. Ekárt, S. Silva, S. Dignum, A. Uyar *Genetic Programming, Lecture Notes in Computer Science*, vol. 6021 (Springer, Berlin, 2010), pp. 196–207
216. A.B. Nielsen, L.K. Hansen, Structure learning by pruning in independent component analysis. Neurocomputing **71**(10–12), 2281–2290 (2008)
217. K. Nigam, A.K. Mccallum, S. Thrun, T. Mitchell, Text classification from labeled and unlabeled documents using EM. Mach. Learn. **39**(2–3), 103–134 (2000)
218. S. Nolfi, O. Miglino, D. Parisi, Phenotypic plasticity in evolving neural networks. in *From Perception to Action (PerAc)* (1994), pp. 146–157
219. S. Nolfi, D. Floreano, *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines* (MIT Press/Bradford Books, Cambridge, 2000)
220. D. Norton, D. Ventura, Improving liquid state machines through iterative renement of the reservoir. Neurocomputing **73**, 2893–2904 (2010)

221. D. Norton, D. Ventura, Preparing more effective liquid state machines using hebbian learning. in *International Joint Conference on Neural Networks (IJCNN)* (2006), pp. 8359–8364
222. B.A. Olshausen, D.J. Field, Sparse coding with an overcomplete basis set: a strategy employed by V1? Vis. Res. **37**(23), 3311–3325 (1997)
223. T.H. Oong, N.A.M.M. Isa, Adaptive evolutionary artificial neural networks for pattern classification. IEEE Trans. Neural Netw. **22**, 1823–1836 (2011)
224. C. Öztürkeri, M.S. Capcarrere, Self-repair ability of a toroidal and non-toroidal cellular developmental model. in *European conference on Advances in Artificial Life (ECAL)* (Springer, 2005), pp. 138–148
225. M.E. Palmer, Evolved neurogenesis and synaptogenesis for robotic control: the L-brain model. in *Conference on Genetic and Evolutionary Computation (GECCO)* (2011), pp. 1515–1522
226. H. Paugam-Moisy, R. Martinez, S. Bengio, Delay learning and polychronization for reservoir computing. Neurocomputing **71**(7–9), 1143–1158 (2008)
227. R. Perin, T.K. Berger, H. Markram, A synaptic organizing principle for cortical neuronal groups. Proc. Natl. Acad. Sci. **108**, 5419–5424 (2011)
228. R. Pfeifer, J. Bongard, *How the Body Shapes the Way We Think: A New View of Intelligence* (Bradford Books, 2006)
229. M. Pigliucci, Is evolvability evolvable? Nat. Rev. Genet. **9**, 75–82 (2008)
230. D.J. Price, A.P. Jarman, J.O. Mason, P.C. Kind, Building brains: an introduction to neural development. 2nd edn. (Wiley, 2009)
231. P. Prusinkiewicz, A. Lindenmayer, *The Algorithmic Beauty of Plants* (Springer, 1990)
232. W.J. Puma-Villanueva, E.P. dos Santos, F.J. Von Zuben, A constructive algorithm to synthesize arbitrarily connected feedforward neural networks. Neurocomputing **75**(1), 14–32 (2012)
233. Z.W. Pylyshyn, Is vision continuous with cognition? the case for cognitive impenetrability of visual perception. Behav. Brain Sci. **22**, 341–423 (1999)
234. S.R. Quartz, T.J. Sejnowski, H. Hughes, The neural basis of cognitive development: a constructivist manifesto. Behav. Brain Sci. **20**, 537–596 (1997)
235. R. Raina, A. Battle, H. Lee, B. Packer, A.Y. Ng, Self-taught learning: transfer learning from unlabeled data. in *ICML '07: Proceedings of the 24th International Conference on Machine Learning* (ACM, New York, 2007), pp. 759–766
236. S. Rebecchi, H. Paugam-Moisy, M. Sebag, Learning sparse features with an auto-associator. in ed. by Kowaliw et al. [160], pp. 144–165
237. T. Reil, Dynamics of gene expression in an artificial genome—implications for biological and artificial ontogeny. in *Proceedings of the 5th European Conference on Artificial Life (ECAL99), Number 1674 in Lecture Notes in Artificial Intelligence* (1999), pp. 457–466
238. J. Reisinger, R. Miikkulainen, Acquiring evolvability through adaptive representations. in *8th Conference on Genetic and Evolutionary Computation (GECCO)* (2007), pp. 1045–1052
239. J. Reisinger, R. Miikkulainen, Selecting for evolvable representations. in *7th Conference on Genetic and Evolutionary Computation (GECCO)* (2006), pp. 1297–1304
240. J. Rieffel, D. Knox, S. Smith, B. Trimmer, Growing and evolving soft robots. Artif. Life 1–20 (2012)
241. J. Rieffel, J. Pollack, The emergence of ontogenic scaffolding in a stochastic development environment. in ed. by K. Deb *Conference on Genetic and Evolutionary Computation (GECCO) of Lecture Notes in Computer Science*, vol. 3102 (Springer, 2004), pp. 804–815
242. B. Roeschies, C. Igel, Structure optimization of reservoir networks. Logic J. IGPL **18**(5), 635–669 (2010)
243. D. Roggen, D. Federici, Multi-cellular development: is there scalability and robustness to gain? in *Parallel Problem Solving from Nature (PPSN)* (2004), pp. 391–400
244. D. Roggen, D. Federici, D. Floreano, Evolutionary morphogenesis for multi-cellular systems. Genet. Program. Evol. Mach. **8**(1), 61–96 (2006)
245. D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors. Nature **323**, 533–536 (1986)
246. R. Salakhutdinov, A. Mnih, G. Hinton, Restricted Boltzmann machines for collaborative filtering. in *Proceedings of the 24th International Conference on Machine Learning, ICML '07* (ACM, New York, 2007), pp. 791–798

247. K. Sano, H. Sayama, Wriggraph: a kinetic graph model that uniformly describes ontogeny and motility of artificial creatures. in *Artificial life X: proceedings of the Tenth International Conference on the Simulation and Synthesis of Living Systems*, vol. 10 (MIT Press, 2006), p. 77
248. L. Schramm, Y. Jin, B. Sendhoff, Redundancy creates opportunity in developmental representations. in *IEEE Symposium on Artificial Life (IEEE-ALIFE)*(2011)
249. L. Schramm, B. Sendhoff, An animat's cell doctrine. in *European Conference on Artificial Life (ECAL)* (MIT Press, 2011), pp. 739–746
250. B. Schrauwen, M. Wardermann, D. Verstraeten, J.J. Steil, D. Stroobandt, Improving reservoirs using intrinsic plasticity. Neurocomputing **71**(7–9), 1159–1171 (2008)
251. E.K. Scott, L.L. Luo, How do dendrites take their shape? Nat. Neurosci. **4**(4), 359–365 (2001)
252. S.I. Sen, A.M. Day, Modelling trees and their interaction with the environment: A survey. Comput. Graph. **29**(5), 805–817 (2005)
253. B. Sendhoff, E. Körner, O. Sporns, Creating brain-like intelligence. in ed. by Sendhoff et al. [255], pp. 1–14
254. B. Sendhoff, E. Körner, O. Sporns, H. Ritter, K. Doya (eds.), *Creating Brain-Like Intelligence* vol. 5436 (Springer, 2009)
255. S.H. Seung, Neuroscience: towards functional connectomics. Nature **471**(7337), 170–172 (2011)
256. C.W. Seys, R.D. Beers, Genotype reuse more important than genotype size in evolvability of embodied neural networks. in *9th European Conference on Advances in Artificial Life (ECAL)* (2007), pp. 915–924
257. S.K. Sharma, P. Chandra, An adaptive slope sigmoidal function cascading neural networks algorithm. in *2010 3rd International Conference on Emerging Trends in Engineering and Technology (ICETET)* (2010), pp. 531–536
258. A.A. Siddiqi, S.M. Lucas, Comparison of matrix rewriting versus direct encoding for evolving neural networks. in *IEEE International Conference on Evolutionary Computation, ICEC'98* (1998), pp. 392–397
259. M.S.M. Siddiqui, B. Bhaumik, Reaction-diffusion based model to develop binocular simple cells in visual cortex along with cortical maps. in *International Joint Conference on Neural Networks (IJCNN)* (2010), pp. 1–8
260. J. Šíma, P. Orponen, General-purpose computation with neural networks: a survey of complexity theoretic results. Neural Comput. **15**(12), 2727–2778 (2003)
261. K. Sims, Evolving virtual creatures. in *Proceedings of SIGGRAPH* (1994), pp. 15–22
262. A. Soltoggio, P. Durr, C. Mattiussi, D. Floreano, Evolving neuromodulatory topologies for reinforcement learning-like problems. *IEEE Congress on Evolutionary Computation (CEC)* (2007), pp. 2471–2478
263. O. Sporns, From complex networks to intelligent systems. in ed. by Sendhoff et al. [255], pp. 15–30
264. K.O. Stanley, R. Miikkulainen, Evolving neural networks through augmenting topologies. Evol. Comput. **10**(2), 99–127 (2002)
265. K.O. Stanley, R. Miikkulainen, A taxonomy for artificial embryogeny. Artif. Life **9**(2), 93–130 (2003)
266. K.O. Stanley, D.B. D'Ambrosio, J. Gauci, A hypercube-based encoding for evolving large-scale neural networks. Artif. Life **15**(2), 185–212 (2009)
267. T. Steiner, Y. Jin, B. Sendhoff, Vector field embryogeny. PLoS ONE **4**(12), e8177 (2009)
268. G.F. Striedter, *Principles of Brain Evolution* (Sinauer Associates, Sunderland, 2005)
269. J.L. Subirats, L. Franco, J.M. Jerez, C-mantec: a novel constructive neural network algorithm incorporating competition between neurons. Neural Netw. **26**, 130–140 (2012)
270. M. Suchorzewski, J. Clune, A novel generative encoding for evolving modular, regular and scalable networks. in *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation - GECCO '11* (2011), pp. 1523–2531
271. R.S. Sutton, A.G. Barto, *Reinforcement Learning: An Introduction* (MIT Press, 1998)

272. H. Tanaka, L.T. Landmesser, Cell death of lumbosacral motoneurons in chick, quail, and chick-quail chimera embryos: a test of the quantitative matching hypothesis of neuronal cell death. J. Neurosci. **6**(10), 2889–2899 (1986)
273. M.E. Taylor, S. Whiteson, P. Stone, Temporal difference and policy search methods for reinforcement learning: an empirical comparison. in *Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI-07)* (2007)
274. G. Tesauro, Practical issues in temporal difference learning. Mach. Learn. **8**(3), 257–277 (1992)
275. R. Thenius, M. Dauschanand, T. Schmickl, K. Crailsheim, Regenerative abilities in modular robots using virtual embryogenesis. in *International Conference on Adaptive and Intelligent Systems (ICAIS)* (2011), pp. 227–237
276. P. Tonelli, J.-B. Mouret, On the relationships between synaptic plasticity and generative systems. in *Conference on Genetic and Evolutionary Computation (GECCO)* (2011)
277. P. Tonelli, J.B. Mouret, On the relationshipd between generative encodings, regularity, and learning abilities when encoding plastic artificial neural networks. PLoS One **8**(11), e79138 (2013)
278. T. Trappenberg, *Fundamentals of Computational Neuroscience* 2nd edn. (Oxford University Press, Oxford, 2009)
279. T. Trappenberg. A brief introduction to probabilistic machine learning and its relation to neuroscience. in ed. by Kowaliw et al. [160], pp. 62–110
280. G. Tufte, P.C. Haddow, Extending artificial development: exploiting environmental information for the achievement of phenotypic plasticity. in *Conference on Evolvable Systems: from Biology to Hardware (ICES)* (Springer, 2007), pp. 297–308
281. A. Turing, The chemical basis of morphogenesis. Philosop. Trans. Roy. Soc. B **237**, 37–72 (1952)
282. M. Ulieru, R. Doursat, Emergent engineering: a radical paradigm shift. Int. J. Auton. Adap. Commun. Syst. **4**(1), 39–60 (2011)
283. V. Valsalam, J.A. Bednar, R. Miikkulainen, Developing complex systems using evolved pattern generators. IEEE Trans. Evol. Comput. **11**(2), 181–198 (2007)
284. P. Verbancsics, K.O. Stanley, Constraining connectivity to encourage modularity in HyperNEAT. in *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation (GECCO)* (ACM Press, New York, 2011), pp. 1483–1490
285. P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res. (2010)
286. C. von der Malsburg, Synaptic plasticity as basis of brain organization. in ed. by J.P. Changeux, M. Konishi *The Neural and Molecular Bases of Learning* (Wiley, 1987), pp. 411–432
287. C. von der Malsburg, The correlation theory of brain function. in *Models of Neural Networks II: Temporal Aspects of Coding and Information Processing in Biological Systems* (Springer, 1981), pp. 95–119
288. C. von der Malsburg, E. Bienenstock, Statistical coding and short-term synaptic plasticity. in *Disordered Systems and Biological Organization* (Springer, 1986), pp. 247–272
289. G.P. Wagner, M. Pavlicev, J.M. Cheverud, The road to modularity. Nat. Rev. Genet. **8**(12), 921–931 (2007)
290. V.J. Wedeen, D.L. Rosene, R. Wang, G. Dai, F. Mortazavi, P. Hagmann, J.H. Kaas, W.-Y.I. Tseng, The geometric structure of the brain fiber pathways. Science **335**(6076), 1628–1634 (2012)
291. J. Weng, J. McClelland, A. Pentland, O. Sporns, I. Stockman, M. Sur, E. Thelen, Autonomous mental development by robots and animals. Science **291**(5504), 599–600 (2001)
292. J. Weng, A computational introduction to the biological brain-mind. Nat. Intell. INNS Mag. **1**(3), 5–16 (2012)
293. D.J. Willshaw, C. von der Malsburg, How patterned neural connections can be set up by self-organization. Proc. Roy. Soc. London Ser. B Biol. Sci. **194**(1117), 431–445 (1976)
294. L. Wolpert, *Developmental Biology* (Oxford University Press, Oxford, 2011)

295. L. Wolpert, Positional information and the spatial pattern of cellular differentiation. J. Theor. Biol. **1**, 1–47 (1969)
296. B. Wróbel, A. Abdelmotaleb, M. Joachimczak, Evolving spiking neural networks in the GReaNs (gene regulatory evolving artificial networks) plaftorm. in *EvoNet2012: Evolving Networks, from Systems/Synthetic Biology to Computational Neuroscience Workshop at Artificial Life XIII* (2012), pp. 19–22
297. B. Wróbel, M. Joachimczak, Using the GReaNs (genetic regulatory evolving artificial networks) platform for signal processing, animat control, and artificial multicellular development. in ed. by Kowaliw et al. [160], pp. 198–214
298. H. Yamada, T. Nakagaki, R.E. Baker, P.K. Maini, Dispersion relation in oscillatory reaction-diffusion systems with self-consistent flow in true slime mold. J. Math. Biol. **54**(6), 745–760 (2007)
299. S.-H. Yang, Y.-P. Chen, An evolutionary constructive and pruning algorithm for artificial neural networks and its prediction applications. Neurocomputing **86**, 140–149 (2012)
300. Yann LeCun, J.S. Denker, S. Solla, R.E. Howard, L.D. Jackel, Optimal brain damage. in ed. by D. Touretzky *NIPS'89* (Morgan Kaufman, 1990)
301. X. Yao, Y. Liu, A new evolutionary system for evolving artificial neural networks. IEEE Trans. Neural Netw. **8**, 694–713 (1997)
302. X. Yao, Evolving neural networks. Proc. IEEE **87**(9), 1423–1447 (1999)
303. I.B. Yildiz, H. Jaeger, S.J. Kiebel, Re-visiting the echo state property. Neural Netw. **35**, 1–9 (2012)
304. J. Yin, Y. Meng, Y. Jin, A developmental approach to structural self-organization in reservoir computing. IEEE Trans. Auton. Ment. Dev. **4**(4), 273–289 (2012)
305. T. Yu, J. Miller, Neutrality and the evolvability of boolean function landscape. in ed. by J. Miller, M. Tomassini, P.L. Lanzi, C. Ryan, A. Tettamanz, W.B. Langdon *Genetic Programming* (Springer, 2001), pp. 204–217
306. C. Yu, M.T. Manry, J. Li, An efficient hidden layer training method for multilayer perceptron. Neurocomputing **70**(1–3), 525–535 (2006)
307. B. Zhang, D.J. Miller, Y. Wang, Nonlinear system modelling with random matrices: echo state networks revisited. IEEE Trans. Neural Netw. Learn. Syst. **23**(1), 175–182 (2012)
308. R. Zhang, Y. Lan, G.-B. Huang, Z.-B. Xu, Universal approximation of extreme learning machine with adaptive growth of hidden nodes. IEEE Trans. Neural Netw. Learn. Syst. **23**(2), 365–371 (2012)
309. P. Zheng, C. Dimitrakakis, J. Triesch, Network self-organization explains the distribution of synaptic efficacies in neocortex. in ed. by Kowaliw et al. [159], pp. 8–9
310. N.E. Ziv, C.C. Garner, Principles of glutamatergic synapse formation: seeing the forest for the trees. Current Opin. Neurobiol. **11**(5), 536–543 (2001)
311. F. Zubler, A. Hauri, S. Pfister, A.M. Whatley, M. Cook, R. Douglas, An instruction language for self-construction in the context of neural networks. Front. Comput. Neurosci. **5**(57), 1–15 (2001)