and Genetic Algorithms

Synthesis and Simulation of Living Systems

Rasmussen

Networks: COGANN-92

July 26-29, 1992 in Vail, Colorado

their Applications

their Applications

Nature, Brussels, Belgium

their Applications

Rasmussen animat

Genetic Algorithms

Genetic Algorithms

Networks and Neural Networks

Genetic Algorithms

Genetic Algorithm

Algorithms

approach to neural modeling based on the idea of searching, with learning methods, for a synaptic learning rule which is biologically plausible and yields networks that are able to learn to perform difficult tasks. The proposed method of automatically finding the learning rule relies on the idea of considering the synaptic modification rule as a parametric function. This function has local inputs and is the same in many neurons. The parameters that define this function can be estimated with known learning methods. For this optimization, particular attention is given to gradient descent and genetic algorithms. In both cases, estimation of this function consists of a joint global optimization of the synaptic modification function and the networks that are learning to perform some tasks. Both network architecture and the learning function can be designed within constraints derived from biological knowledge.

Networks

School

Implementation of a Boltzmann Machine and Associated Robotic Experimentations Conference on Artificial Life by natural life mechanisms to an artificial life creature, namely a small mobile robot, called KitBorg. Probabilistic inference suggests that any cognitive problem may be split in two optimization problems. The first one called the dynamic inference problem is an abstraction of learning, the second one, namely, the static inference problem, being a mathematical metaphor of pattern association. Other optimization technics should be considered in that context and especially genetic algorithms. The purpose of this paper is to describe the state of the art of the investigations which the Author is "akin"about that question using a parallel genetic algorithm. The author first "ecall"the principles of probabilistic inference, then he presents briefly the parallel genetic algorithm and the ways it is used to deal with both optimization problems, to finally conclude about ongoing robotic experimentations and future planned extensions.

Prediction and Genetic Algorithms

Algorithms

Algorithms: A Learning Rule for Temporal Patterns neural networks with an arbitrary structure is presented. It is suited for the learning of temporal patterns and leads to stable neural networks with feedback.

l'Institut Electrotechnique Montefiore industrial problems which caused trouble to classical optimization techniques (they were usually trapped into local minima), without positive results. One of the reasons was that the solutions among the population were too close to one another too early in the search process. Another was the unsuitability of the operators employed to create new solutions for the neural network optimization problem. Attempts at application to control problems, where backpropagation could not be used, yielded disappointing results except for very simple problems such as the inverted pendulum. An explanation of these findings is suggested.

and Genetic Algorithms

name when presented with a sample from one of several classes. In forming the sample to present to the classifier, there may be a large number of measurements one can make. Feature selection addresses the problem of determining which of these measurements are the most useful for determining the pattern's class. In this paper, we describe experiments using a genetic algorithm for feature selection in the context of neural network classifiers, specifically, counterpropagation networks. We present two novel techniques in our application of genetic algorithms. First, we configure our genetic algorithm to use an approximate evaluation in order to reduce significantly the computation required. In particular, though our desired classifiers are counterpropagation networks, we use a nearest-neighbor classifier to evaluate feature sets. We show that the features selected by this method are effective in the context of counterpropagation networks. Second, we propose a method we call training set sampling, in which only a portion of the training set is used on any given evaluation. Again, significant computational savings can be made by using this method, i.e., evaluations can be made over an order of magnitude faster. This method selects feature sets that are as good as and occasionally better for counterpropagation than those chosen by an evaluation that uses the entire training set.

Neurocomputers.

Recognition, using Genetic Algorithms

search.

Algorithms search.

Genetic Algorithm batch processes. The consideration of the dynamics of batch processes, a cascade neural network which is the combination of BPN and Euler's numerical integration method, is successfully used to model of batch processes. In terms of this neural network model, an extended genetic algorithm is adopted to generate the optimal trajectory for improving the desired process performance. The genetic algorithm is a general methodology for searching a solution space in a manner analogous to the natural selection procedure in biological evolution. With the motivation of modern genetic techonology, the rule-inducer genetic algorithm is proposed for dynamic optimization of batch processes. The simulation study of a typical biochemical process shows this neural network modeling technique has a good generalization of the batch process and the extended real-value genetic algorithm has a good capability to solve the complicated dynamical optimization problems.

Networks (GAs) for multilayered networks is described. The present method does not require that the input-output pairs for each layer to be known "a priori since all modules are trained concurrently. For an N-module system, N separate pools of chromosomes are maintained and updated. The algorithm is tested using the 4-bit parity problem and a classification problem. Experiment results are presented and discussed.

Neural Networks

Algorithm. Algorithms and Neural Networks

classification by mapping input patterns into one of several categories. Rather than being specifically programmed, backpropagation networks (BPNs) 'learn' this mapping by exposure to a training set, a collection of input pattern samples matched with their corresponding output classification. The proper construction of this training set is crucial to successful training of a BPN. One of the criteria to be met for proper construction of a training set is that each of the classes must be adequately represented. A class that is represented less often in the training data may not be learned as completely or correctly, impairing the network's discrimination ability. The degree of impairment is a function of (among other factors) the relative number of samples of each class used for training. The paper addresses the problem of unequal representation in training sets by proposing two alternative methods of learning. One adjusts the learning rate for each class to achieve user-specified goals. The other utilizes a genetic algorithm to set the connection weights with a fitness function based on these same goals. These methods are tested using both artificial and real-world training data.

data points due to errors in the measurement of the data. One such instance of measuring devices recording anomalies occurs in the crash testing of vehicles. Force and acceleration data is collected which an engineer inspects for anomalies, correcting those that are found. Artificial Neural Network (ANN) technology was successfully applied to this problem to eliminate the cost and delay of this manual process. The Author employed "machine learning algorithm that simulates the Darwinian concept of survival of the fittest known as the Genetic Learning Algorithm (GLA). By combining the strength of the GLA and ANNs, a network architecture was created that optimized the size, speed, and accuracy of the ANN. This hybridized system also used the GLA to determine the smallest number of inputs into the ANN that were necessary to detect anomalies in data. This algorithm is known as GENENET, and is described in the paper.

Implemented on Multiple Transputers

of Structure

performance for neural network weight optimization are really genetic hill-climbers, with a strong reliance on mutation rather than hyperplane sampling. Neural control problems are more appropriate for these genetic hill-climbers than supervised learning applications because in reinforcement learning applications gradient information is not directly available. Genetic reinforcement learning produces competitive results with AHC, another reinforcement learning paradigm for neural networks that employs temporal difference methods. The genetic hill-climbing algorithm appears to be robust over a wide range of learning conditions.

Networks

Explanation Facilities Neural Networks

artificial neuron which can be used to process and classify dynamic signals is described. The electrical circuit architecture is modeled after complex neurons in the vertebrate brain which have spatially extensive dendritic tree structures that support large numbers of synapses. The circuit is primarily analog and, as in the biological model system, is virtually immune to process variations and other factors which often plague more conventional circuits. The nonlinear circuit is sensitive to both temporal and spatial signal characteristics but does not make use of the conventional neural network concept of weights, and as such does not use multipliers, adders, look-up-tables, microprocessors or other complex computational devices. The Author shows "ha"artificial neural networks with passive dendritic tree structures can be trained, using a specialized genetic algorithm, to produce control signals useful for target tracking and other dynamic signal processing applications.

Network Neural Networks

Algorithms

Neural-Network Driven Robot

Playing Tic-Tac-Toe

Playing Tic-Tac-Toe

Programming

Supervised Learning

Finite-State Behaviour Conference on Artificial Life proposed, which is loosely based on the marker structure of biological DNA. The mechanism allows all aspects of the network structure, including the number of nodes and their connectivity, to be evolved through genetic algorithms. The effectiveness of the encoding scheme is demonstrated in an object recognition task that requires artificial creatures (whose behavior is driven by a neural network) to develop high-level finite-state exploration and discrimination strategies. The task requires solving the sensory-motor grounding problem, i.e., developing a functional understanding of the effects that a creature's movement has on its sensory input.

on Simuation of Adaptive Behaviour (SAB 92)

Values in Neural Networks for Attitude Control Systems

Pair of Sticks to Walk

Network Modules.

which Teaches a Pair of Stick Legs to Walk

Qualitatively New Behaviors in Recurrent Neural Networks convergent networks (until the recent rise of 'recurrent backpropagation' algorithms [e.g. WILLIAMS & ZIPSER 1989ab]) has not been unreasonable. Relatively little analytical work had been done on neural networks whose inputs and/or outputs are time-dependent, hence few guidelines existed on how to train such networks. Consequently, research concentrated on more restrictive 'static' neural nets such as 'feedforward' (Backprop) [RUMELHART & McCLELLAND 1986] and "Hopfield"(clamped inputs, convergent outputs) [HOPFIELD 1982]. This emphasis on convergence was unfortunate, because the true richness of neural network dynamics is to be found when inputs and/or outputs are time-dependent. This paper shows that Genetic Programming techniques (i.e. using Genetic Algorithms to build/evolve complex systems) can be applied successfully to training nonconvergent networks, and presents some examples of their extraordinary behavioral versatility. This paper terminates by comparing GenNet behaviors with those generated by the new 'recurrent backpropagation' algorithms [WILLIAMS & ZIPSER 1989ab]. It is claimed that the GenNet behaviors are a lot more flexible and interesting because they do not require the training process to be "closely supervised".

Conference on Artificial Life evolution, or building complex systems using the genetic algorithm) can be used to evolve dynamic behaviors in neural systems which are controllable or steerable. The genetic algorithm evolves the weights of a fully-connnected time-dependent neural network (called a GenNet), such that the same GenNet is capable of generating two separate time-dependent behaviors, depending upon the setting of two different values of a clamped input control variable. By freezing these weights in the GenNet and then applying intermediate control values, one obtains intermediate behaviors, showing that the GenNet has generalized its behavioral learning. It has become controllable or steerable. This principle is applicable to the evolution of many controllable neural behaviors and is useful in the construction of artificial creatures (with artificial nervous systems) based on neural modules. One simply evolves two behaviors at different settings of the control input so that the GenNet generalizes its behavioral learning. In this paper, a concrete example of this process is given in the form of the genetic programming of a variable frequency generator GenNet. This paper ends with a discussion on the handcrafters vs. evolutionists controversy, concerning future approaches to artificial creature (biot) building.

Nervous Systems

Steps Evolution of neural networks, which is defined to be the art of evolving neural networks in incremental steps, using Genetic Algorithms. One evolves the weight values of a fully connected neural network (called a GenNet [de Garis 1990, 1993]) containing N neurons to perform T tasks, and then takes the result (i.e. the evolved weights of the N neurons) and adds a few more neurons dN, to evolve the performance of a few more tasks dT. This paper investigates (a) whether this can be done at all, (b) whether is is faster to evolve an N + dN GenNet performing T + dT tasks from scratch or to do it incrementally (1.e. [N,T] then [N+dN,T+dT]), and (c) how the two approaches (i.e. from scratch or incremental) compare in task performance quality. Incremental Evolution will become an important issue when the various brain builder groups around the world (i.e.groups using evolved neural network modules to build artificial nervous systems for biological robots (biots), e.g. Beer's group at Case Western Reserve University USA, Cliff et al's group at Sussex University UK, and the Author's group ATR Japan [de Garis 1993] are confronted with the decision whether to start from scratch when desiring to evolve biots with a greater number of behaviors, or to increment their already evolved nervous systems. Nature obviously had to increment.

been somewhat one directional. In most cases a genetic algorithm has been used to generate better neural networks. In this paper we combine the use of genetics algorithms and neural networks, but from a conceptually different point of view. We show that it is possible to use a genetics algorithm as the learning algorithm for a neural network. In our model the neural network has a fixed architecture and processes binary strings using genetic operators.

Developmental Process Algorithms and Neural Networks

PhD94-01-E.ps.Z (english) PhD94-01-F.ps.Z (french)

Evolution

Alternatives networks into a set of labeled trees. Such sets of trees can be evolved by the genetic algorithm so as to find a particular set of trees that encodes a family of Boolean neural networks for computing a family of Boolean functions. Cellular encoding is presented as a graph grammar. A method is proposed for translating a cellular encoding into a set of graph grammar rewriting rules of the kind used in the Berlin algebraic approach to graph rewriting. The genetic search of neural networks via cellular encoding appears as a grammatical inference process where the language to parse is implicitly specified, instead of explicitly by positive and negative examples. Experimental results shows that the genetic algorithm can infer grammars that derive neural networks for the parity, symmetry and decoder Boolean function of arbitrary large size.

Networks

and Neural Networks

Brighton

Operators for Neural Net Structure Specification

design using genetic algorithms. A genetic algorithm is a robust optimization method particularly well suited for search spaces that are high-dimensional, discontinuous and noisy-features that typify the neural network design problem. Our approach is relevant to virtually all neural network applications: it is network-model independent and it permits optimization for arbitrary, user-defined criteria. We have developed an experimental system, NeuroGENESYS, and have conducted several experiments on small-scale problems. Performance improvements over manual designs have been observed, the interplay between performance criteria and network design aspects has been demonstrated, and general design principles have been uncovered.

Activations based on genetic search, in hidden target space, and gradient descent learning strategies. Our simulations show that the new algorithm combines the global optimization capabilities of genetic algorithms with the speed of gradient descent local search in order to outperform pure descent-based algorithms such as backpropagation. In addition, we show that genetic search in hidden target space is less complex than that of weight space.

(GAs). GAs are compared with other standard optimization methods like gradient descent or simulated annealing (SA). It is shown that SA is just a special case of GA. The role of a population in the optimization process is demonstrated by an example. GA was applied as a learning algorithm to neural networks.

Presented at the Conference on Parallel Computing: Achievements, Problems and Prospects interacting, distributed artificial agents on distributed-memory, MIMD computers is presented. Interactions among aggregates of intelligent agents in an organization are restricted to obey competition and cooperation criteria. Each intelligent agent in an organization is a parallel implementation of a feedforward multilayer perceptrons neural network using error backpropagation (BP) as the learning rule. In this preliminary study, domination, viewed as a type of deterministic genetic algorithm (GA,) is chosen to be the preferred form of interaction. The framework exploits the hierarchical nature intrinsic in an organizational approach to problem-solving. It takes advantage of parallelism at different levels of granularity, from domain decomposition within the agents to coarse grain team-level interaction. Transputer-based simulation results for a test problem of learning the solution to a parity function of predicate order 10 is discussed.

apparently complex task of controlling walking in a real quadrupedal robot with highly nonlinear interactions between the control elements can be learned quickly by a crude and simple reinforcement learning algorithm. They can as yet say little that is useful about the contribution of reflexes to learned walking, and nothing about the quality of evolved solutions other than that their discovery by applying genetic algorithms to real robots is likely to take a prohibitively long time. However, they hope that their experiences will point the way to more controlled studies of the applications of reinforcement learning to real-world problems, especially to control problems associated with autonomous mobile robots.

Reweighting of its Links

Perceptual Learning in Connectionist Networks Intelligence

Networks

Algorithms and Neural Networks

Classification

Control

System

Networks networks in which several summing units are replaced by units capable of calculating a weighted product of inputs. While such networks can be trained using traditional backpropagation, the solution involves the manipulation of complex-valued expressions. As an alternative, this paper investigates the training of product networks using genetic algorithms. Results are presented on the training of a neural network to calculate the optimum width of transistors in a CMOS switch given desired operating parameters. It is shown how local minima affect the performance of the genetic algorithm, and one method of overcoming this is presented.

Transputer-Based Genetic Algorithms

Parameters

unknown, potentially nonlinear systems is a difficult, fundamental problem in machine learning and engineering control. In this paper, a *genetic algorithm* (GA) based technique is used to iteratively form polynomial networks that model the behavior of nonlinear systems. This approach is motivated by the *group method of data handling* (GMDH) (Ivakhnenko, 1971), but attempts to overcome the computational overhead and locality associated with the original GMDH. The approach presented here uses a multi-modal GA (Deb, 1989) to select nodes for a network based on an information-theoretic fitness measure. Preliminary results show that the GA is successful in modeling continuous-time and discrete-time chaotic systems. Implications and extensions of this work are discussed.

Optimization by Micro-Genetic Algorithms and optimization is investigated. A recurrent neural network is trained on a set of fermentation data, and thereafter used as a nonlinear process model to estimate nonmeasurable process states at different conditions. With the bioprocess state variable information available, an optimization technique can be used to generate optimum controls settings to improve the process performance. This paper explores the use of Micro-Genetic Algorithms as a technique for bioreactor optimization. Simulation results will be discussed based in the fermentative ethanol production by the anaerobic bacteria Zymomonas mobilis.

Algorithms and Neural Networks

Learning Trials Affect the Rate of Evolution

Using Genetic Algorithms (AAAI-90)

System

Networks using Genetic Algorithms

algorithms use a discrete generation model in which all individuals in a population synchronize mating period. The discrete generation model, however, wastes processor time in parallel implementations when the fitness of each individual (proportionally or reversely) correlates with the computational cost of its evaluation. An example of such a task is neural network design and training. In some cases, over 80been wasted. The continuous generation model mitigates this problem by introducing asynchronous mating, the continuous generation model increases the number of reproduction per a unit-time over 500discrete model. CPU idle time has been minimized to 1/25. Also, a significant improvement in convergence speed has been estimated.

Geographical Isolation Model a genetic-algorithm. The neural network is a perceptron, which has three outputs; the logical AND, OR and XOR of two inputs The evaluation function for optimization is a linear combination of the correctness, the network sizes, and an auxiliary term inducing the optimum solution The chromosome is a vector of the link weights of the network. The genetic operators used are crossing-over and point-mutation on the parent chromosomes Two genetic rules were tested. In the haploidy rule, each individual has single chromosome, and the offspring is generated by crossing-over the parents' chromosomes at a randomly chosen locus and taking one of those crossed-over chromosomes. In the diploidy rule, each individual has a pair of chromosomes, and the offspring's chromosomes are generated by combining the gamete produced through the meiosis of the parents' chromosomes. The other model used in the genetic algorithm is the geographical isolation model, where the entire population is divided into four sub-populations, in which the local selection and reproduction are carried out, though, in some time interval, randomly sampled individuals are exchanged among sub-populations. Comparison was made among four combinations of haploid or diploid, and single-population or multiple sub-populations. Diploidy together with the sub-population model was proved to be the best for this optimization problem. Thus, the optimum structure of network was found.

Selection

Algorithms and Neural Networks

Architectures

have been used to solve the travelling salesman problem. These networks usually require a set of parameters to be carefully selected and tuned to produce sensible solutions. Genetic Algorithms are basically adaptive systems that transform a population of individuals into new populations, using relatively simple mechanisms. It has the ability to efficiently explore the problem sub-space to produce approximate solutions that are globally competitive. This paper will show how Genetic Algorithms may be used in conjunction with the Hopfield/Tank neural net by breeding an effective set of control parameters in the parameter sub-space to be used by the artificial neural network.

Neuronaler Netze

Control of a Walking Robot pattern generator (MPG) for the control of a walking robot. The experiments were carried out on a six-legged, Brooks-style insect robot. The MPG was composed of a network of neurons with weights determined by genetic algorithm optimization. Staged evolution was used to improve the convergence rate of the algorithm. First, an oscillator for the individual leg movements was evolved. Then, a network of these oscillators was evolved to coordinate the movements of the different legs. By introducing a staged set of manageable challenges, the algorithm's performance was improved.

Algorithms and Neural Networks

adaptive systems. Although in the worst case there may be no hope of effective adaptation, not all forms of environmental variability need be so disabling. We consider a broad class of non-stationary environments, those which combine a variable *result function* with an invariant *utility function*, and demonstrate via simulation that an adaptive strategy employing both evolution and learning can tolerate a much higher rate of environmental variation than an evolution-only strategy. We suggest that in many cases where stability has previously been assumed, the constant utility non-stationary environment may in fact be a more robust description.

Neural Networks

genetic algorithms, producing a flexible and powerful learning paradigm, called evolutive learning.

Evolutive learning combines as complementary tools both inductive learning through synaptic weight adjustment and deductive learning through the modification of the network topology to obtain the automatic adaptation of system knowledge to the problem domain environment. Algorithms for the development of an evolutive learning machine are presented. A fuzzy criterion based on entropy is proposed to select the architecture for a fuzzy neural network best suited to a specific problem domain.

learning rule of neural networks. The genetic algorithm consists of four operations: selection; reproduction; crossover; and mutation. They look into the learning efficiency of two kinds of the crossover for the unsupervised learning rule. Moreover, they investigate the learning rate with respect to the mutation rate.

International Conference at Innsbruck, Austria

Feedforward Neural Networks.

Algorithms

Representation a given system in order to perform optimization functions. The actual structural layout of this representation, called a genome, has a crucial impact on the outcome of the optimization process. The purpose of this paper is to study the effects of different internal representations in a GA, which generates neural networks. A second GA was used to optimize the genome structure. This structure produces an optimized system within a shorter time interval.

within the neural networks field. After generating a robust GA engine, the system was used to generate neural network circuit architectures. This was accomplished by using the GA to determine the weights in a fully interconnected network. The importance of the internal genetic representation was shown by testing different approaches. The effects in speed of optimization of varying the constraints imposed upon the desired network were also studied. It was observed that relatively loose constraints provided results comparable to a fully constrained system. The typeof neural network circuits generated were recurrent competitive fields as described by Grossberg (1982).

technique, evolutionary programming, for developing self-organizing neural networks. The chosen stochastic search method is capable of simultaneously evolving both network architecture and weights. The number of synapses and neurons are incorporated into an objective function so that network parameter optimization is done with respect to computational costs as well as mean pattern error. Experiments are conducted using feedforward networks for simple binary mapping problems.

a stochastic search technique, for simultaneously determining the weights and the number of hidden units in a fully-connected, multi-layer neural network. The simulated evolution search paradigm provides a means for optimizing both network structure and weight coefficients. Orthogonal learning is implemented by independently modifying network structure and weight parameters. Different structural level search strategies are investigated by comparing the training processes for the 3-bit parity problem. The results indicate that evolutionary programming provides a robust framework for evolving neural networks.

Neural Networks

Optimization of Neural Networks complex functions over multidimensional domains, such as the space of the connection weights in a neural network. A feed-forward layered network is used to simulate the life cycle of a synthetic animal that moves in an environment and captures food objects. The adaptation of the animal (i.e. of the network's weight matrix) to the environment can be measured by the amount of reached food objects in a given lifetime. The Authors consider "hi"amount as a fitness function to be optimized by a genetic algorithm over the space of the connection weights. The network can learn the weights that solve the survival task only by means of its genetic evolution. The recombination genetic operator (crossover) can be seen as a model of sexual recombination for the population, while mutation models agamic reproduction. The central problem in trying to apply crossover is the difficult mapping between the genetic code string (genotype) and the network's weight matrix (phenotype). For this reason crossover has been considered unsuitable for this kind of problem in the past. The Authors propose "simple mapping and compare the effects of sexual versus agamic reproduction in such a problem. The results of several parametric simulations are outlined, showing that crossover actually helps to speed up the genetic learning.

Random Drift? Conference on Artificial Life biology: most high organisms use some form of sexual recombination of the genetic material in the process of reproduction, thus there should be an adaptive advantage in recombination if sex was selected in the course of evolution. One might hope that the new tools offered by the simulation methods of artificial life, genetic algorithms, (GA) and neural networks, might help the investigation by allowing the study of simplified models and of their detailed consequences. The Authors start "ro"some results on the effects of introducing crossover in a GA used for evolving a population of artificial animals trained on a simple task. Since there is a clear advantage in applying crossover versus simple mutations alone, this advantage could be retained by the population through selection: this hypothesis is tested in a model with local, individual genetic operators' probabilities by studying the emergent recombination frequencies. It is unexpectedly hard to analyze the results of the simulations, as the operator probabilities do not enter directly in the computation of fitness, while they have a well-known indirect influence on the 'behaviour' of fitness. The Authors are "onitorin"a trait that is not directly selected, thus being subject to the strong action of random drift.

learning process of neural networks simulating artificial life. In previous research the Authors (1990) "ompare"mutation and crossover as genetic operators on neural networks directly encoded as real vectors. With reference to crossover they were actually testing the building blocks hypothesis, as the effectiveness of recombination relies on the validity of such hypothesis. Even with the real genotype used, it was found that the average fitness of the population of neural networks is optimized much more quickly by crossover than it is by mutation. This indicated that the intrinsic parallelism of crossover is not reduced by the high cardinality. In this paper the Authors first "ummariz"such findings and then propose an interpretation in terms of the spatial correlation of the fitness function with respect to the metric defined by the average steps of the genetic operators. Some numerical evidence of such interpretation is given, showing that the fitness surface appears smoother to crossover than it does to mutation. This confirms indirectly that crossover moves along privileged directions, and at the same time provides a geometric rationale for hyperplanes.

Classification Tasks

Proceedings of the Second European Conference on Artificial Life

Nets)

Intelligence

Neural Networks

Satisfaction Problems

(AAAI-94)

Engineers D-II proposed. Each network has mutual connections and is assumed to be a living thing whose genes denote the connections among its units. In order to find out a network available to the current task, the simulation of evolution processes of the networks is executed.

Genetic Method

its Application to Action Control Neural Networks, Nagoya (Japan)

Response to Warfarin normalised ratio (INR) for patients treated with Warfarin was investigated. Neural networks were obtained by using all the predictor variables in the neural network, or by using a genetic algorithm to select an optimal subset of predictor variables in a neural network. The use of a genetic algorithm gave a marked and significant improvement in the prediction of the INR in two of the three cases investigated. The mean error in these cases, typically, reduced from 1.02+or-0.29 to 0.28+or-0.25 (paired t-test, t=-4.71, $p<0.001$, n=30). The use of a genetic algorithm with Warfarin data offers a significant enhancement of the predictive ability of a neural network with Warfarin data, identifies significant predictor variables, reduces the size of the neural network and thus the speed at which the reduced network can be trained, and reduces the sensitivity of a network to over-training.

two-layer, optoelectronic neural network using a genetic algorithm is demonstrated, and results for the 3 bit exclusive-or function are presented.

Applications Yonezaki

developed using a distributed genetic algorithm. A search for the optimal architecture and weights of a neural network comprising binary, linear threshold units is performed. For each individual unit, the Authors look "o"the optimal set of connections and associated weights under the restriction of a feedforward network structure. This is accomplished with the modified genetic algorithm, using an objective function-fitness-that considers, primarily, the overall network error; and, secondarily, using the unit's possible connections and weights that are preferable for continuity of the convergence process. Examples are given showing the potential of the proposed approach.

Genetic-Based Learning

91, pp.207-216, Varela,F.J. and Bourgine,P. (Eds)

Algorithm

Technology

Algorithms and Neural Networks

Genetic-Algorithm Techniques

Neurons GenLearn, for training multilayered neural networks. GenLearn uses techniques from the field of genetic algorithms to perform a global search of weight space and, thereby, to avoid the common problem of getting stuck in local minima. GenLearn is based on survival of the fittest hidden neuron. In searching for the most fit hidden neurons, GenLearn searches for a globally optimal internal representation of the input data. A big advantage of the GenLearn procedure over the generalized delta rule (GDR) in training three-layered neural nets is that, during each iteration of GenLearn, each weight in the first matrix is modified only once, whereas, in the GDR procedure, each weight in the first matrix is modified once for each output-layer neuron. What makes this such a big advantage is that, although GenLearn often reaches the desired mean square error in about the same number of iterations as the GDR, each iteration takes considerably less time.

Neural Networks Informatics simulating natural processes in order to solve large and difficult problems. One example which is attracting increasing attention is the idea of a genetic algorithm (GA). The first part of this paper provides a review of the basic concepts underlying genetic algorithms. The methodology is illustrated by a simple example, and some of the issues involved in more advanced GAs are discussed. Finally, it describes some of their applications. The second part describes in some detail research carried out in applying genetic algorithms to the field of neural networks, in particular to the multi-layer perceptron (MLP). This work falls into two main areas. The first of these deals with the question of the design of a neural network architecture, and the choice of a training regime for a particular problem. The second area of application is to the basic learning process itself. Traditionally, the MLP has been trained by a process called back-propagation. This paper reports on an alternative method based on a GA, and it is argued that such an approach has many advantages over back-propagation.

Evolution-Theoretic Principles multilayer perceptrons, i.e. a simple form of feed-forward neural nets. The training of a neural net can be regarded to be a parametric optimization problem, for which several possible algorithms are known. These differ in efficiency, given a fixed complexity of structure and classification task, and in the implementation constrains on parallel hardware. The Authors introduce "new method based on evolution-theoretic principles using operators from genetic algorithms, that is well suited for a parallel implementation on MIMD architectures. Finally they provide some results on parity problems.

Propagation Neural Networks and Genetic Algorithms

Networks

Design of Artificial Neural Networks

network, we wish to find a superior network architecture; this is called the network design problem. One approach is to use evolutionary methods, or evolutionary network design (END). The contiguity problem consists of counting the number of clumps of 1's in a binary input field. It is a good test problem for END because, for back-propagation networks, the space of network architectures has been characterized with respect to network generalization ability. We present experience gained using END to find superior network architectures for the contiguity problem.

State of the Art Neural Networks

backpropagation, often employ some form of gradient descent in their search for an optimal weight set. The problem with such algorithms is their tendency to converge to local minima, or not to converge at all. Genetic algorithms simulate evolutionary operators in their search for optimality. The techniques of genetic search are applied to training a neural network for target detection in infrared imagery. The algorithm design, parameters, and experimental results are detailed. Testing verifies that genetic algorithms are a useful and effective approach for neural network training.

Multilayer Perceptrons and Genetic Algorithms

Classification Neural Networks

Computers

Programming computer-aided design and testing of cerebellar model arithmetic computer (CMAC) encoded neural network regulators. The design and testing problem is viewed as a game in that the controller parameters are to be chosen with a minimax criterion, i.e. to minimize the loss associated with their use on the worst possible plant parameters. The technique permits analysis of neural strategies against a set of plants. This gives both the best choice of control parameters and identification of the plant configuration which is most difficult for the best controller to handle.

Unsupervised Learning by Genetic Algorithm Neural Networks, Nagoya (Japan)

Hierarchical Intelligent Control Neural Networks, Nagoya (Japan)

Algorithms for Heirarchical Intelligent Control Neural Networks, Nagoya (Japan)

Optimization Problems Neural Networks

which will embody both neural network and genetic algorithm optimization procedures. The system is modularly structured, with neural networks at the lower (recognition) level of the simple brain of the robot, and at the higher level prescribed decision behaviors are followed. It is the higher level parameters determining the nature of the decisions made that are to be optimized via genetic algorithms. Having sketched the structure and method of operation of the prototype robot, a community of robots situation is introduced as the next stage, for optimization within a robot-inhabited world.

Logic complex, nonlinear, and ill-conditioned optimization problems. Often, traditional optimization schemes are inadequate or inapplicable for such tasks. Genetic Algorithms (GA's) are a class of optimization procedures whose mechanics are based on those of natural genetics. Mathematical arguments show how GAs bring substantial computational leverage to search problems, without requiring the mathematical characteristics often necessary for traditional optimization schemes (e.g., modality, continuity, availability of derivative information, etc.). GA's have proven effective in a variety of search tasks that arise in neural networks and fuzzy systems. This presentation begins by introducing the mechanism and theoretical underpinnings of GA's. GA's are then related to a class of rule-based machine learning systems called learning classifier systems (LCS's). An LCS implements a low-level production-system that uses a GA as its primary rule discovery mechanism. This presentation illustrates how, despite its rule-based framework, an LCS can be thought of as a competitive neural network. Neural network simulator code for an LCS is presented. In this context, the GA is doing more than optimizing and objective function. It is searching for an ecology of hidden nodes with limited connectivity. The GA attempts to evolve this ecology such that effective neural network performance results. The GA is particularly well adapted to this task, given its naturally-inspired basis. The LCS/neural network analogy extends itself to other, more traditional neural networks. Conclusions to the presentation discuss the implications of using GA's in ecological search problems that arise in neural and fuzzy systems.

Problems

Processing Conference on Artificial Life time-dependent processing with recurrent networks. Both structure and weights of these networks are evolved by a fine-grained parallel genetic algorithm. The parallel nature of this algorithm, which enables the co-evolution of clusters of networks, made it possible to successfully solve three non-trivial temporal processing problems. One of these problems consists of evolving a trail-following behaviour for an artificial ant.

Algorithms

Using Genetic Algorithms correlational associative memory model using the Genetic Algorithms and a new training algorithm for this model is described. The recalling process of a model described by direction cosine is insufficient for the better understanding of the dynamical behavior of the model. In order to know the characteristics of memorized states, the methodology of the Genetic Algorithms applied to analyze the recalling process concerned with each memorized state is proposed. Furthermore, before the analyzing, the LU-algorithm is proposed to give the model the ability of keeping a wide basin in both highly memorized rates and mutually non-orthogonal states. Finally, results of experiments related to the basin analysis are shown.

executable programs from labeled training data. It differs from the conventional methods of Genetic Algorithms because it manipulates tree structures of arbitrary size and shape rather than fixed length binary strings. We apply GP to the development of a processing tree with a dendritic, or neuron-like structure: measurements from a set of input nodes are weighted and combined through linear and nonlinear operations to form an output response. Unlike conventional neural methods, no constraints are placed upon size, shape, or order of processing withing the network. This network is used to classify feature vectors extracted from IR imagery into target/nontarget catagories using a database of 2000 training samples. Performance is tested against a separate database of 7000 samples. For purposes of comparison, the same training and test sets are used to train two other adaptive classifier systems, the binary tree classifier and the Backpropagation neural network. The GP network acheives higher performance with reduced computational requirements.

algorithms have been used for auto-designing fuzzy systems, (2) how neural networks are combined with fuzzy systems in commecial applications, and (3) how fuzzy systems are used to improve the performance of neural networks and genetic algorithms.

demonstrates a three-phase automated design approach to pattern recognition. The experiment generates morphological feature detectors and then uses a novel application of genetic algorithms to select cooperative sets of features to pass to a neural net classifier. The self-organizing hybrid learning approach embodied in this closed-loop design methodology is complementary to conventional artificial intelligence (AI) expert systems that utilize rule-based approaches and a specific set of design elements. This experiment is part of a study directed to emulating the nondirected processes of biological evolution. The approach we discuss is semiautomatic in that initialization of computer programs requires human experience and expertise to select representations, develop search strategies, choose performance measures, and devise resource-allocation strategies. The hope is that these tasks will become easier with experience and will provide the means to exploit parallel processing without the need to analyze or program an entire design solution.

International Conference at Innsbruck, Austria with a genetic algorithm is discussed. The search by the recombination operator is hampered by the existence of two functionally equivalent symmetries in feedforward neural networks. To sidestep these representation redundancies we reorder the hidden neurons on the genotype before recombination according to a weight sign matching criterion, and flip the weight signs of a hidden neuron's connections whenever there are more inhibitory than excitatory incoming and outgoing links. As an example we optimize a feedforward neural network that implements a nonlinear optimal control law. The neural controller has to swing up the inverted pendulum from its lower equilibrium point to its upper equilibrium point and stabilize it there. Finding weights of the network represents a nonlinear optimization problem which is solved by the genetic algorithm.

Learning Behavior: From Animals to Animats

of temporal processing with recurrent networks. A genetic algorithm is used to evolve both structure and weights, so as to alleviate the design and learning problem recurrent networks suffer from. The viability of this approach is demonstrated by successfully solving two nontrivial temporal processing problems. The important technique of teacher forcing is identified and its influence on the performance of the algorithm is empirically demonstrated.

are treated with the help of a Kohonen network extended by a genetic algorithm (GA). The optimal solution is assumed to have continuous dependence on the external variables. The GA was generalized to organize individuals into subpopulations, which were allocated in the space of the external variables in an optimal fashion by Kohnonen digitization. Individuals were allowed to breed within their own subpopulations and in neighboring ones (migration). To illustrate the strength of the modified GA the optimal control of a simulated robot-arm is treated: a falling ping-pong ball has to be caught by a bat without bouncing. It is shown that the simultaneous optimization problem (for different values of the external parameter) can be solved successfully, and the migration can considerably reduce computation time.

Operators

Genetic Algorithms

Engineers)

Applications

The Authors propose algorithm highly inspired on biological concepts for generating neural networks oriented to solve particular problems given on terms of input and output. With this algorithm they intend to specify formal tools of general use for network definition, and to disclose underlying processing structures of the living organisms. The concepts of genetic code, embryogenesis and evolution are the main keys in the development of the algorithm they propose.

1990

future military aircraft system designs. A recently developed concept of vectoring aircraft thrust makes use of flexible exhaust nozzles. Subtle modifications in the nozzle wall contours produce a nonuniform flowfield containing a complex pattern of shock and expansion waves. The end result, due to the asymmetric velocity and pressure distributions, is vectored thrust. Specification of the nozzle contours required for a desired thrust vector angle (an inverse design problem) has been achieved with genetic algorithms. However, this approach is computationally intensive, preventing nozzles from being designed on demand, which is necessary for an operational aircraft system. An investigation was conducted into using genetic algorithms to train a neural network in an attempt to obtain, in real time, two-dimensional nozzle contours. Results show that genetic-algorithm trained neural networks provide a viable, time-efficient alternative for designing thrust vectoring nozzle contours. Thrust vector angles up to 20 deg were obtained within an average error of 0.0914 deg. The error surfaces encountered were highly degenerate and thus the robustness of genetic algorithms was well suited for minimizing global errors.

Input Space

Algorithms

Connectivity

Artificial Intelligence and Simulated Behavior, Sussex, England 1989. Pitman Publishers.

Schedules

have been shown to yield good performance for neural network weight optimization are really genetic hill-climbers, with a strong reliance on mutation rather than hyperplane sampling. These results are consistent with the theoretical results of Goldberg (1991) analyzing real-coded genetic algorithms. We argue that neural network learning applications such as neurocontrol problems are perhaps more appropriate for these genetic hill-climbers than supervised learning applications because in reinforcement learning applications gradient information is not directly available. On an inverted pendulum control problem reinforcement learning produces competitive results with AHC, another well-known reinforcement learning paradigm for neural networks that employs temporal difference methods. The genetic hill-climbing algorithm appears to be robust over a wide range of learning conditions. We also discuss several approaches for evaluating neural network performance.

produced good performance for neural network weight optimization are really genetic hill-climbers, with a strong reliance on mutation rather than hyperplane sampling. Initial results are presented using genetic hill-climbers for reinforcement learning with multilayer neural networks for the control of a simulated cart-centering and pole-balancing dynamical system. "Genetic reinforcement learning"produces competitive results with AHC, a well-known reinforcement learning paradigm for neural networks that employs temporal difference methods.

Computing Environment Using NeuroGraph execution of neural networks and genetic algorithms in a distributed computing environment. The simulator parts either run on single computers or as distributed applications on Unix/X-based networks, consisting of personal computers, workstations, or multi-processors. The parallelization component offers the possibility to divide computational tasks into concurrently executable modules, according to restrictions due to the neural net topology and computer net capabilities, ie. NeuroGraph tries to select the best configuration out of the available distributed hardware environment to fit performance requirements.

Topology with a Genetic Algorithm Euclidian space. The leaf layer consists of a set of local linear classifiers, and the whole system can be trained in a supervised manner to form a piecewise linear model. In this paper a Genetic Algorithm (GA) is used to optimise the topology of the tree. We discuss the properties of the genetic coding scheme, and argue that the GA/bumptree does not suffer from the same scaling problems as other GA/neural-net hybrids. Results on test problems, including a non-trivial classification task, are encouraging, with the GA able to discover topologies which give improved performance over those generated by a constructive algorithm.

Boltzmann Methods

Vision and Behavior or PolyWorld: Life in a New Context

Neural Networks, Nagoya (Japan)

systems, i.e., artificial neural networks (ANNs), and evolutionary search procedures, like genetic algorithms (GAs), has attracted a lot of attention. Evolutionary ANNs (EANNs) can be considered as the combination of ANNs and evolutionary search procedures. This article first distinguishes among three kinds of evolution in EANNs, i.e., the evolution of connection weights, of architectures, and of learning rules. Then it reviews each kind of evolution in detail and analyzes critical issues related to different evolutions. The review shows that although a lot of work has been done on the evolution of connection weights and architectures, few attempts have been made to understand the evolution of learning rules. Interactions among different evolutions are seldom mentioned in current research. However, the evolution of learning rules and its interactions with other kinds of evolution, play a vital role in EANNs. Finally, this article briefly describes a general framework for EANNs, which not only includes the aforementioned three kinds of evolution, but also considers interactions among them.

Examples networks. In contrast to the passive paradigm, the learning in the active paradigm is initiated by the machine learner instead of its environment or teacher. The Authors present "learning algorithm that uses a genetic algorithm for creating novel examples to teach multilayer feedforward networks. The creative learning networks, based on their own knowledge, discover new examples, criticize and select useful ones, train themselves, and thereby extend their existing knowledge. Experiments on function extrapolation show that the self-teaching neural networks not only reduce the teaching efforts of the human, but the genetically created examples also contribute robustly to the improvement of generalization performance and the interpretation of the connectionist knowledge.