A Review on Merging Some Recent Techniques with Artificial Neural Networks

Manish Sarkar and B. Yegnanarayana
Department of Computer Science & Engineering
Indian Institute of Technology, Madras - 600 036, India
e-mail: manish@bronto.iitm.ernet.in

ABSTRACT

In the last few years, there has been a large upswing in research activities aimed at synthesizing artificial neural networks with other well-established paradigms, like evolutionary computation, fuzzy logic, rough sets and chaos. In this paper, we briefly discuss the merits of these paradigms from artificial neural networks point of view, and we illustrate how these paradigms can be fused with the existing artificial neural network models to make the later one more efficient.

I. INTRODUCTION

During the past decade, there has been a considerable growth of interest in integrating Artificial Neural Network (ANN) with other existing paradigms. This paper is intended to briefly review some of these integration attempts.

ANNs are a new breed of information processing systems that are constructed to exploit some of the organizational principles that characterize the human brain [114] [62]. The central theme of the ANN research focuses on modeling the brain as a parallel computational device for various computational tasks, that are performed poorly by conventional serial computers. Models of ANN are specified by three basic entities: models of the processing element, architecture, and learning laws [62]. ANNs have a large number of highly interconnected processing elements that generally operate in parallel, and are configured in regular architectures. The collective behavior of an ANN, like a human brain, demonstrates the ability to *learn* [14] [12], *recall* [61] [71] [86], and *generalize* [103] from training patterns or data. Through learning, the network architecture and the weights connecting the processing elements are updated such that a network can perform a specific human reasoning task [66], [46]. The task may be *pattern association*, *pattern mapping*, *pattern classification*, *pattern clustering*, *feature mapping* [114] [39] [38], etc. Pattern association task involves capturing a set of patterns or a set of input-output pattern pairs in such a way that when a test pattern is presented, the pattern or pattern pair corresponding to the input pattern is recalled [114]. In pattern mapping or function approximation, given a set of input patterns and the corresponding output patterns, the objective is to capture the implicit functional relationship between the input pattern and the output, so that when a test input is given, the corresponding output pattern is retrieved [114]. In pattern classification, there is a fixed number of classes into which the input pattern has to be classified [38]. In the case of pattern clustering, the task is to identify the subset of patterns possessing similar features and group them together. The idea of a feature map is to design a network that would organize the given set of patterns in accordance with similarity of features among them. By looking at the output of the feature map network one can visually obtain an idea of how different patterns are related [114]. To learn these tasks, there exist several learning laws, like Hebbian learning law [36], perceptron learning law [87], backpropagation learning law [87], Boltzmann learning law [58] [36], etc.

The learning law in an ANN basically optimizes certain objective function which reflects the constraints associated with the given task. A set of optima in this objective function generally corresponds to different stable states of the network. Most of the learning laws utilize gradient based approach for this optimization purpose. However, due to its deterministic nature, gradient based methods frequently get stuck in local optima or saddle points. This is because, the step size and step direction of the optimization process are dictated by the local information supplied by the gradient. This drawback, however, can be avoided by choosing the step size and step direction stochastically in a controlled manner. The efficiency of the search for the global optimum can be enhanced further, if it is carried out in parallel. *Evolutionary Computation* [29] is one such biologically inspired method, where a population of solutions are probabilistically explored over a sequence of generations to reach the globally optimum solution. The integration of the evolutionary computational technique into ANN models is called *Neuro-Evolutionary* technique, which can be used to enhance the learning capability of the ANN model [48]. This technique is also useful to determine the suitable topology of a network and to select the proper learning law [110].

In a classification task, an ANN is used to find the decision regions in the input pattern space. But, if the patterns from different classes are overlapping, then it is difficult for the ANN to find the class boundary. In pattern mapping also similar problems may arise when the inputs or target outputs are ill-defined or *fuzzy*. These situations are common in many pattern recognition tasks because of inherent fuzziness associated with the human reasoning. The pattern recognition capability of ANNs can be made powerful, if fuzzy logic is incorporated into the conventional ANN models. The resulting systems are called *Neuro-Fuzzy* systems [62] [49].

In some cases, ANN training faces difficulties to find a class boundary when the same input training pattern belongs to one class in some examples, and to another class in some other examples. This scenario is due to the presence of *rough* uncertainty, which arises from the *indiscernibility* of the objects based on the input features. The classification ability of an ANN can be significantly improved if the input data set is processed to reduce the rough uncertainty. Motivated by this idea, a new promising area based on *Neuro-Rough* synergism is emerging.

The ability of a feedback network to store patterns can be improved, if we can exploit the *chaotic* part of the networks dynamics. This observation has resulted in proposing hybrid neurons, known as *chaotic neurons*. Different models of chaotic neurons are studied, and initial results are quite promising [3].

II. RECENT MERGING TECHNIQUES

A. Evolutionary Computation

Evolutionary computation (EC) [29] is a technique to encompass a variety of population-based problem solving techniques that mimic the natural process of Darwinian evolution. Current research in the evolutionary computation has resulted in powerful and versatile problem solving mechanisms for global searching, adaptation, learning and optimization in a variety of pattern recognition domains. The main avenues for research in evolutionary computation are *genetic algorithms* [42] [34], *genetic programming* [60], *evolutionary strategies* [98] and *evolutionary programming* [31] [28] [30]. Genetic algorithms deal with chromosomal operators, while genetic programming stresses on operators of more general hierarchical structures. Evolutionary strategies emphasize behavioral changes at the level of the individual, whereas evolutionary programming focuses on behavioral changes at the level

This work was carried out while Manish Sarkar held a Dr. K. S. Krishnan Research Fellowship from Department of Atomic Energy, Government of India.

0-7803-4778-1/98 $10.00 © 1998 IEEE

of the species. The common factor underlying all these approaches is the emphasis on an ensemble of solution structures, and the evaluation and evolution of these structures through specialized operators that mimic their biological counterparts, in response to an ever changing environment. Specifically, all of them maintain a population of trial solutions, impose random changes to those solutions, and incorporate the use of selection to determine which solutions are to be maintained into future generation and which are to be removed from the pool of trials.

From a mathematical point of view, all the EC techniques are *controlled, parallel, stochastic search and optimization* techniques. Since different learning techniques used in ANNs hinge on the optimization of various objective functions, it is possible to employ the EC for learning weights, learning network architectures, learning the learning laws, input feature selection and so on [110]. For instance, in a feedforward neural network, gradient based local search methods [38] can be substituted by the EC for weight training [84] [88]. In some cases, a more ambitious approach may be to exploit local search methods, like gradient descent, and global search methods, like EC, simultaneously [85]. The advantages of local search methods are better accuracy and fast computation. The disadvantages of the local search methods are stagnation at the suboptimal solutions and sensitiveness to the initialization. The EC, on the otherhand, is a global search method which can avoid local optima, and does not have the initialization problem [97]. However, the EC can suffer from extremely slow convergence before arriving at the accurate solution. This is because, the EC uses minimal *a priori* knowledge, and does not exploit available local information [85]. In fact, in the search space the EC is good for exploration, whereas the gradient descent is good for exploitation. Therefore, by utilizing both of them, merits of both methods, i.e., speed, accuracy, reliability and computation time can be obtained. Yao *et al.* [111] have proposed one such method to evolve the topology (weights and architecture) of a feedforward neural network, where they exploit both evolutionary programming and backpropagation algorithm [87], simultaneously.

In the EC techniques, the whole population set evolves over and over again generations after generations. At the last generation, the network which has the highest fitness is considered to be the desired optimal network for the given task. Instead of choosing a single network as the desired network, in [112], all the networks in the population are considered as the desired networks. Here, the final result is obtained by combining all the individuals in the last generation to make best use of all the information contained in the whole population. This result, in fact, confirms the fact that a population contains more information than a single individual, and the EC is used to exploit that. In particular, in [112], the classification outputs of all the networks in the population are combined and thereafter, the output class is determined by a majority voting scheme.

There are, two different approaches to evolve the topology of the networks using the EC. In one method (also known as *Pitt's approach* in genetic algorithm community [70]), each element of the population represents one complete network. Consequently, whole population is an ensemble of many networks with different topologies. In the process of evolution they compete among themselves, the weak individual dies, the strong survives and reproduces. In the other approach (also known as *Michigan approach* in genetic algorithm community [70]), the whole population represents only one network, i.e., each member of the population represents a part of the network. The second method is more time and space efficient. But, it needs, (a) delicate credit assignments, for which a heuristic method should distribute positive or negative credits among the members of the population, and (b) the members of the population, i.e., different parts of the network, to cooperate with each other so that they can build the complete network [70] [108].

In [108] [107] [11], the EC-based techniques are successfully used to optimally configure radial basis function networks so that the networks generalize well. In another significant development, Angeline *et al.* [4] have used the EC to configure recurrent neural networks. It should be noted that, gradient based approach is completely useless here as it needs the objective function to be differentiable. Using the EC, Jockusch *et al.* [50] have introduced an attractive self-organizing map [57] training strategy, where it is possible to find the number of self-organizing map output units automatically. Moreover, in their scheme the training of the self-organizing map is less likely to be stuck in local optima. In literature, the EC has been used for clustering also [89] [97]. Here, the proposed algorithm automatically finds the optimal number of clusters present in the input data set. Therefore, the clustered output can further be used to construct a probabilistic neural network optimally [92]. Currently, researchers are working on different evolutionary methods, which can be utilized to learn weights, architectures and learning laws, simultaneously [110]. However, the problem here is that the search space for this type of problem becomes prohibitively large, and it needs an enormous amount of computing time. These drawbacks can be removed if parallel machines are used to implement the search operation, or the search operation is made more efficient and less time consuming using adaptive EC operators [100] [67] [93].

B. Fuzzy Logic

The concept of fuzzy sets was first introduced by L. Zadeh in 1965 [115], as a mathematical way to represent vagueness present in the human reasoning. Fuzzy sets can be considered as a generalization of classical set theory. In the classical set, an element of the universe either belongs to or does not belong to a set. That is, the belongingness of the element is crisp–it is either *yes* (in the set) or *no* (not in the set). In fuzzy sets, the belongingness of the element can be anything in between *yes* or *no*; for example, a set of *tall* persons. We cannot identify a person as tall in a *yes/no* manner, as there does not exist any well-defined boundary for the set *tall* [75]. Mathematically, a fuzzy set is a mapping (known as *membership function*) from the universe of discourse to [0, 1]. The higher the membership value of an input pattern to a class, the more is the belongingness of the pattern to the class [51] [52]. Therefore, any concept that uses fuzzy sets requires the membership function to be defined. There may be various possible membership functions for the set *tall*. Non uniqueness of membership functions may raise a question: how does a designer know which one to use? In fact, the designer can obtain the membership function from an expert (subjective computation) or from the data (objective computation) [75] [7] [10] [9]. Following the idea of fuzzy sets, the concept of crisp numbers has been generalized to fuzzy numbers [53]. The reasoning with fuzzy sets and fuzzy numbers is known as fuzzy logic [59].

ANNs adopt numerical computations for learning; however, numerical quantities evidently suffer from a lack of representative power [76]. There are many applications where information cannot be obtained in terms of numerical values–rather it is possible to represent the information in linguistic terms only [63]. In a washing machine the input and output of the machine can only be represented in linguistic terms, like *dirty* clothes, *clean* clothes, etc. The linguistic terms, *dirty* and *clean* do not have any precise numerical value as the concepts of *dirty* and *clean* are overlapping, and there is no strict boundary between them. Therefore, if the operation of the washing machine is to be modeled by an ANN, it should be capable of dealing with the fuzziness associated with the linguistic terms. While training an ANN for a classification task, we generally use crisp target values, which can be either zero or one. This kind of target assignment can be generalized by exploiting fuzzy sets, where target values can be anything in between zero and one. Use of fuzzy concepts in ANNs are also supported by the fact that the psycho-physiological process involved in the human reasoning does not employ precise mathematical formulation [73]. Specifically, fuzzy theory can be incorporated in an ANN at the following levels: (a) at output and target levels, (b) at input level, and (c) at each neuron level in terms of weight value, basis function and output function. The appropriate level of incorporation of fuzzy theory depends on the given problem.

For a classification task, perceptron algorithms exhibit an erratic behavior when the data is not linearly separable. In [55], fuzzy set theory is introduced into the perceptron objective function to ameliorate this convergence problem in linearly nonseparable cases. This improvement is possible, because introduction of fuzzy sets into the learning algorithm makes the decision boundary a soft one, that is, near the decision boundary, class labels of the input pattern space slowly change from one class to another class.

In [91] [74], ANN outputs are interpreted as fuzzy membership values, and using this idea the conventional mean square error objective function has been extended to various fuzzy objective functions. Here, the learning laws are derived by minimizing the fuzzy objective functions in a gradient descent manner. In [94] [91], a similar idea is employed to extend the concept of cross entropy [23], which is another popularly used crisp error function. It has been found that incorporation of fuzziness in the objective functions leads to better classification rate.

A neural network reinforcement learning algorithm, with linguistic critic signals like *good*, *bad*, is proposed in [63]. The network is able to process and learn numerical information as well as linguistic information in control theoretic applications.

In [21], three existing competitive learning algorithms, namely the unsupervised competitive learning, learning vector quantization, and frequency sensitive competitive learning, are fuzzified to form a class of fuzzy competitive learning algorithms. Unlike the crisp counterpart, where only one output unit wins, here all the output units win with different degrees. Thus, the concept of win has been formulated as a fuzzy membership function. It has been observed that this scheme leads to better convergence and better classification rate.

In [102], Kohonen's clustering network has been generalized to its fuzzy counterpart. One advantage of this approach is that final weight vectors of the clustering network do not depend on the input sequence. Moreover, it uses a systematic approach to determine the learning rate parameter and size of the neighborhood.

A fuzzy adaptive resonance theory model, capable of rapid learning of recognition categories in response to an arbitrary sequence of binary input patterns, is proposed in [15]. This upgradation from binary ART1 [32] to fuzzy ART is achieved by converting the crisp logical operators used in binary ART1 to the corresponding fuzzy logical operators. As a result of this upgradation, the learning becomes fast, and previously learned memories are not rapidly erased in response to statistically unreliable input fluctuations. Fuzzy ARTMAP provides a more powerful realization of ART concepts [16]. It can autonomously learn, recognize and make prediction.

Wang *et al.* [105] have proposed fuzzy basis functions to design a radial basis function network [38], which can accept both numerical inputs as well as fuzzy linguistic inputs.

In [82], Pedrycz has proposed an ANN model based on fuzzy logical connectives. Instead of using linear basis functions, he has utilized fuzzy aggregation operators. In [83] and [40], this technique has been extended to a more general one where inhibitory and excitatory characteristics of the inputs are captured by employing direct and complemented, i.e., negated input signals. The advantage of this approach is that problem specific fuzzy *a priori* knowledge can be incorporated into the network easily.

In [47], Ishibuchi *et al.* have proposed an ANN learning algorithm where expert's *a priori* knowledge, in terms of fuzzy *if-then* rules, can be exploited to learn the information supplied by the numerical data. This type of approach has been used for both function approximation and classification. The whole scheme is based on clever manipulations of fuzzy numbers.

In [58], Kosko generalized the concept of conventional associative memory [58] to fuzzy associative memory. Unlike conventional associative memory, where the association is between two crisp sets, here the association is between two fuzzy sets.

In many complicated problems, it is beneficial to divide the original pattern recognition task into several smaller subtasks and combine their individual solutions. Each of these subtasks can be accomplished by a neural subnetwork. The conflicting information supplied by the information sources, i.e., the subnetworks, can be fused by applying the concept of fuzzy integral [54]. Instead of treating each module identically, the subjective evaluation potential of fuzzy integral-based method stresses on those modules or sets of modules, which supply the most evidence toward the determination of the output [18] [17] [90].

In addition to the above applications, fuzzy theory can be employed to speed up the training of an ANN. In [19], a fuzzy rule base is used to dynamically adapt the learning rate and momentum parameters of a feedforward neural network with backpropagation learning algorithm. In a similar approach [20], Choi *et al.* have proposed an incremental updating scheme to control the value of vigilance parameters of ART networks.

C. Rough Sets

In any classification task the aim is to form various classes where each class contains objects that are not noticeably different. These *indiscernible* or indistinguishable objects can be viewed as basic building blocks (concepts) used to build up a knowledge base about the real world. For example, if the objects are classified according to color (red, black) and shape (triangle, square and circle), then the classes are: red triangles, black squares, red circles, etc. Thus, these two attributes make a *partition* in the set of objects and the universe becomes coarse. Now, if two red triangles with different areas belong to different classes, it is impossible for anyone to correctly classify these two red triangles based on the given two attributes. This kind of uncertainty is referred to as *rough uncertainty* [78] [80]. The rough uncertainty is formulated in terms of *rough sets* [79]. Obviously, the rough uncertainty can be completely avoided if we can successfully extract the essential features so that distinct feature vectors are used to represent different objects. But, it may not be possible to guarantee as our knowledge about the system generating the data is limited.

In any classification problem, two input training patterns $\mathbf{x}_r$ and $\mathbf{x}_s$ (where $\mathbf{x}_r, \mathbf{x}_s \in \mathbf{X}$, the set of all input training patterns) are called *indiscernible* with respect to the qth feature, when the qth component of these two patterns have the same value. Mathematically, this indiscernibility can be represented as $\mathbf{x}_r \mathbf{R}^q \mathbf{x}_s$ iff $x_{rq} = x_{sq}$, where $\mathbf{R}^q$ is a binary relation over $\mathbf{X} \times \mathbf{X}$. Obviously, $\mathbf{R}^q$ is an equivalence relation, that partitions the universal set $\mathbf{X}$ into different equivalence classes. This idea can be generalized to take some or all the features into our consideration. Without loss of generality, based on a particular set of features, let $\mathbf{R}$ be an equivalence relation on the universal set $\mathbf{X}$. Moreover, let $\mathbf{X}/\mathbf{R}$ denote the family of all equivalence classes induced on $\mathbf{X}$ by $\mathbf{R}$. One such equivalence class in $\mathbf{X}/\mathbf{R}$, that contains $\mathbf{x} \in \mathbf{X}$, is designated by $[\mathbf{x}]_{\mathbf{R}}$. Now, in any classification problem, the objective is to approximate the given output class $\mathbf{A} \subseteq \mathbf{X}$ by $\mathbf{X}/\mathbf{R}$. For the output class $\mathbf{A}$, we can define lower $\overline{\mathbf{R}}(\mathbf{A})$ and upper $\underline{\mathbf{R}}(\mathbf{A})$ approximations, which approach $\mathbf{A}$ as closely as possibly from inside and outside, respectively [56]. Here, $\underline{\mathbf{R}}(\mathbf{A}) = \cup\{[\mathbf{x}]_{\mathbf{R}} \mid [\mathbf{x}]_{\mathbf{R}} \subseteq \mathbf{A}, \mathbf{x} \in \mathbf{X}\}$ is the union of all equivalence classes in $\mathbf{X}/\mathbf{R}$ that are contained in $\mathbf{A}$, and $\overline{\mathbf{R}}(\mathbf{A}) = \cup\{[\mathbf{x}]_{\mathbf{R}} \mid [\mathbf{x}]_{\mathbf{R}} \cap \mathbf{A} \neq \phi, \mathbf{x} \in \mathbf{X}\}$ is the union of all equivalence classes in $\mathbf{X}/\mathbf{R}$ that overlap with $\mathbf{A}$. A rough set $\mathbf{R}(\mathbf{A}) = \left\langle \overline{\mathbf{R}}(\mathbf{A}), \underline{\mathbf{R}}(\mathbf{A}) \right\rangle$ is a representation of the given set

A by $\underline{\mathbf{R}}(\mathbf{A})$ and $\overline{\mathbf{R}}(\mathbf{A})$. The set difference, $\overline{\mathbf{R}}(\mathbf{A}) - \underline{\mathbf{R}}(\mathbf{A})$, is a rough description of the boundary of **A** by the equivalence classes of $\mathbf{X}/\mathbf{R}$. The approximation is rough uncertainty free if $\overline{\mathbf{R}}(\mathbf{A}) = \underline{\mathbf{R}}(\mathbf{A})$. Thus, when all the patterns from an equivalence class do not carry the same output class labels, rough ambiguity is generated as a manifestation of the one-to-many relationship between that granule and the output class labels.

In ANN design, one critical problem is to determine how many input units are essential. Obviously, it depends on the dimension, i.e., the number of features present in the input data. Using rough sets, in many cases, it is possible to decrease the dimension of the input data without losing any information. A set of features is sufficient to classify all the input patterns if the rough ambiguity, i.e., the quantity $(\overline{\mathbf{R}}(\mathbf{A}) - \underline{\mathbf{R}}(\mathbf{A}))$, for this set of features is equal to zero. Thus, using this quantity, it is possible to select a proper set of features from the given set of data [81].

In any classification task, all the features do not usually carry equal weightage. Hence, to facilitate the ANN training as well as to increase the classification efficiency, it is possible to put different weightages on the collected input features so that the class separability increases. The proper weightage can be evaluated from the importance of each feature, which can further be determined by rough sets [95]. In [80], it is claimed that, for a classification task, the number of hidden units needed in a feedforward neural network is equal to the minimal number of features required to represent the data set without increasing the rough uncertainty.

One way to accelerate the ANN training is, to initialize the weights of the networks in such a manner that the initial decision region is closer to the desired one. For that, a set of training data is collected, and the knowledge extracted from them through rough sets is used to initialize the ANN [5].

In [64], Lingras has proposed an architecture for rough neural networks, which consist of a combination of rough neurons and conventional neurons. Here, rough neurons use pairs of the upper and lower bounds as values for inputs and outputs. In certain practical situations, like road traffic control, it is preferable to develop prediction models that use tolerance ranges as values for input and output variables. It implies that each input value is a rough pattern represented in terms of the lower and upper bounds. Following similar approach, Lingras has applied [65] rough neurons in self-organizing map [38] for unsupervised classification of rough patterns. The author has demonstrated that, in the field of traffic control predictions obtained using rough neural networks are significantly better than the conventional ANN models.

In [109], it is argued that the underlying assumption of the randomness of the sample elements of a population causes each element of the population to lose its specific detail and identity, if it is described in terms of statistical parameters. In contrast, rough sets enhance each object identity by looking for its contexts in available data. They also claimed that the conventional ANNs, which depend on some of the principles of statistical regression and discriminant models, inherit the drawbacks of statistics. In order to circumvent these problems, they have proposed a rough neural network approach for faster and more accurate data processing.

D. Chaos

In many simple physical systems, it has been observed that there is no apparent relationship between causes and effects. For example, while water falls from a tap, the flow remains steady, laminar and regular for some time, and then it becomes unsteady, turbulent and irregular. Although the flow rate is constant, the behavior of the water flow becomes unpredictable. This kind of uncertainty, however, is not random in the sense that gathering more information does not help to avoid the uncertainty involved in the relationship between the cause and the effect. Therefore, standard statistical results cannot be applied here to solve this problem. Although the whole process is absolutely deterministic, this kind of apparent randomness is, in fact, generated by small differences in the initial values of the physical systems. This type of uncertainty has been termed as ***deterministic chaos*** or simply ***chaos*** [22] [41] [68]. Chaos theory attempts to explain the fact that complicated and unpredictable results can occur in the systems that are sensitive to their initial conditions. A popular example (known as ***butterfly effect*** [33]) of chaos states that, in theory, the flutter of a butterfly's wings in India could effect weather patterns in New York City, thousands of miles away. It implies that a very small occurrence can produce unpredictable and sometimes drastic results by triggering a series of increasingly significant events [33]. The results of discovering chaos are: (a) it draws a fundamental limit of the ability on the prediction, and (b) many real life random phenomena are more deterministic than that had been thought.

It has been claimed that chaotic dynamics exists in biological neurons [35]. Nonlinearity in brain arises because biological neurons contain appropriate feedbacks that can generate rhythms, which are essential for regularizing the neural functions. Here, the neurons act as oscillators, and while transferring information through the interactions among these neurons, chaos is generated. It has also been argued that stability-plasticity in the brain is observed due to its ability to convert the brain dynamics from highly ordered state to chaotic state and vice versa. Naturally, in order to mimic the human reasoning on machines, we must exploit the chaos part that already exists in the currently available feedback type ANNs. In fact, the response of a feedback type ANN may be so sensitive to the initial condition that unless a computer of infinite word length is employed in the simulation, no long term prediction is possible [77]. Such extreme sensitivity of feedback networks is but one, among many, talltale manifestation of chaos.

The dynamics of a feedback neural network can be explained by a state space and a guideline, that describes how the state evolves over time. Any such network that comes to a rest with the passage of time can be characterized by a fixed point in the state space. In some cases, the system does not come to the rest, but it cycles through a sequence of states to create a periodic orbit. Any such region, where the system settles down to, or attracted to, is termed as an ***attractor***. There can be several types of attractors like ***fixed point***, ***quasi-periodic***, ***periodic*** and ***strange attractors***. The strange attractor is an example of chaotic attractor, whereas the fixed point, periodic and quasi-periodic attractors are not [106]. In the state space, the orbits of the strange attractors sometimes diverge. But, the divergence cannot be continued forever as the state space is finite. Hence, the attractors must fold over onto itself. These stretching and folding operations continue repeatedly, creating folds within folds. Consequently, chaotic attractors generate a ***fractal*** [69] like structure that reveals more and more as it is increasingly magnified. This stretching operation systematically removes the initial information, and makes small scale uncertainty large. The folding operation also removes the initial information, but makes large scale uncertainty small. If we know the initial state of the network with some uncertainty (due to measurement error), after a short period of time the uncertainty specified initially covers the entire attractor and all predictive power is lost, and therefore, there exist no relationships between the past and the future, or the cause and the effect.

There are several avenues to exploit chaos under the ANN paradigm. It is believed that several limitations of the existing artificial neuron models are due to its grossly simplified structure. For example, the output of an ANN is smooth, whereas the output of a biological neuron actually forms a train of electrical spikes or neural pulses. Hence, using ***Hodgkin-Huxley cell equations*** [58], attempts are being made to create more complicated artificial neuron models. In this approach, the chaos, generated by Hodgkin-Huxley cell equations, is exploited for proper functioning of the neurons. In an impressive work, Freeman and his co-workers have demonstrated that different kinds of stimuli in animal cortex can be represented as chaotic attractors [113]. They successfully developed an artificial olfactory model, where the artificial neurons exploit chaos for its functioning [27].

Another group of researchers, mostly engineers and mathematicians, are extending the existing artificial neurons to exploit its chaotic behaviors [43] [99] [37]. Hence, they formulated different measures to find how chaotic the neural dynamics is, and based on that, they control the chaotic behavior of the neurons [72] [44]. Here, the chaotic variables may be the output activities of the neurons, and the control parameters may be the synaptic weights or the outputs of the external neurons [13]. In many cases, however, the proposed models do not have any direct physiological interpretation [13].

A large group of scientists are recently analyzing the chaotic dynamics of the existing feedback type ANNs [104] [101]. They are employing the periodic attractors, embedded in each chaotic attractor, to store the input patterns. Following this strategy, in [1] a chaotic associative memory is constructed. It has been observed that this type of model has the possibility to store a huge number of spatio-temporal patterns [3].

III. CONCLUSION

From the above discussion it is evident that ANNs enable computers to learn from the past data; evolutionary computation, fuzzy logic, rough sets and chaos allow the networks to manage uncertainty better. Together, they create the ability to solve many classes of human reasoning problems more efficiently. However, to integrate evolutionary computation, fuzzy logic, rough sets and chaos with the existing ANN models, the designer must be able to identify what type of uncertainty is present in the given problem. Unfortunately, probabilistic, fuzzy, rough and chaotic uncertainties are often confused. In fact, they are different facets of uncertainty. Fuzziness deals with vagueness between the overlapping sets [52] [8] [10], while probability concerns the likelihood of randomness of a phenomenon [62]; on the other hand, rough sets deal with coarse non-overlapping concepts [25] [26]. Both roughness and fuzziness do not depend on the occurrence of the event; whereas probability does. Fuzziness lies in the subsets defined by the linguistic variables, like *tall*, *big*, whereas indiscernibility is a property of the referential itself, as perceived by some observers, not of its subsets [26]. In fuzzy sets, each granule of knowledge can have only one membership value to a particular class. However, rough sets assert that each granule may have different membership values to the same class. Fuzzy sets deal with overlapping classes and *fine* concepts; whereas rough sets deal with nonoverlapping classes and coarse concepts. Chaos, on the otherhand, deals with uncertainty which is created by small differences in the initial values. This kind of uncertainty, however, is not random in the sense that gathering more information does not help to avoid the uncertainty involved in the relationship between the cause and the effect. This is neither fuzzy, nor rough, as it does not deal with overlapping classes or coarse concepts.

The above discussed four paradigms along with the ANN paradigm are collectively called *soft computing paradigm*; because they can tolerate imprecision, uncertainty and partial truth. Although, in this paper, we have discussed only neuro-fuzzy, neuro-evolutionary and chaotic neurons, it is possible to have more complicated fusions like *neuro-fuzzy-evolutionary* [92], *neuro-rough-fuzzy* [96], *neuro-fuzzy-chaotic*, *neuro-rough-fuzzy-evolutionary*, etc. These integration techniques are based on partnerships, in which each of the partners contributes a distinct methodology for addressing problems in its domain.

There are several other attractive paradigms which can be fused with the current ANN techniques. For example, *Artificial Ant System* [24], *Cultural Evolution* [6], *Immunity Net* [45] and *DNA Computing* [2], seem to be attractive and viable approaches that can be amalgamed with ANNs.

One key advantage of the ANN is that it is adaptive. Many existing paradigms can be fused into it easily. Although, as of now there are no strict guidelines for developing hybrid paradigms, the urge to develop models to perform human cognition tasks will continue to motivate researchers to explore new directions in this field.

REFERENCES

[1] M. Adachi and K. Ahira. Associative dynamics in a chaotic neural networks. *Neural Networks*, 10(1):83–98, January 1997.

[2] L. Adleman. Molecular computation of solutions to combinatorial problems. *Science*, (266):1021–1024., 1994.

[3] Y. V. Andreyev, Y. L. Belsky, A. S. Dmitreiv, and D. A. Kuminov. Neural networks information processing by dynamic chaos. *IEEE Transactions on Neural Networks*, 2(2):290–299, March 1996.

[4] P. J. Angeline, G. M. Saunders, and J. B. Pollack. An evolutionary algorithm that constructs recurrent neural networks. *IEEE Transactions on Neural Networks*, 5(1):54–64, January 1994.

[5] M. Banerjee, S. Mitra, and S. K. Pal. Knowledge-based fuzzy MLP with rough sets. In *IEEE International Conference on Neural Networks (Houston, USA)*, June 9-12 1997.

[6] R. K. Belew. When both individuals and populations search: Adding simple learning to genetic algorithm. In *Proceedings of Third International Conference on GA (George Mason University)*, pages 34–41, June 1989.

[7] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York, 1981.

[8] J. C. Bezdek. The thirsty traveller visits gamont: A rejoinder to "comments on fuzzy set – what are they and why?". *IEEE Transactions on Fuzzy Systems*, 2(1):43–45, February 1994.

[9] J. C. Bezdek. A review of probabilistic, fuzzy, and neural models for pattern recognition. In C. H. Chen, editor, *Fuzzy Logic and Neural Network Handbook*. McGraw-Hill, Inc, New York, 1996.

[10] J. C. Bezdek and S. K. Pal. *Fuzzy Models for Pattern Recognition, Eds.* IEEE Press, New York, 1992.

[11] S. A. Billings and G. L. Zheng. Radial basis function network configuration using genetic algorithm. *Neural Networks*, 8(6):877–890, 1995.

[12] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford Univ Press, 1995.

[13] F. C. Blondeau and G. Chauvet. Stable, oscillatory and chaotic regimes in the dynamics of small neural networks with delay. *Neural Networks*, 5:735–743, 1992.

[14] N. K. Bose and P. Liang. *Neural Network Fundamentals with Graphs, Algorithms and Applications*. McGraw-Hill Inc., New York, 1996.

[15] G. Carpenter, S. Grossberg, and D. B. Rosen. Fuzzy ART: Fast stable learning and catagorization of analog patterns by an adaptive resonance system. *Neural Networks*, 4:759–771, 1991.

[16] G. A. Carpenter and S. Grossberg. Learning, catagorization, and prediction by fuzzy neural networks. In C. H. Chen, editor, *Fuzzy Logic and Neural Network Handbook*. McGraw-Hill, Inc, New York, 1996.

[17] S. B. Cho. Fuzzy aggregation of modular neural networks with ordered weighted averaging operators. *Approximate Reasoning*, 13:359–375, 1995.

[18] S. B. Cho and J. H. Kim. Combining multiple neural networks by fuzzy integral for robust classification. *IEEE Transactions on System, Man and Cybernetics*, 25(2):380–384, February 1995.

[19] J. J. Choi, R. J. Arabshahi, R. J. Marks, and T. P. Candell. Fuzzy parameter adaptation in neural systems. In *Proceedings of IEEE International Conference on Neural Networks*, pages 232–238, 1992.

[20] J. J. Choi, H. O'Keefe, and P. K. Baruah. Non-linear system diagnosis using neural networks and fuzzy logic. In *Proceedings of IEEE International Conference on Fuzzy Systems (San Diago)*, pages 813–820, 1992.

[21] F. L. Chung and T. Lee. Fuzzy competitive learning. *Neural Networks*, 7(3):539–551, 1994.

[22] J. P. Crutchfield, J. D. Farmer, N. H. Packard, and R. S. Shaw. Chaos. *Scientific American*, pages 38–49, 1986.

[23] G. Deco and D. Obradovic. *An Information-Theoretic Approach to Neural Computing*. Springer, New York, 1996.

[24] M. Dorigo, V. Maniezzo, and A. Colorni. Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on System, Man and Cybernetics*, 26(1):29–41, 1996.

[25] D. Dubois and H. Prade. Rough-fuzzy sets and fuzzy-rough sets. *International Journal of General Systems*, 17(2-3):191–209, 1990.

[26] D. Dubois and H. Prade. Putting rough sets and fuzzy sets together. In R. Slowinski, editor, *Intelligent Decision Support. Handbook of Applications and Advances of the Rough Set Theory*. Kluwer Academic Publishers, Dordrecht, 1992.

[27] J. Eisenberg, W. J. Freeman, and B. Burke. Hardware architecture of a neural network model simulating pattern recognition by the olfactory bulb. *Neural Networks*, 2(4):315–325, 1989.

[28] D. B. Fogel. *System Identification through Simulated Evolution: A Machine Learning Approach of Modeling*. Ginn Press, Needham, MA, 1991.

[29] D. B. Fogel. An introduction to simulated evolutionary optimization. *IEEE Transactions on Neural Networks*, 5(1):3–14, January 1994.

[30] D. B. Fogel. *Evolutionary Computation: Toward a New Philosophy of Machine Learning*. IEEE Press, Piscataway, 1995.

[31] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence through Simulated Evolution*. John Wiley, New York, 1966.

[32] J. A. Freeman and D. M. Skapura. *Neural Networks: Algorithms, Applications and Programming Techniques*. Addison-Wesley, Reading MA, 1991.

[33] J. Gleick. *Chaos: Making a New Science*. Penguin Books, 1988.

[34] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine learning*. Addison Wesley, Reading MA, 1989.

[35] E. Harth. Order and chaos in neural system: An approach to the dynamics of higher brain function. *IEEE Transactions on System, Man and Cybernetics*, 13(5):782–798, September-October 1983.

[36] M. F. Hassoun. *Fundamentals of Artificial Neural Networks.* MIT Press, Cambridge, MA, 1995.
[37] Y. Hayashi. Oscillatory neural networks and learning of continuously transformed patterns. *Neural Networks*, 7:219–231, 1994.
[38] S. Haykin. *Neural Networks - A Comprehensive Foundation.* Macmillan College Publishing Company, New York, 1994.
[39] J. Hertz, A. Krogh, and R. G. Palmer. *Introduction to the Theory of Neural Computation.* Addison-Wesley, Reading, MA, 1991.
[40] K. Hirota and W. Pedrycz. OR/AND neurons in modeling fuzzy set connectives. *IEEE Transactions on Neural Networks*, 2(2):151–161, May 1994.
[41] A. V. Holden. *Chaos.* Manchester University Press, Manchester, 1986.
[42] J. H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, 1975.
[43] C. C. Hsu, D. Gobovic, M. E. Zaghloul, and H. H. Szu. Chaotic neuron models and their VLSI circuit implementation. *IEEE Transactions on Neural Networks*, 7(6):1339–1350, November 1996.
[44] E. R. Hunt. Stabilising high period in a chaotic systems: The diode resonator. *Physical Review Letters*, 67:1953–1955, 1991.
[45] J. E. Hunt and D. E. Cooke. Learning using an artificial immune system. *Journal of Networks and Computer Applications*, 19:189–212, 1996.
[46] D. R. Hush and B. G. Horne. Progress in supervised neural networks. *IEEE Transactions on Signal Processing*, pages 8–39, January 1993.
[47] H. Ishibuchi, R. Fujioka, and H. Tanaka. Neural networks that learn from fuzzy if-then rules. *IEEE Transactions on Fuzzy Systems*, 1(2):85–97, May 1993.
[48] Special issue on Evolutionary Computation. *IEEE Transactions on Neural Networks*, 5(1), January 1994.
[49] Special issue on Fuzzy Logic and Neural Networks. *IEEE Transactions on Neural Networks*, 3(5), September 1992.
[50] Jockusch and H. Ritter. Self-organizing maps: Local competition and evolutionary optimization. *Neural Networks*, 7(8):1229–1239, 1994.
[51] A. Kandle. *Fuzzy Techniques in Pattern Recognition.* Wiley, New York, 1982.
[52] A. Kandle. *Fuzzy Mathematical Techniques with Applications.* Addison-Wesley Publishing Company, Reading, Massachusetts, 1986.
[53] A. Kauffman and M. M. Gupta. *Introduction to Fuzzy Mathematics.* Van Nostrand Reinhold, New York, 1985.
[54] J. M. Keller, P. Gader, H. Tahani, J. H. Chiang, and M. Mohamed. Advances in fuzzy integration for pattern recognition. *Fuzzy Sets and Systems*, (65):273–283, 1994.
[55] J. M. Keller and D. J. Hunt. Incorporating fuzzy membership functions into the perceptron algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 693–699, July/August 1985.
[56] G. S. Klir and B. Yuan. *Fuzzy sets and Fuzzy Logic – Theory and Applications.* Prentice-Hall, Englewood Cliffs, NJ, 1995.
[57] T. Kohonen. *Self Organization and Associative Memory, 3rd edition.* Springer Verlag, Berlin, Germany, 1989.
[58] B. Kosko. *Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence.* Prentice Hall, Englewood Cliffs, NJ, 1992.
[59] B. Kosko. *Fuzzy Thinking.* Harper Collins, Glasgow, UK, 1993.
[60] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT, Press, Cambridge, Massachusetts, 1992.
[61] S. Y. Kung. *Digital Neural Networks.* Prentice Hall, Englewood Cliffs, New Jersey, 1993.
[62] C. T. Lin and C. S. G. Lee. *Neural Fuzzy Systems.* Prentice Hall, Englewood Cliffs, New Jersey, 1996.
[63] C. T. Lin and Y. C. Lu. A neural fuzzy system with linguistic teaching signals. *IEEE Transactions on Fuzzy Systems*, 3(2):169–189, May 1995.
[64] P. Lingras. Rough neural networks. In *Proceedings of the International Conference on Information Processing and Management of Uncertainty, Granada, Spain*, 1996.
[65] P. Lingras. Unsupervised learning using rough kohonen neural network classifiers. In *Proceedings of the CESA'96 Symposium on Modelling, Analysis and Simulation*, 1996.
[66] R. Lippman. An introduction to neural computing. *IEEE ASSP Magazine*, pages 4–22, April 1987.
[67] S. Loncaric and P. Dhawan. Near-optimal mst-based shape description using genetic algorithm. *Pattern Recognition*, 28(4):571–579, 1995.
[68] E. Lorenz. *The Essence of Chaos.* Univ. of Wash. Press, 1993.
[69] B. Mandlebrot. *The Fractal Geometry of Nature.* Freeman, Freeman, 1983.
[70] Z. Michalewicz. *Genetic Algorithm + Data Structure.* Springer Verlag, New York, 1992.
[71] B. Muller and J. Reinhardt. *Neural Networks: An Introduction.* Springer Verlag, New York, 1990.
[72] E. Ott, C. Grebogi, and J. A. Yorke. Controlling chaos. *Physical Review Letters*, 64:1196–1199, 1990.
[73] S. K. Pal and D. Dutta Majumder. *Fuzzy Mathematical Approach to Pattern Recognition.* Wiley (Halsted Press), New York, 1986.
[74] S. K. Pal and S. Mitra. Multilayer perceptron, fuzzy sets and classification. *IEEE Transactions on Neural Networks*, 3(5):683–697, September 1992.
[75] S. K. Pal and N. R. Pal. Soft computing: Goals, tools and feasibility. *Journal of IETE*, 42(5):195–204, July-October 1996.
[76] Y. H. Pao. *Adaptive Pattern Recognition and Neural Networks.* Addison-Wesley, Reading MA, 1989.
[77] T. S. Parker and L. O. Chua. Chaos: A tutorial for engineers. *Proceedings of IEEE*, 75(8):982–1007, August 1987.
[78] Z. Pawlak. Rough sets. *International Journal of Computer and Information Science*, 11:341–356, 1982.
[79] Z. Pawlak. *Rough Sets: Theoretical Aspects of Reasoning About Data.* Kluwer, Dordrecht, 1991.
[80] Z. Pawlak, J. G. Busse, R. Slowinsky, and W. Ziarko. Rough sets. *Communications of the ACM*, 38(11):89–95, November 1995.
[81] Z. Pawlak, S. K. M. Wong, and W. Ziarko. Rough sets: Probabilistic verses deterministic approach. *International Journal of Man-Machine Studies*, 29:81–95, 1988.
[82] W. Pedrycz. Fuzzy neural networks with reference neurons as pattern classifiers. *IEEE Transactions on Neural Networks*, 3(5):770–775, September 1992.
[83] W. Pedrycz and A. F. Rocha. Fuzzy-set based models of neurons and knowledge-based networks. *IEEE Transactions on Fuzzy Systems*, 1(4):254–266, November 1993.
[84] W. Porto, D. B. Fogel, and L. J. Fogel. Alternative neural network training methods. *IEEE Expert*, pages 16–22, June 1995.
[85] J. M. Redners and S. P. Flasse. Hybrid methods using genetic algorithms for global optimization. *IEEE Transactions on System, Man and Cybernetics*, 26(2):243–258, 1996.
[86] B. D. Ripley. *Pattern Recognition and Neural Networks.* Cambridge University Press, Cambridge, 1996.
[87] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and McClelland, editors, *Parallel and Distributed Processing.* MIT Press, Cambridge, MA, 1986.
[88] N. Saravanan and D. B. Fogel. Evolving neural control systems. *IEEE Expert*, pages 23–27, June 1995.
[89] M. Sarkar. Evolutionary programming based fuzzy clustering. In *Proceedings of Fifth Annual Conference on Evolutionary Programming (San Diego)*, pages 247–256, 1996.
[90] M. Sarkar and B. Yegnanarayana. Application of fuzzy integral in modular networks for contract bridge bidding. Accepted in *Fuzzy Sets and Systems.*
[91] M. Sarkar and B. Yegnanarayana. Feedforward neural network classifiers: Backpropagation learning algorithm with fuzzy objective functions. Accepted in *IEEE Transactions on System, Man and Cybernetics.*
[92] M. Sarkar and B. Yegnanarayana. An evolutionary programming based probabilistic neural network construction technique. In *Proceedings of IEEE International Conference on Neural Networks (Houston, USA)*, pages 456–461, June 9-12 1997.
[93] M. Sarkar and B. Yegnanarayana. Feedforward neural networks configuration using evolutionary programming. In *Proceedings of IEEE International Conference on Neural Networks (Houston, USA)*, pages 438–443, June 9-12 1997.
[94] M. Sarkar and B. Yegnanarayana. Incorporation of fuzzy classification properties into backpropagation learning algorithm. In *Proceedings of IEEE International Conference on Fuzzy Systems (Barcelona, Spain)*, pages 1701–1706, July 1-5 1997.
[95] M. Sarkar and B. Yegnanarayana. Rough-fuzzy set theoretic approach to evaluate the importance of input features in classification. In *Proceedings of IEEE International Conference on Neural Networks (Houston, USA)*, pages 1590–1595, June 9-12 1997.
[96] M. Sarkar and B. Yegnanarayana. Fuzzy-rough neural networks for vowel recognition. In *Proceedings of IEEE International Conference on System, Man and Cybernetics (California, USA)*, October, 11-14 1998.
[97] M. Sarkar, B. Yegnanarayana, and D. Khemani. A clustering algorithm using an evolutionary programming-based approach. *Pattern Recognition Letters*, 18(10):975–986, 1997.
[98] H. P. Schwefel. *Numerical Optimization of Computer Models.* John Wiley, Chichester, 1981.
[99] H. Sompolinsky, A. Crisanti, and H. J. Commers. Chaos in random neural networks. *Physical Review Letters*, 61:259–262, 1988.
[100] M. Srinivas and L. M. Patnaik. Adaptive probabilities of crossover and mutation in genetic algorithm. *IEEE Transactions on System, Man and Cybernetics*, 24(4):656–667, 1994.
[101] H. Szu, C. Hsu, and M. Zaghloul. Chaotic neural network architecture. In C. H. Chen, editor, *Fuzzy Logic and Neural Network Handbook.* McGraw-Hill, Inc, New York, 1996.
[102] E. C. K. Tsao, J. C. Bezdek, and N. R. Pal. Fuzzy kohonen clustering algorithm. *Pattern Recognition*, 27(5):757–767, 1994.
[103] M. Vidyasagar. *A Theory of Learning and Generalization with Application to Neural Networks and Control Systems.* Kluwer, Dordrecht, 1996.
[104] L. Wang. Oscillatory and chaotic dynamics in neural networks under varying operating conditions. *IEEE Transactions on Neural Networks*, 7(6):1382–1388, November 1996.
[105] L. X. Wang and J. M. Mendel. Fuzzy basis functions, universal approximations, and orthogonal least square learning. *IEEE Transactions on Neural Networks*, 3(5):807–814, September 1997.
[106] P. D. Wasserman. *Advanced Methods in Neural Computing.* Van Nostrand Reinhold, New York, 1993.
[107] B. A. Whitehead. Genetic evolution of radial basis function coverage using orthogonal niches. *IEEE Transactions on Neural Networks*, 7(6):1525–1528, November 1996.
[108] B. A. Whitehead and T. D. Choate. Cooperative-competitive genetic evolution of radial basis function centers and widths for time series prediction. *IEEE Transactions on Neural Networks*, 7(4):869–880, July 1996.
[109] Z. M. Wojcik and B. E. Wojcik. Intelligence of rough sets inferences vs. statistics, neural nets and fuzzy sets. In *Proceedings of the CESA'96 Symposium on Modelling, Analysis and Simulation*, 1996.
[110] X. Yao. Evolutionary artificial neural networks. *International Journal of Neural Networks*, 4(3):203–222, 1993.
[111] X. Yao and Y. Liu. A new evolutionary system for evolving artificial neural networks. *IEEE Transactions on Neural Networks*, 8(3), May 1997.
[112] X. Yao and Y. Liu. Making use of population information in evolutionary artificial neural networks. *IEEE Transactions on System, Man and Cybernetics*, 28(B2), April 1998.
[113] Y. Yao and W. J. Freeman. Model of biological pattern recognition with spatially chaotic dynamics. *Neural Networks*, 3:153–170, 1990.
[114] B. Yegnanarayana. Artificial neural networks for pattern recognition. *Sadhana*, 19(1):147–169, April 1994.
[115] L. A. Zadeh. Fuzzy sets. *Information and Control*, pages 338–353, 1965.