Adaptive Categorization in Complex Systems

Seyed Shahrestani

School of Computing and Mathematics
University of Western Sydney
Penrith Campus, Locked Bag 1797
PENRITH SOUTH DC NSW 1797
AUSTRALIA
seyed@computer.org

Abstract: A fast and reliable method for categorization of patterns that may be encountered in complex systems is described. Most pattern recognition and classification approaches are founded on discovering the connections and similarities between the members of each class. In this work, a different view of classification is presented. The classification is based on identification of distinctive features of patterns. It will be shown that the members of different classes have different values for some or all of such features. The paper will also show that by making use of the distinctive features and their corresponding values, classification of all patterns, even for complex systems, can be accomplished. The classification process does not rely on any heuristic rules. In this process, patterns are grouped together in such a way that their distinctive features can be explored. Such features are then used for identification purposes.

Key-Words: - Adaptive recognition, Categorization and classification, Distinctive Features, Complex Systems, Feature Extraction, Negative Recognition.

1. Introduction

Many expert systems have been developed for resolving various kinds of problems in complex systems [4] and [16]. In particular, systems for fault identification and diagnosis have received considerable attention. Most of the proposed systems depend on heuristic rules for performing their functions. In many cases, a rule is fired and, if matched, a hypothesis is made in accordance with some hierarchy. The supervisor can then accept or reject the hypothesis. In case of rejection, another rule in accordance with the hierarchy is fired that gives rise to a new hypothesis. The process is repeated until either the correct hypothesis is made or all of the appropriate rules have been fired.

Generally speaking, systems relying on heuristic rules are considered to be fragile. More specifically, for new situation falling outside the rules, they are unable to function and fresh rules have to be generated. Thus, a very large knowledge base must be created and stored for retrieval purposes. In general, heuristic rules are hard to come up with and are always incomplete. The rules are usually inconsistent – no two experts come up with the same set [19].

Alternatively, going to the fundamentals, it can be noted that a pattern can be considered as an extract of information regarding various characteristics or features of an object, state of a system, and the like. The pattern of an object with n features under consideration, is normally represented as an n-dimensional vector, $\mathbf{p}_x$. Classification can then be regarded as as the act of partitioning the feature space into K_1 regions or classes, and identification of

necessary and sufficient conditions that describe membership criteria for each class, C_x. Many methods and surveys of these methods for such adaptive pattern recognitions do exist, for example see [3]-[8].

To increase classification speed and to reduce misclassification error, particularly when K_1 is large or not known some suggestions have been made for grouping of classes and clustering. For example, patterns – on the basis of their similarities – can be mapped into a generalized indicator vector, which is then used in combination with a standard search tree technique for identification purposes [16] and [15]. Another approach, based on distance metric first computes a similarity measure between each pattern and every other pattern, and merges close samples with each other. For each group, its center is then computed. Subsequently, the centers are used to represent the patterns for each group. The process is repeated on centers, until the number of centers stabilizes. The resulting hierarchy and representative patterns are used for identification of new patterns [2] and [14]. Yet another proposed method is to find a pattern prototype – a typical example of some classes – and use that for establishing the category of a new pattern, before comparing it with all other examples of that category to recover its specific identity [1].

Among the areas where adaptive pattern recognition approaches are well justified to be applied, are development of expert systems, fault identification and diagnosis of complex systems [7]. So, for evaluation purposes, the approach proposed here is applied to identification of faults in a power distribution network [12]. In that work, it is shown that after a detailed training, completely successful identification of all faults can be achieved with minimal supervision.

In this work, which is an extension of author's previous works, for instance see [10] and [11], using training sets, distinctive features for all or at least some of the classes are determined. The distinctive features are then used to classify all objects, even for complex systems. These are further explained in the next section. Issues relevant to actual implementation of the proposed approach are discussed in Section 4. Discussion on the strengths and limitations of the proposed approach are presented in Section 5. Tests and discussion on results are the subject of sections 67. This is followed by concluding remarks.

2. Classification and Confining Problem Difficulties

The need for the utilization of artificially intelligent approaches in classification and categorization is already well established. In general, the classification process involves a combination of issues relating to amount of available information and their complexity. In this context various new requirements need to be met by categorization solutions. Some of these requirements are mentioned in this section, while some possible enabling approaches for complying with them are discussed in later parts.

The overwhelming amount of information that is available in most modern system environments requires new approaches to managing problems in such systems. Many problems that may not have been traditionally resolved through classification techniques can benefit from categorization to narrow them down to more manageable sub-problems. AI techniques can help in both the categorization process. As well as resolving the resulting sub-problem.

For instance, consider the help desk environments. Help desk systems are designed to provide customer support through a range of different technology and Information Retrieval (IR) tools play a

fundamental role in this activity. Efficiency and effectiveness in data retrieval being crucial for the overall problem solution process heavily depend on the abstraction models. The abstraction associated with an object should capture all its peculiarities in an easily manageable representation. Identification of relevant features of achieving an object abstraction is a complex task and presence of uncertainties makes this task even harder. For diagnosis purposes, focusing on case-based reasoning (CBR) paradigm, models that capture the relevance and uncertainty of information in a dynamic manner are essential. This is a requirement for models used in both diagnostic knowledge and processes. Based on such models a conversational CBR shell implementing nearest-neighbor (NN) retrieval mechanisms for example can then be utilized to achieve high precision case-retrieval [9].

Distributed applications are evolving towards compositions of modular software components with user interfaces based on web browsers. Each of these components provides well-defined services that interact with other components via network. The increase in the complexity of distribution makes it more difficult to manage the end-to-end Quality-of-Service (QoS). The challenge derives in part from the need for interaction of different management scopes of network and computing domains. A management system deployed to diagnose QoS de-gradation should address two major issues. First, to measure the performance of applications, it needs a low-overhead, scalable system for measuring software components. Second, the performance management system must monitor selected measurements, diagnose QoS degradation, adapt to the environment and integrate with the management systems [6].

3. Categorization and Artificial Intelligence

The interest in building machines and systems with human-like capabilities has lead to considerable research activity and results. Important features of human capabilities that researchers are interested in implementing in artificial systems include learning, adaptability, self-organization, cognition (and recognition), reasoning, planning, decision making, action, and the like. All of which are related to `intelligence. These research activities form the core of Artificial Intelligence (AI) [18]. To achieve higher levels of automation, a number of AI techniques have already been applied to categorization problems. It can be noted that fuzzy logic can be used to improve most other AI approaches, e.g. knowledge presentation in expert systems. Therefore, while this section gives a more elaborate treatment to fuzzy logic, a very brief overview of some other AI approaches that are found to be of value in the classification process is also given.

3.1. Fuzzy Systems

The subject of fuzzy logic is the representation of imprecise descriptions and uncertainties in a logical manner. Many artificial intelligence based systems are mainly dependent on knowledge bases or input/output descriptions of the operation, rather than on deterministic models. Inadequacies in the knowledge base, insufficiency or unreliability of data on the particular object under consideration, or stochastic relations between propositions may lead to uncertainty. In expert systems, lack of consensus among experts can also be considered as uncertainty. Also, humans (operators, experts…) prefer to think and reason qualitatively, which leads to imprecise descriptions, models, and required actions. Zadeh introduced the calculus of fuzzy logic as a means for representing imprecise propositions (in a

natural language) as non-crisp, fuzzy constraints on a variable [21].

A fuzzy system must accept crisp inputs (e.g. measurements) and produce crisp outputs (e.g. control actions), while fuzzy logical operations are used (internally) to reach fuzzy inferences. The representation of real valued (crisp) inputs as fuzzy sets is referred to as fuzzification and is closely related to the concept of Membership Function (MF). A membership function is a numerical representation of the belief about the degree that a fuzzy variable belongs to a fuzzy set A; it is the key concept in fuzzy set theory. The membership function is bounded by [0,1] and the larger it is the stronger is the belief about the degree of membership of a set. If the input space is denoted by Ω with elements $\mathbf{x} \in R^n$, then the fuzzy set A ($\subset \Omega$) is defined as a set of ordered pairs $A=\{\mathbf{x}, \mu_A(\mathbf{x}) | \mathbf{x} \in \Omega\}$; where the membership function μ_A ,maps each element of Ω to a membership value in the range 0 to 1. The center of a fuzzy set is the point at which MF achieves its maximum. Many types of membership functions can be defined (with the sole restriction that it must be bounded by [0,1]). Probably, the most popular choices are B-spline and Gaussian basis functions [15].

It must be noted that the antecedent part of the production rules may have multiple parts consisting of logical operators: fuzzy intersection or conjunction (AND), fuzzy union or disjunction (OR), and fuzzy negation (NOT). By generalizing the similar concepts from Boolean logic, the fuzzy operators can be defined. After applying the logical operations, the antecedent of each rule can be considered to have been satisfied to some degree, referred to as the firing strength. The firing strength is a single number that will also be used to shape the consequent of the rule, the output fuzzy set. This is done by considering the consequent of a rule being true to the same degree of membership as its antecedent. The above implication process is applied to all rules, and all the fuzzy sets that represent the output of each rule are aggregated to form a single fuzzy set.

The defuzzification process maps the resulting (output) fuzzy set B into a crisp output u. Various methods have been nominated; a possibility for the output membership function is based on considering singleton output membership functions. That is, the output MF is considered as a single spike rather than a continuous distribution. With fuzzy sets A_1^i to A_n^i and $\mathbf{x}=[x_1 \ldots x_n]^T$ the (crisp) input, a general (typical) inference based on m fuzzy rules in this model can be described by

IF x_1 is A_1^i … AND x_n is A_n^i THEN $u^i=f^i(x_1,\ldots, x_n)$,
i=1, 2,…, m.

Where $f^i(.)$ is defined crisply. This is a highly efficient approach as the weighted average of (at most) m data points needs to be calculated. μ_A^I represents the

$$u(x) = (\sum_{i=1}^{m} u^i(x) \mu_A^i(x)) / (\sum_{i=1}^{m} \mu_A^i(x))$$

multivariable MF resulting from evaluation of the n conjunction parts in the antecedent of the rule. The linear dependence of each rule on the system input variables combined with the efficient centroid calculation, make this approach ideal for supervisory level construction. That is, for systems that are considered by several models the interpolative capabilities of the described fuzzy system can be efficiently utilized for modeling the overall behavior of the system. At the higher layer, it can be effectively used to supervise and coordinate multiple control actions and tasks that have been designed or identified for different conditions of a system.

3.2. Knowledge Based and Expert Systems

Knowledge Based Systems (KBS) are modular structures in which the knowledge is separate from the inference procedure. Knowledge may be utilized in many forms, e.g. collection of facts, heuristics, common sense, etc. When the knowledge is acquired from (and represents) some particular domain expert, the system is considered to be an expert system. In many cases, knowledge is represented by production rules or specification of the conditions that must be satisfied for the rule to become applicable. Also included are the provisions of what should be done in case a rule is activated. Production rules are IF--THEN statements; a `conclusion' is arrived at, upon the establishment of validity of a `premise' or a number of premises [11].

Rule-based systems are popular in AI because rules are easy to understand and readily testable. Each rule can be considered to be independent of the others, allowing for continual updating and incremental construction of the AI programs. Broadly speaking, systems relying on heuristic rules are considered to be brittle. When a new situation falls outside the rules, they are unable to function and new rules have to be generated. Thus, a very large knowledge base must be created and stored for retrieval purposes. In general, heuristic rules are hard to come up with and are always incomplete. The rules are usually inconsistent; i.e. no two experts come up with the same set [12].

3.3. Artificial neural networks

Artificial Neural Networks (ANNs) are dense parallel layers of simple computational nodes. The strengths of the links between the nodes are defined as connection weights. In most cases, one input layer, one output layer, and two internal (hidden) layers will be considered adequate to solve most problems. This is considered as a Multi-Layer Perceptron (MLP) and is widely popular. The connection weights are usually adapted during the training period by back-propagation of errors, which results in a feed-forward network.

3.4. Pattern recognition

Pattern recognition is the ability to perceive structure in some data; it is one of the aspects common to all AI methods. The raw input data is pre-processed to form a pattern. A pattern is an extract of information regarding various characteristics or features of an object, state of a system, etc. Patterns either implicitly or explicitly contain names and values of features, and if they exist, relationships among features. The entire act of recognition can be carried out in two steps. In the first step a particular manifestation of an object is described in terms of suitably selected features. The second step, which is much easier than the first one, is to define and implement an unambiguous mapping of these features into class--membership space.

Patterns whose feature values are real numbers can be viewed as vectors in n-dimensional space, where n is the number of features in each pattern. With this representation, each pattern corresponds to a point in the n-dimensional metric feature space. In such a space, distance between two points indicates similarities (or differences) of the corresponding two patterns. Partitioning the feature space by any of the many available methods; e.g. maximum likelihood, K-nearest neighbors, decision surfaces and discriminate functions then carry out the actual classification.

3.5. Case-based reasoning

Case-based reasoning (CBR) paradigm starts from the assumption that cognitive process is structured as a cycle. The first step is to gather some knowledge, then the knowledge is used to solve a problem and, depending on the result, one may decide to keep track of the new experience.

Experience is accumulated either by adding new information or by adapting the existing knowledge. The idea is to solve a problem with the existing skills and, at the same time, improving these skills for future use. From the actual implementation point of view, the focus is on how to aggregate and store the information (cases) and how to retrieve them. The solution of a problem depends on the ability of the system to retrieve similar cases for which a solution is already known. The more common retrieval techniques are inductive retrieval and nearest neighbor [8].

4. Knowledge Base Creation and Extraction of Distinctive Features

Clearly, the majority of pattern recognition methods mainly classify them on the basis of similarities among them or their representing patterns. Obviously, patterns representing the same class of objects should have features and some feature values in common. But plausibly, patterns describing members of a different class should have different values for some or all of these features. In other words, objects can be classified as members of a particular class if they have some *distinctive features* making them distinguished from other objects present in the universe of objects. Consequently, it is also reasonable to start categorization on the basis of differences, or through *negative recognition*. That is, place objects or classes which have some evident differences, or distinctive features, from all other objects or classes, into one group. It should be noted that a feature that may be distinctive for a class, among a particular set of classes, is not necessarily distinctive in another set which also includes that particular class.

The approach proposed in this work, is based on identifying features that are distinctive for some objects in the universe of objects being considered. The database containing masks, mask types, training set patterns, and class name and/or number will act as the *knowledge base* for later stages of the procedure. Figure 2, is a summary of how the knowledge base is created.

```
SET mask type to 0
WHILE there is a class whose
mask is not found
    - INCREMENT mask type;
    - SET mask of each class
to distinctive features of that
      class, among all
patterns present in the
training set;
    - RECORD class
numbers/names, masks, mask
types, and patterns
      whose masks are found in
the knowledge base;
    - CONSTRUCT the new
training set, consisting of
patterns whose
      class masks have not been
found;
ENDWHILE.
```

Figure 2: Knowledge base creation

During the new pattern recognition stage, the above mentioned rules are fired in the order of mask types. Among the classes with a particular mask type, only one class has the possibility of being identified as the class for the new pattern. In case that class membership of the pattern is not identifiable using the highest mask type in the knowledge base, and if its class membership can be identified by other means (i.e. supervisor), then the pattern will be included in the training set. With this extended knowledge the masks can be updated, and the pattern is added to the knowledge base. Figure 3 shows a summary of class identification process for new patterns. It can be noted that, knowledge base updating is necessary only if there is a mask change associated with a (new) pattern; which is beneficial in terms

of execution speed and compactness of database.

The main objective of this approach is to find distinctive features, or differences of each class with other classes present in the training set. If these are not evident for all classes in the set, smaller subsets are formed, so that exploration of distinctive features is facilitated.

Let α_i be the set of K_i classes under consideration, and the l^{th} pattern $\mathbf{p}_{al}$ representing class $C_a(\in\alpha_i)$ have n features: $x^l_{a1},x^l_{a2},\ldots,x^l_{an}$, with corresponding values: $v^l_{a1},v^l_{a2},\ldots,v^l_{an}$. If for class C_a, represented by q_a patterns, there are m distinctive features;
For j=1, 2, …, n when the j^{th} feature is distinct:

$$x^{(dist)}_{ak}=x_{aj},\ k=1,2,\ldots,m. \quad (1)$$

These features can be found by considering all patterns representing classes in α_i and using:

for all $t=1,2,\ldots,q_a$ v^t_{aj} does not change;
and for any $l=1,2,\ldots,q_a$
and for each $r=1,2,\ldots,K_i$
where $r\neq a$:

$$v^l_{aj}\neq v^s_{rj} \quad \text{for all } s=1,2,\ldots,q_r \quad (2)$$

This relation states that, the distinctive features are the ones whose values are the same in all patterns belonging to the same class, and their values are different in patterns representing any other class. If conditions (2) are satisfied then the j^{th} feature is distinct for class C_a. An n dimensional *mask* vector, $\mathbf{m}^i_a$ can be defined as:

$$\mathbf{m}^i_a=\{m_1,m_2,\ldots,m_n\}$$

where for $j=1, 2, \ldots, n$:

$m_j=1$ if the j^{th} feature is distinct, and
$m_j=0$ otherwise. (3)

The mask vector $\mathbf{m}^i_a$, along with any pattern $\mathbf{p}_{al}$, carry all the necessary and sufficient conditions for identifying any pattern that can be a member of class C_a. The index i of the mask vector, is to signify that the set α_i has been used in finding it, and it will be called *mask type*.

For the trivial case when only one class C_c is present in α_i, all the features present in the pattern representing this class are considered to be distinct.

$$\mathbf{m}^i_c=\{1,1,\ldots,1\}. \quad (4)$$

For such a case if there are q_c patterns $\mathbf{p}_{c1},\mathbf{p}_{c2},\ldots,\mathbf{p}_{cq}$, representing class C_c, the features of interest will be found using the first part of the conditions in (2).

If there are two classes C_a and C_b present in α_i, each represented by a number of patterns, using (2) and (3), it can be shown that:

$$C_a=C_b\Rightarrow\mathbf{m}^i_a=\mathbf{m}^i_b=\mathbf{0}. \quad (5)$$

$$\mathbf{m}^i_a=\mathbf{m}^i_b\neq\mathbf{0}\Rightarrow C_a\neq C_b. \quad (6)$$

5. Adaptive Recognition and Classification Rules

Each pattern is compared with *all* the rest of the patterns, present in the training set, so that its distinctive features can be identified by relation (2). These features will then serve as *type one masks* for further class identification purposes, relation (3). A mask of type 1 implies that the features signified by the mask are distinctive of this particular class among *all* classes present in the training set. As a classification rule, this means that:

> Any pattern is in the same class as the pattern in the training set, if they have the same value for any feature distinguished by the corresponding mask.

So any class whose mask is found, has been actually described by the necessary and sufficient set of conditions that can distinguish (and describe) other members of the same class.

If a mask of type 1 is not associated with each and every pattern, higher type masks will be found. To find type 2 masks, the patterns representing classes with type 1 masks will be eliminated from the training set, and the same procedure as the one for finding masks of type 1 is repeated on this new smaller training set. The classification rule will be:

> If the pattern is not a member of any class with a type 1 mask, then it is in the same class as the pattern in the training set, if they have the same value for any feature distinguished by the mask (of type 2).

The whole process is repeated until all classes in the training set have a mask of some type associated with them. In other words classification rules for all classes present in the training set have been found.

6. Training and Tests

Although the proposed methodology is rather straight forward, but many different approaches can be taken for its implementation as a code. Finding proper values and types for the masks is the core of this approach. So, the emphasis of the code should be on finding these values reliably and fast. For example given a new pattern, there *will* be some mask changes if the pattern cannot be identified with previously found masks. But even so, the masks *might* retain their previous types. Consequently, execution time may be reduced, if the new masks can be found within the previous grouping of classes. This will be important, for cases where the number of classes and/or patterns is very large, as with general cases in complex systems. Relation (2) shows that the mask for the class that the new pattern is a member of will change. As the new pattern has not been identified, some of the features must have different values from the previous ones. But using the same relation, if the mask types remain as before, to find the new mask, it is just necessary to eliminate (from the mask) the features whose values have changed between the new and any previous pattern in this class. For other classes the masks might change by virtue of the second part of relation (2). Again any pattern in any particular class along with the new pattern can be used for that test, and any feature whose value is the same can be eliminated from the previous mask to arrive at the new mask for this class. Of course if any of the masks become null, a complete mask recalculation is needed, which means some mask types (classes groupings) will need to be changed.

7. Results and Discussions

If the training set, the first one or the ones after some time of code implementation, contains large number of noisy patterns in a given class, it may happen that the mask type for one of the similar classes gets improper assignment. Similar classes will have patterns which will resemble each other very closely. In such cases, if patterns are distinct, one solution may be to eventually take only one pattern from such a class, and keep those other rare patterns, unidentifiable with the resulting mask, in the knowledge base and identify their class membership by a table look up. Another solution may be to combine these classes and associate a mask (and a class number) with both of them, and to identify them by a table look up.

When the patterns are not distinct, actually the same pattern is representing two classes, so even a supervisor will need some other information to identify the class membership of the pattern. The approach will actually eliminate one of these two classes from the α_i to find a mask (of type i) for the other class. Then along with other classes whose masks are found, this class is also eliminated from

the set of classes under consideration, and a mask for the other class is found. This mask will be of a type greater than i. In cases like this, all of the classification rules should be fired. In other words, conflicts should be allowed to rise. Conflicts may arise when the (new) pattern can be identified as the member of two classes with two different mask types. As a single distinctive feature is enough to establish class membership, for distinct patterns, conflicts can be resolved by other features which are distinct between the conflicting classes.

The dynamic status of the system may contain valuable information, and they can easily be implemented in this approach, if the need arises. This will lead to longer patterns and if the only goal is to identify fault locations, as above, this will unnecessarily increase the complexity of the procedure.

The proposed approach and the code based on it can act in a general way to recognition of patterns adaptively. The required knowledge base is compact, the language used is quite general, no heuristic approach is necessary, and the code is fast and reliable. Only two rather simple routines for I/O manipulations are needed for any new job. The input file manager gets the data output of another system. It may also do some data manipulation itself, depending on the nature of the problem. It will then do the proper packing of the features, to get the patterns into their required format, and writes them into an intermediary file, which will in turn be read by the actual code. As the proposed approach conveniently allows it, basic machine operations are the main decision making routines, which results in a fast code. The output file manager gets the result from another intermediary file, and simply converts it into a suitable form for the required action – print out, corrective action, and so forth.

8. Conclusions

The proposed method and the code based on it can act in an adaptive and general manner in recognition of patterns encountered in complex systems. This has clear advantages, as the appropriateness and performance of the method and the related algorithms can be evaluated based on previously appraised data. This will in turn facilitate its implementation and utilization in many other domains. The required knowledge base is compact, the language used is quite general, no heuristic approach is necessary, and the code is fast and reliable. Generalization and specialization are readily achievable, and used for classification purposes. Updating of the knowledge base and machine learning can easily be implemented. In this method, complete rules are found, so in general a single rule will suffice to establish if a pattern is member of a class or not. Given these, it can be expected that the machine will require very little supervision after a thorough training. The method can be of particular interest when the aim is fast classification of a pattern among a large number of classes. The proposed approach appears to be applicable to many classification problems encountered in complex systems that may be solved by artificial intelligence techniques. Although, conceptual clustering is not the goal, but many different concepts may be discovered when the grouping of classes is carried out.The method seems to be applicable to many practical problems that may be solved by artificial intelligence methods.

References:

[1] R. Basri, “Recognition by Prototypes", *Proc. of IEEE International Conf. on Computer Vision and Pattern Recognition*, pp. 161–167, 1993.

[2] Y. Commike & J.J. Hull, “Syntactic Pattern Classification by Branch and Bound Search", *Proc. of IEEE International Conf. on Computer*

Vision and Pattern Recognition, pp. 432–437, 1991.

[3] Y. C. Lee (Editor), *Evolution, Learning and Cognition*, World Scientific Publishing Co., 1988.

[4] S. Liao, "Expert system methodologies and applications—a decade review from 1995 to 2004," *Expert Systems with Applications*, vol 28, pp 93-103, 2005.

[5] H. Long, X. Wang, "Aircraft fuel system diagnostic fault detection through expert system," *Proc. IEEE Sixth World Congress on Intelligent Control and Automation*, WCICA2008, pp. 7104-7107, 2008.

[6] J. Martinka, J. Pruyne and M. Jain "Quality-of-service measurements with model-based management for networked applications," *Technical Report HPL-97-167*, HP Laboratories, Palo Alto, 1998. Available at http://www.hpl.hp.com/techreports/97/HPL-97-167R1.html

[7] G. Mourot, S. Bouschiri, & J. Ragot, "Pattern Recognition for Diagnosis of Technological Systems: A Review", *Proc. of IEEE International Conf. on Systems, Man and Cybernetics*, Vol. 5, pp. 275–281, 1993.

[8] Y. Pao, *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley, 1989.

[9] G. Piccinelli, "Uncertainty modelling in diagnostic systems: An adaptive solution," *Technical Report HPL-98-37*, HP Laboratories, Bristol, 1998. Available at http://www.hpl.hp.com/techreports/98/HPL-98-37.html

[10] S. Shahrestani, H. Yee, and J. Ypsilantis, "Adaptive Recognition by Specialized Grouping of Classes," in *Proc. 4th IEEE Conference on Control Applications*, Albany, NY, 1995, pp. 637-642.

[11] S. Shahrestani, "Exploiting Structure and Concepts for Enhanced Management of Information and Networked Systems," *WSEAS Trans. on Computers*, issue7, vol 4, pp 769-777, 2005.

[12] S. Shahrestani, "Complex System Diagnosis through Adaptive Recognition," to appear in *Proc. 11th International Conf. on Mathematical Methods, Computational Techniques and Intelligent Systems, MAMECTIS09,* Canary Islands, Spain, 2009.

[13] C. W. Therrein, *Decision Estimation and Classification*, John Wiley & Sons, 1989.

[14] M. Todd, S. McArthur, J. McDonald, S. Shaw, "A Semiautomatic Approach to Deriving Turbine Generator Diagnostic Knowledge," *IEEE Trans. Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol 37, Issue 5, pp. 979-992, Sept. 2007.

[15] S. Vaucher, H. Sahraoui, J. Vaucher, "Discovering New Change Patterns in Object-Oriented Systems," *Proc. IEEE International Conf. Reverse Engineering*, pp. 37-41, 2008.

[16] L. Wang, *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*. Englewood Cliffs, USA, 1994.

[17] D. Wilkes & J. K. Tsotsos, "Efficient Serial Memory", *Proc. of IEEE International Conf. on Computer Vision and Pattern Recognition*, pp. 701–702, 1993.

[18] A. Wilson, L. McNamara, and G. Wilson, "Information integration for complex systems," *Reliability Engineering & System Safety*, vol 92, pp 121-130 2007.

[19] P. Winston, *Artificial Intelligence*. Addison-Wesley, USA, 1984.

[20] K. Wong and K. Doan, "Explanation Based Generalization Method for Learning Network Fault Diagnosis

Rules", *Proceedings of IEE International Conference on Advances in Power System Control, Operation, and Management*, vol 2, pp 855-860, 1993.

[21] L. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338-353, 1965.

[22] W. Zhao, X. Bai, W. Wang, J. Ding, "A Novel Alarm Processing and Fault Diagnosis Expert System Based on BNF Rules," *Proc. IEEE International Transmission and Distribution Conf. and Exhibition*, pp. 1-6, 2005.

[23] X. Zheng, Z. Wang, F. Qian, "An Expert System for Real-time Fault Diagnosis and Its Application in PTA Process," *Proc. IEEE Sixth World Congress on Intelligent Control and Automation*, WCICA 2006, pp. 5623-5627, 2006.