Control and Cybernetics

vol. **39** (2010) No. 1

Intelligence in manufacturing systems: the pattern recognition perspective[*]

by

Marek B. Zaremba

Département d'informatique et d'ingénierie
Université du Québec (UQO)
Gatineau, QC J8Y 3G5 Canada
e-mail: zaremba@uqo.ca

Abstract: The field of Intelligent Manufacturing Systems (IMS) has been generally equated with the use of Artificial Intelligence and Computational Intelligence methods and techniques in the design and operation of manufacturing systems. Those methods and techniques are now applied in many different technological domains to deal with such pervasive problems as data imprecision and nonlinear system behavior. The focus in IMS is now shifting to a broader understanding of the intelligent behavior of manufacturing systems. The questions debated by researchers today relate more to what kind and what level of adaptability to instill in the structure and operation of a manufacturing system, with the discussions increasingly gravitating to the issue of system self-organization. This paper explores the changing face of IMS from the perspective of the pattern recognition domain. It presents design criteria for techniques that will allow us to implement manufacturing systems exhibiting adaptive and intelligent behaviour. Examples are given to show how incorporating pattern recognition capabilities can help us build more intelligence and self-organization into the manufacturing systems of the future.

Keywords: Intelligent Manufacturing Systems, pattern recognition, computational intelligence, neural networks, distributed systems, spatial filtering, feature selection, dimensionality reduction, supervised classification.

1. Introduction

What has defined Intelligent Manufacturing Systems (IMS), especially in the initial stages of the emergence of this technology, has been the very fact of employment of methods and techniques of Artificial Intelligence (expert systems)

[*]Submitted: January 2009; Accepted: May 2009.

and Computational Intelligence (predominantly neural networks, fuzzy logic and genetic algorithms) in the design and operation of manufacturing systems. For decades now, the power of those methods and the need to effectively deal with such pervasive problems as data imprecision and nonlinear system behaviour have sustained research on their application in many different technological domains.

Most of the problems involved in the design of manufacturing systems are, by nature, combinatorial and NP-hard. In order to perform combinatorial optimization, various meta-heuristics such as genetic algorithms, simulated annealing, and tabu search, have been extensively studied and reported. Results of a survey of recently published literature on assembly-line balancing, including genetic algorithms, were compiled by Tasan and Tunali (2008). A computer-aided, intelligent programming system based on genetic algorithms (GA) for selection of numerically controlled cutting tools, tool sequence planning and optimization of cutting conditions was designed and described in Balic, Kovacic and Vaupotic (2006). Adaptive Hierarchical Ant Colony Optimization (AHACO) was proposed to resolve the traditional machine loading problem in Flexible Manufacturing Systems (FMS). This problem is formulated in terms of minimizing the system unbalance and maximizing the throughput by considering job sequencing, optional machines and technological constraints (Prakash, Tiwari and Shankar, 2008).

Neural network methods have been of particular interest, and they have been investigated in the context of a large number of applications. The prediction of surface roughness in Electrical Discharge Machining was proposed by Markopulos, Manolakos and Vaxevanidis (2008). A method based on a two-layer dynamic Elman neural network for detecting faults in the assembly of thread-forming screws was presented by Chumakov (2008). The applicability of neural networks for the selection of all possible operations for machining rotationally symmetrical components, by pre-structuring the neural network with prior domain knowledge in the form of heuristic fuzzy-logic rules was investigated by Deb, Ghosh and Paul (2006).

Given that pattern recognition deals with automatic detection of any relations, regularities or structure inherent in data sources, a number of operations encountered in manufacturing system design are closely related to pattern recognition problems. The main tasks in pattern recognition are pattern representation, pattern classification, and reference model learning and adaptation. An example of a direct application of pattern classification using image processing is a case-based evolutionary identification model developed for printed circuit board (PCB) inspection and defect classification (Chang, Chen and Fan, 2008). The segmentation of PCB images is performed in two phases. In the first phase, a set of defect images of several existing basic patterns is stored to form a concept space. In the second phase, a new pattern is generated evolutionally by calculating the relative position of several similar cases in the concept space. The features, required to deliver new patterns so that user requirements are satisfied, are then determined by a case-based reasoning system drawing on past experience within the domain database. Broad areas of the application of pattern recognition are non-destructive testing (Chen, 1999) and diagnosis of manufacturing processes (Ding, Ceglarek and Shi, 2002).

Pattern recognition problems are closely related to data mining technology (Wang, 2007). In many modern manufacturing plants, data that characterize the manufacturing process are electronically collected and stored in the organization's databases. Thus, data mining tools can be used for automatically discovering interesting and useful patterns in the manufacturing processes. The literature presents several studies that examine the implementation of data mining tools in manufacturing (Kusiak and Smith, 2007). The discovered patterns can be subsequently exploited to enhance the whole manufacturing process by improving, for example, product quality. However, data accumulated in manufacturing plants often have unique characteristics, such as unbalanced distribution of the target attribute, and a small training set relative to the number of input features. Thus, conventional methods are inaccurate in quality improvement cases. A feature set decomposition methodology (Rokach and Maimon, 2006), tested on various real-world manufacturing datasets, was shown to be capable of dealing with the manufacturing data characteristics associated with quality improvement.

The process of classification and interpretation of patterns often employs Artificial Intelligence (AI) techniques. This process is involved and was shown beneficial in the design and planning stages of manufacturing. It can assist in solving scheduling and control problems, and can be used in manufacturing integration. A variety of AI techniques and methods can be applied: symbolic representations and expert systems, geometric reasoning, abstraction-based search, intelligent agents, machine learning, etc. Consequently, a question arises concerning selection of the appropriate method for the task at hand. The present paper addresses this issue from the perspective of pattern recognition theory, whereas the comprehension of system intelligence is approached from the standpoint of achieving system goal. The paper casts the problem of designing optimal feature selection and classification systems against the background of the computer science theory that establishes the link between the feature selection and the classification problem on the one hand and cognitive aspects of machine intelligence on the other.

A multi-level framework for the design of Intelligent Manufacturing Systems, presented in the context of AI, is outlined in Section 2. An introduction to the issues associated with the employment of pattern recognition methods is given in Section 3. Section 4 discusses problems associated with initial stages of a pattern recognition system design, i.e., data representation and preprocessing. The issue of feature extraction and selection is presented in Section 5. Section 6 discusses the classification problem in more detail. Architectures pertaining to the design and implementation of intelligent systems are discussed in Section 7.

2. Framework for IMS design

The notion of intelligence in the design and operation of a manufacturing system has been subject to various interpretations, since it relates to the particular comprehension of the concept of machine intelligence. There are several definitions of intelligence. The most important or widely used can be grouped as follows:

1) Implicit definitions based on tests. The tests can explore the capability of a system to generate alternatives and select one of them properly, extract the meaning of a paper, match two sets of interrelated objects, etc.
2) Definitions based on descriptive enumeration of the required properties of an intelligent system. For example, the property of recognizing a scene or constructing a correct response from a perceived situation.
3) Definitions linking intelligence with cognition (Newell, 1990).
4) Pragmatic definitions, stating, for example, that if a system uses fuzzy logic and neural networks, then it is considered intelligent.

In current practice, the pragmatic definition has been predominant in defining Intelligent Manufacturing Systems. We find, and this will be demonstrated further in the paper, that Newell's definition corresponds best to the practice of intelligent system design, particularly in the context of solving pattern recognition problems. The definition postulates that

> *the essence of being an intelligent system is that the system's behavior can be predicted based only on the content of its representations plus its knowledge of its goals.*

Intelligent system operation is technically achieved through implementation of the principles, methods and technologies of AI and Computational Intelligence (CI). Implementation of those technologies in manufacturing systems can be addressed at two levels:

Direct level - relates to the application of AI/CI methods and techniques in manufacturing systems.

Meta-level - relates to the issue of how one can best exploit the capabilities made available by a variety of AI/CI methods and techniques to construct goal-oriented IMS operating in a wide range of environments.

A hierarchical framework introduced by Davedzic and Radovic (1999), primarily for the development of software architectures, is also relevant to IMS design. It defines five levels of abstraction: the primitives level, the components level, the blocks level, the system level, and the integration level. Table 1 depicts some concepts, processes, methods, approaches and tools typical of each of the abstraction levels. The component techniques, such as expert systems, neural networks or genetic optimization, used in the design and development of intelligent behaviour of manufacturing systems can be combined in many ways, depending not only on the technological application, but also on the type of the

Table 1. Components of system intelligence at different abstraction levels

Level	**Level of abstraction**	**Semantics**	**Knowledge representation**	**AI/CI methods**	**Tools**
1	Integration	IMS system with multiple intelligent agents.	Integrated problem solving.	Blackboard-based methods, content-based search.	Integrated development tools.
2	System	IMS sub-system instances defining complex concepts, tasks, and tools for specific application types.	Robotic cells, assembly lines, CIM cells.	Heuristic classification, scene interpretation, knowledge-based planning and scheduling.	Integrated learning tools. Process planning.
3	Blocks	Functional blocks defining knowledge structures and activities for solving common problem classes.	Planners, classifiers, monitors and controllers.	Image classification, learning agents, learning from examples, problem solving strategies.	Special-purpose development tools, knowledge acquisition tools.
4	Components	Learning methods, reasoning mechanisms, knowledge inference.	Neural network models, fuzzy sets, semantic nets, rules sets, frames.	Supervised and unsupervised learning, pattern identification, forward and backward chaining.	Knowledge acquisition tools, development tools (neural networks, genetic algorithms, fuzzy logic).
5	Primitives	Data structures, features, formulas, uncertainty measures, objects.	Image and text data, linguistic terms.	Pattern matching, inheritance, conflict resolution.	Data acquisition and preprocessing tools, text editors.

process in terms of the level of its abstraction. The diversity of the techniques, methods and tools extends toward the higher levels of abstraction.

In relation to manufacturing systems, the components at the lowest levels are generic and can be applied in various IMSs. With an increase in abstraction level, the use of pattern recognition techniques becomes increasingly domain-dependent. The problem-solving approach may be geared more to the needs of an assembly line control, a robotic cell or a Computer-Aided Production and Planning system.

Newell's cognition-based definition of intelligence, unlike other definitions, embraces the meta-level by placing the issues of integration and knowledge organization in the perspective of achieving system goals. It also puts the role of intelligence in engineering systems in the right perspective. Application of AI methods, including pattern recognition and interpretation methods, should:

- focus on problems that otherwise cannot be solved;
- provide a tool for fighting complexity;
- employ cognitive properties of intelligence: generalization, attention focusing, etc.

Typical pattern recognition tasks span all abstraction levels to some extent, but weigh more heavily on the lower levels. The lowest level is associated with template matching tasks. Pattern identification and statistical learning methods operate mostly at the components level. The blocks level incorporates more complex tasks, such as learning techniques included in problem solving strategies. The system level employs heuristic classification methods for the interpretation of an entire scene. The integration level calls for a multi-agent approach to content-based search in integrated problem solving.

3. Pattern recognition problems

Proper design of a pattern recognition system requires a thorough investigation of several specific theoretical and practical issues:

Immense search space: The exponential growth of the search space with a corresponding increase in the number of features makes any direct search method impractical.

Noise-contaminated observations: The training set may contain unreliable or missing data.

Multiple representations: Multiple sources of data as well as multi-level data/knowledge structures applied in a complex recognition system require focusing of attention on subspaces of different resolution and representation.

Multi-modal and non-linear objective function: The objective function may contain many local minima.

Discrete/continuous search space: A complete pattern recognition system usually embodies both discrete and continuous optimization problems.

These issues are normally addressed in the context of two typical pattern recognition problems: supervised and unsupervised. The aim of supervised learning and classification is to build a classifier from a set of pre-classified instances. In unsupervised problem (clustering), the pattern is assigned to a hitherto unknown class. The classes are learned based on the similarity of patterns. The pattern recognition process can be broadly illustrated as in Fig. 1.

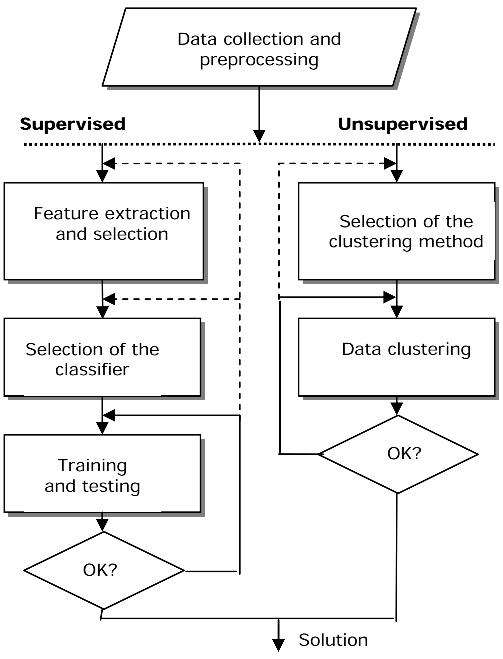

Data collection and preprocessing
Supervised
Unsupervised
Feature extraction and selection
Selection of the clustering method
Selection of the classifier
Data clustering
Training and testing
OK?
OK?
Solution

Figure 1. Structure of the pattern recognition process

The information to solve the supervised problem usually comes in the form of a labeled data set $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$ defined in the n-dimensional feature space $\Re^n$. The feature values for a given object are arranged as a vector $\mathbf{x} = [x_1, \ldots, x_n]^T \in \Re^n$. Let us denote the class label of $\mathbf{z}_j$ by $\lambda(\mathbf{z}_j) \in \Lambda$, $j = 1, \ldots, N$. A classifier is any function:

$$D : \Re^n \rightarrow \Lambda. \tag{1}$$

Class labels are assigned to the classified objects $\mathbf{x}$ using c discriminant functions $G = \{g_1(\mathbf{x}), \ldots, g_c(\mathbf{x})\}$,

$$g_i : \Re^n \rightarrow \Re, \quad i = 1, \ldots, c. \tag{2}$$

If each discriminant function g_i yields a score for the i-th class, and $\mathbf{x}$ is labeled in the class with the highest score, then the classifier

$$D(\mathbf{x}) = \lambda_i \in \Lambda \leftrightarrow g_i(\mathbf{x}) = \max_{i=1,\ldots,c}\{g_i(\mathbf{x})\} \tag{3}$$

follows the maximum membership rule. The discriminant functions partition the feature space $\Re$ into c classification (decision) regions:

$$R_i = \{\mathbf{x}|\mathbf{x} \in \Re^n, g_i(\mathbf{x}) = \max_{l=1,\ldots,c}\{g_l(\mathbf{x})\} \tag{4}$$

enclosed by the classification boundaries. If the classes in the object set $\mathbf{Z}$ can be separated from each other by the classification boundaries in the form of a hyperplane, the classes are called linearly separable.

The unsupervised (clustering) problem explores an unlabeled data set $\mathbf{Z}$ and searches for groups of data similar to one another. A clustering method produces a partition $P = (\mathbf{Z}_1, \ldots, \mathbf{Z}_c)$, $\mathbf{Z}_i \subseteq \mathbf{Z}$. Unsupervised classification algorithms are optimal in cases where detailed knowledge such as ground truth data is not readily available. Based on user-defined parameters, data are iteratively grouped together in clusters. Classes can be determined by distinctions inherent in the data.

The effective use of multiple features and the selection of a suitable classification method are especially significant for improving classification accuracy. There is a variety of classification and learning algorithms. Let us address the issue of feature selection and the selection of the classifier from the machine learning theory perspective, again in the broader context of the purpose and the goal of the application of the particular algorithm.

As shown by Watanabe (1969) in the Ugly Duckling theorem (UDT), *"From the formal point of view there exists no such thing as a class of similar objects in the world, insofar as all predicates (of the same dimension) have the same importance. Conversely, if we acknowledge the empirical existence of classes of similar objects, it means that we are attaching nonuniform importance to various predicates, and that this weighting has an extralogical origin."* In other words, in the absence of assumptions there is no "best" feature representation of the objects classified. If the similarity is judged by the number of predicates the patterns share, then any two patterns are equally similar. A twin theorem, the No Free Lunch (NFL) theorem (Wolpert and Macready, 1995), states that learning and classification algorithms cannot be universally good, i.e., any two algorithms are equivalent when their performance is averaged across all possible problems.

There are many implications of the above two theorems. The results indicate that matching algorithms to problems gives higher average performance than does applying a fixed algorithm to all. Statements indicating that a learning algorithm a_1 is better than a learning algorithm a_2 are ultimately statements about the relevant target functions. The two fundamental theorems imply that there is no problem-independent optimal learning machine. Comparisons between the algorithms should therefore include a sufficiently diverse set of alternative algorithms and should cover a number of real world benchmark data sets. In practical terms, experience with a broad range of techniques is the best insurance for solving arbitrary new pattern classification problems. The UGT and NFL theorems are consistent with Newell's definition of AI given in Section 2 in their emphasis on the goal of the pattern recognition task. They assert that classification is impossible without some bias coming from the knowledge of the goals of the intelligent system under design.

The main components of the pattern recognition process in Fig. 1, i.e., data preprocessing, feature selection, and classification, are addressed in more detail in the next sections.

4. Data representation and preprocessing

Designing a suitable image preprocessing procedure is a prerequisite for successful classification. Some of the crucial problems encountered in practice with processing of sensory data are:

– spatial sparseness and multi-resolution data sources,
– nonlinearity of data,
– high dimensionality of data.

A) Spatial data sparseness

There are several sources of sparseness in spatial data. The data may be acquired in the form of a sparse image due to the discrepancy between the spatial resolution of the sensor and the size and distribution of the objects of interest. The results of unsupervised classification tend to produce salt-and-pepper-like effects. The output of a change-detection algorithm where decisions are made independently at different scales is often in the form of sparse, noisy data. Examples of sparse data are shown in Fig. 2. They range from planar shape images acquired under poor conditions (Fig. 2a) through point cloud data, such as those generated by a LiDAR (Light Detection And Ranging) sensor depicted in Fig. 2b, to the results of the segmentation of terrain data obtained from a Synthetic Aperture Radar (SAR) satellite image (Fig. 2c).

Conventional image processing techniques perform poorly on sparse shapes, due to the anisotropy and noncontiguity of the shape regions. Drawing on the human vision analogy, attention-based analysis, including wavelet operators with such wavelet types as curvelets and ridgelets (Starck, 1998), offers tools for effectively addressing the above issues. In contrast to the standard *isotropic* image analysis, attention operators incorporate multi-scale analysis (essential for processing of data obtained from sources with different resolution) of the anisotropy of objects of interest. In addition, statistical approaches, based on the maximum likelihood strategy, have been reported for directed attention during visual search (Tagare, Toyama and Wang, 2001).

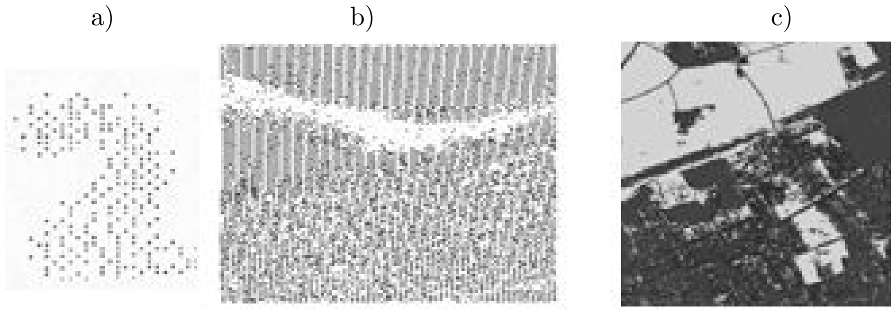


Figure 2. Data sparseness: a) sparse planar shape; b) projection of a cloud of altimetry data; c) result of multichannel SAR classification

A probabilistic Bayesian approach, called the relevance function approach (Palenichka, Zaremba and Missaoui, 2006), can be successfully used in various situations to effectively deal with multi-scale and sparse data. The point that is located at the centre of an object of interest and corresponds to the maximal value of a certain likelihood function allows for an optimal location of the object of interest. The relevance function is an image local operator (a non-linear spatial filter) that has local maximal values at centres of location of the objects of interest or their salient parts. The operator typically processes data obtained by an image transformation that involves extraction of an image property relevant to the determined objects. The result of such a transformation, $f(i,j)$, is called object *property map*. This approach works especially well if the image conforms to the underlying model, i.e., when the image intensity and the object shape satisfy certain explicit constraints.

An implementation of the relevance function, the Multi-Scale Isotropic Matched Filter (MIMF) (Eq. 5) accounts for four saliency conditions (tokens) of a sparse image: contrast between the object O and the background B, local non-homogeneity of the scene $f(i,j)$, radial symmetry, and scale S. The spatial scale is a measure of the size of the filtered region, expressed in the number of pixels. In general, the MIMF is computed within the given scale range $(S_{\max}, S_{\min})$ at a point (i,j) as:

$$MIMF\{f(i,j), S_k\} = c(i,j,S_k) - \alpha \cdot d(i,j,S_k) + \beta \cdot s(i,j,S_k) + \gamma \cdot p(i,j,S_k) \quad (5)$$

where:

$c(i,j,S_k)$ is an estimate of the symmetric local contrast at point (i,j) and the current scale S_k,

$d(i,j,S_k)$ is an estimate of the object homogeneity, defined as the mean intensity deviation in the object region,

$s(i,j,S_k)$ is the difference between the current scale and the minimal possible scale $S_{\min}$,

$p(i,j,S_k)$ is the object shape symmetry (compactness), and

α, β, γ are weight coefficients relative to the first saliency token.

In general, any object-relevant feature can be used as an object property token. Equation (6) provides an example of a more detailed form of (5) defined for a discrete (raster) property map, and for two tokens: contrast and homogeneity.

$$M\{f(i,j)\} = \left(\frac{1}{|O|}\sum_{(m,n)\in O(i,j)} f(m,n) - \frac{1}{|B|}\sum_{(m,n)\in B(i,j)} f(m,n)\right)^2 - \gamma \cdot \left(a - \frac{1}{|O|}\sum_{(m,n)\in O(i,j)} f(m,n)\right)^2 \qquad (6)$$

Fig. 3 illustrates the topological relationship between the object O and background B regions used to calculate (6).

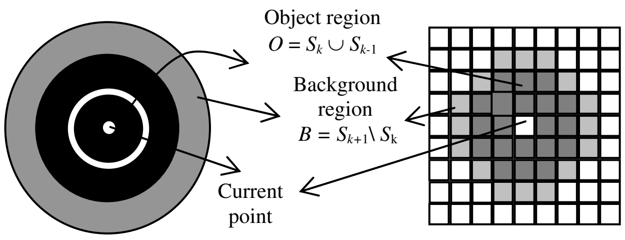


Figure 3. Object and background regions

Detection of features of interest in an image is optimal at a certain scale (Jähne, 2002). So, efficient processing of an image requires processing at different scales. Typically, multi-scale methods are based on multi-grid representations, such as Gaussian or Laplacian pyramids, or wavelets. In the MIMF method, the scale can be estimated for a given image on a single grid in an adaptive way by maximizing local contrast with a homogeneity constraint (Palenichka and Zaremba, 2007):

$$\hat{S}(i,j) = \underset{0\le k\le N-1}{\arg\max}\,\{c(i,j,S_k) - \alpha\, d(i,j,S_k)\} \qquad (7)$$

where N is the total number of scales in the given scale range. Fast recursive algorithms have been developed to time-efficiently implement the formula.

Although particularly efficient for filtering spatially sparse data, the MIMF method also offers several advantages in detecting objects of interest invariant to size and orientation changes, through its use of the multi-scale model-based approach and optimal decision making, since the approach is based on the maximum likelihood rule. Figs. 4 and 5 illustrate two examples of the application of the MIMF method: for image representations for content retrieval (Fig. 4) and for the diagnosis of the quality of photodetector surfaces (Fig. 5).

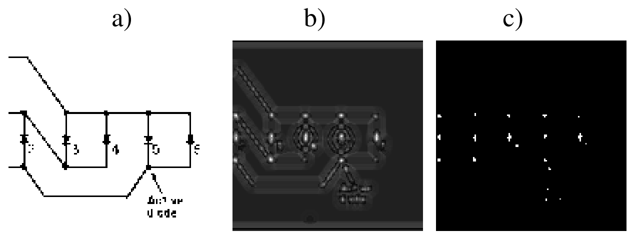


Figure 4. Image content representation by a set of feature vectors in salient locations: (a) original image; (b) property map calculated over the whole scale range; (c) detected salient locations

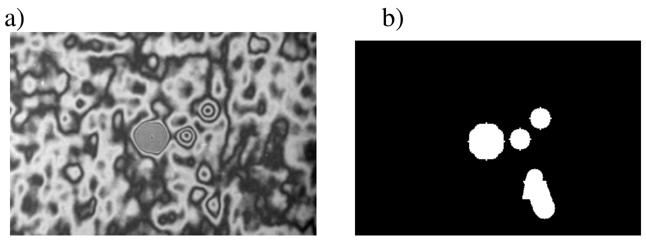


Figure 5. Defect detection in photodetector surfaces. a) original image of a photoreceptor surface; b) defect regions

B) Nonlinear data relationships

Linear patterns can be detected efficiently by classical techniques such as least square regression. The use of those well-known tools for discovering nonlinear patterns requires mapping of the data into a suitable feature space. Mapping of nonlinear relations into linear ones without the explicit computation of feature mapping is made feasible through the employment of kernels (Shawe-Taylor and Cristianini, 2004). Many linear parametric models for classification, regression and novelty detection can be reformulated in terms of a dual representation in which the predictions are based on linear combinations of a kernel function:

$$K(\mathbf{x}, \mathbf{x'}) =< \phi(\mathbf{x}), \phi(\mathbf{x'} >, \tag{8}$$

where ϕ is a mapping from X to a feature space F for all $\mathbf{x}, \mathbf{x'} \in X$.

Kernel functions perform mapping from an initial nonlinear space to an inner product, possibly high-dimensional, feature space. This technique of kernel substitution (the so-called kernel trick) can be applied in the development of nonlinear variants of several methods used in pattern recognition, such as Principal Component Analysis, nearest-neighbour classifier or Fisher discriminant analysis. The kernel concept (Aizerman, Braverman and Rozonoer, 1964) was re-introduced into machine learning in the context of large-margin classifiers (Vapnik, 1998) and Support Vector Machines (SVM). Since then, there has been considerable research interest in kernel methods, both in terms of theoretical investigations and practical applications (Bishop, 2006).

The advantage of using a kernel function is that, since the number of tuneable parameters no longer depends on the number of attributes used, mapping to high-dimensional feature space does not increase the number of these parameters. This provides a solution to the curse of dimensionality problem. Different kernel functions can be used:

$$K(\mathbf{x},\mathbf{x'}) = (<\mathbf{x},\mathbf{x'}> +R)^d \quad \text{— polynomial kernel}$$
$$K(\mathbf{x},\mathbf{x'}) = exp(- \mid \mathbf{x}-\mathbf{x'} \mid^2 /2\sigma^2) \quad \text{— Radial Basis Function kernel}$$
$$K(\mathbf{x},\mathbf{x'}) = \prod_{i=1}^{n}(1+a_i x_i x'_i) \quad \text{— all-subsets kernel}$$
$$K(\mathbf{x},\mathbf{x'}) = \tanh(a\mathbf{x}^T\mathbf{x}'+b) \quad \text{— sigmoidal kernel,}$$

and other that satisfy Mercer's condition (Schölkopf and Smola, 2002). Selection of the best kernel for a particular problem is a question that arises in many applications. In the context of SVM theory, a means of comparing different kernels is the evaluation of the upper bound of the Vapnik-Chervonenkis (VC) dimension (Hastie, Tibshirani and Friedman, 2001). The VC dimension of the class of functions $\{f(x,\alpha)\}$, indexed by a parameter vector α, is defined to be the largest number of points that can be *shattered* by members of $\{f(x,\alpha)\}$. A set of points is said to be shattered by a class of functions if, no matter how we assign a binary label to each point, a member of the class can perfectly separate them. The VC approach fits a sequence of models (in this case, kernel functions) of increasing VC dimensions, and then chooses the model with the smallest value of the upper bound. Recent research has focused on developing efficient, specialized kernel optimization algorithms (Tsang and Kwok, 2006; Zien and Ong, 2007).

A method that uses kernel functions consists of two phases: kernel mapping into the feature space and a learning phase that aims at discovering *linear* patterns in that space. The kernel approach is modular and flexible. What is of practical importance, it has been extended to handle strings of symbols (including text), trees, and general structured data.

C) Immense search space and data complexity

A defining feature of a large class of information systems in manufacturing, especially in multi-sensor systems, is the high complexity of data, which influences the performance of the classification techniques that use the data. Such techniques as fusion of multi-sensory or multi-resolution data, the use of multi-temporal data or image transforms data used for improving classification accuracy contribute to the rapid increase of data dimensionality in a variety of problems. The curse of dimensionality is a well-known but not entirely well-understood phenomenon. The relationship between the expected classification accuracy and the number of training samples and the measurement complexity was initially investigated in Hughes (1968). The results show that for a fixed number of training samples there is an optimal measurement complexity. The analysis process applied in low-dimensional spaces is in most cases not appropriate in spaces with higher dimensionality. A problem with distance metrics in a high-dimensional space is that distance is typically measured across volume. Volume increases exponentially as dimensionality increases, and points tend to become equidistant. A closely related problem is the issue of the high dimensionality of data, regarded in more general terms as the dimensionality of the feature space, which is addressed in the next section.

5. Feature extraction and selection

Classifiers can be built on different sets of features, the number of which can be very large, particularly in such applications as machine vision or multi-sensor diagnostic systems. In order to improve the generalization capability, smaller sets of features are generated from the original input variables. Several approaches and methods have been developed to reduce feature dimensionality. They can be broadly categorized as feature extraction and feature selection. Feature extraction implies generation of new features as a function of the original inputs, whereas feature selection techniques aim at improving the performance of a pattern recognition system by discarding bad or irrelevant features from the available set of features.

Table 2 compares typical dimensionality reduction methods (Zaremba, 2008) that apply to the feature extraction approach: Principal Component Analysis (Cooley and Lohnes, 1971), Self-Organizing Maps (Kohonen et al., 1996), ISOMAP (Tenenbaum, de Silva and Langford, 2000), and Locally Linear Embedding (Roweis and Soul, 2000). It should be noted that the extracted features lose the physical meaning of their original counterparts, which can be disadvantageous in certain applications.

A widely used feature selection procedure consists in the forward or backward selection method using a selection criterion. In *backward selection*, we start from a large set of features and consecutively delete those, which impair the selection criterion the least. In *forward selection*, we start from a minimum set

Table 2. Feature extraction methods for dimensionality reduction

Method	**Principle**	**Type of reduction**	**Optimization criterion**	**Limitations/Notes**
Principal Component Analysis (PCA)	Second-order statistics.	Linear decomposition to covariance eigenvectors.	Axes maximize the variance of orthogonal projections.	Not efficient when the data are poorly correlated.
Self-Organizing Maps (SOM)	Neural network (non-supervised learning).	Projection of data on a map (usually 2-D or 3-D).	Maximum number of iterations.	Dimensions of the map selected *a priori*.
ISOMAP	Geodesic distances incorporated with metric multidimensional scaling.	Projection on nonlinear hyperplanes.	Minimum geodesic distance on a neighborhood graph.	Dimensionality results from the algorithm.
Locally Linear Embedding (LLE)	Maintains local linearity.	Computation of neighborhood-preserving embeddings of high-dimensional inputs.	Minimum reconstruction error.	Inputs mapped into a single global coordinate system. Dimensions selected *a priori*.

of features and consecutively add those, which improve the selection criterion the most. Since the selection criteria usually apply local optimization techniques, global optimality of feature selection is not guaranteed. Backward selection tends to be slower but more stable in selecting optimal features than forward selection. Many feature selection algorithms use CI techniques: neural networks (Verikas and Bacauskiene, 2002; Liu et al., 2004), genetic algorithms (Oh, et al., 2004; Zhang, Verma and Kumar, 2005), fuzzy logic (Bhatt and Gopal, 2005), with applications ranging from diagnosis (Jack and Nandi, 2002) to target discrimination in Synthetic Aperture Radar images (Bhanu and Lin, 2003). Specialized methods have also been developed to deal with feature selection for very large dimensional data sets (Liu and Lu, 2005).

An important practical consideration is that in the classification problem, the complexity of feature selection is closely related to the degree of imbalance of the learning set and our a priori knowledge of the distribution of $(\mathbf{z}|\mathbf{y})$. With an unbalanced number of positive versus negative examples in the learning set, significant evidence is required to demonstrate that a feature is not meaningful. If the learning set is balanced, features are more easily selected. A framework to overcome high dimensionality in unbalanced problems was proposed by Evangelista, Embrechts and Szymanski (2006). The framework explores subspaces of the data, training a separate model for each subspace, and then fusing the decision variables produced by the test data for each subspace.

6. Pattern classification and learning

A great amount of research has been directed toward developing advanced classification approaches and techniques for improving classification accuracy. Many classification approaches, such as artificial neural networks, discriminant analysis, tree classifiers and expert systems, have been widely applied. Fig. 6 provides a taxonomy of classification methods including representative examples for each type of method.

We can distinguish two major groups of methods. The first group assumes true prior probabilities, $P(\lambda_i)$, for a set of class labels $\Lambda = \{\lambda_i\}$, $i = 1, \ldots, c$, where c denotes the number of classes, as well as class-conditional probability density functions (pdf), $p(\mathbf{x}|\lambda_i)$, estimated from the data set. In general terms, Bayes decision theory is applied when class-conditional densities are known. Both parametric and non-parametric methods can be used. If the classes are normally distributed, then either LDC or QDC is the optimal classifier. The parameters (e.g., mean vector and covariance matrix) are often generated from training samples. In nonparametric designs, $p(\mathbf{x}|\lambda_i)$ has to be estimated in the vicinity of $\mathbf{x}$ in a certain region $R \subset \Re^n$. In order to derive the multi-modal classifier, the feature space is divided into cells (bins), and the probabilities are calculated based on the number of points contained in the cells which belong to different classes. Group A (Fig. 6) classifiers may often produce noisy results. Another major drawback is that it is difficult to integrate ancillary data, spatial

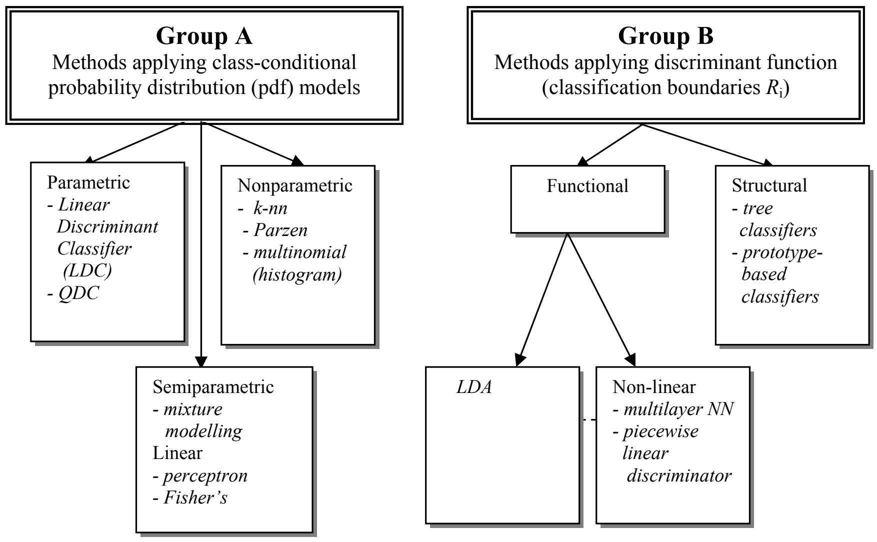

Group A
Methods applying class-conditional probability distribution (pdf) models
Parametric
- Linear Discriminant Classifier (LDC)
- QDC
Nonparametric
- k-nn
- Parzen
- multinomial (histogram)
Semiparametric
- mixture modelling
Linear
- perceptron
- Fisher's
Group B
Methods applying discriminant function (classification boundaries R_i)
Functional
Structural
- tree classifiers
- prototype-based classifiers
LDA
Non-linear
- multilayer NN
- piecewise linear discriminator

Figure 6. Classification methods

and contextual attributes, and non-statistical information into a classification procedure. Those types of data and information often have to be handled in engineering systems.

There is a variety of classification and learning methods applying discriminant functions algorithms: generalized linear discriminators, neural networks, fuzzy (soft) classifiers (Bezdek et al., 1999), and Support Vector Machines, SVM (Abe, 2005), just to name a few. Their detailed analysis and comparison is beyond the scope of this paper. Let us just mention the most typical methods. A broad class of classifiers in this group is constructed around neural network architectures. The main characteristics of neural networks are that they have the ability to learn complex nonlinear input-output relationships, use sequential training procedures, and adapt themselves to the data. SVM are generally recognized as powerful tools for various machine learning problems, performing structural risk minimization, which minimizes an upper bound on the generalization error in contrast to empirical risk minimization, employed by neural networks, which minimizes the error on the training data. This is obtained by mapping the training data into a higher-dimensional feature space via kernel mapping (see Section 4B), and constructing a separating hyperplane with a maximum margin. One potential disadvantage of the SVM is that the resulting classifier is often not as compact as other classifiers. Solutions have been proposed in the form of sparse large margin classifiers or the reduced set method. In order to obtain both a good kernel and a compact representation of the SVM, integrating multiple-kernel learning with the sparse kernel SVM was presented in Hu, Chen and Kwok (2009). Tree-based methods are conceptually simple yet powerful. They divide the feature space into a set of nodes, and then classify the observations in each node into the majority class. A key advantage of the tree-based methods is their interpretability, a characteristic important in diagnostic systems. A similar hierarchical method, the object-based structural classification (Repaka and Truax, 2004), attempts to describe classes in terms of several categories of characteristics, each of which may be assigned weighting factors. Class-related features involve a connection to nearby objects, e.g., super- and sub-objects in a hierarchy. Standard segmentation can be further augmented by knowledge-based partitioning and the construction of sub-objects for special classification tasks. Segmented sets of one class can be merged at the same level or grouped beneath a new, higher level. Relationship-based classification is possible as each object is aware of its neighbor, and of sub- and super-objects.

Many factors, such as different sources of the data and variations in their spatial resolution, the classification system, and the availability of classification software must be taken into account when selecting a classification method for practical use. Each classification method has its own merits, and different classification results may be obtained depending on the classifier(s) chosen. Full knowledge required to properly assess the information needed for optimal classification or learning is rarely, if at all, available. What insight can we get in order to find criteria that can guide us in selecting the best algorithm? Let us first explore the nature of the generalization error, i.e., the prediction error of the selected trained model on new data, in the context of bias-variance decomposition. Achieving good generalization requires finding the right balance between the estimation error (low bias) and the approximation error (low variance). In general, bias is associated with underfitting the data, that is, the classifier cannot match well the optimal model. Variance, in turn, is associated with overfitting, that is, the performance of the classifier depends on the data set drawn from the distribution of the problem. Typically, the more complex the model, the lower the bias obtained, but at the cost of higher variance. A widely used method for estimating the prediction error is cross-validation. However, cross-validation yields meaningful results if the validation set and test set are drawn from the same population. In many applications, the structure of the system being studied evolves over time. This can introduce systematic differences between the training and validation sets. More accurate classification decision can be obtained by designing multiple classifier systems. There are two main strategies in combining classifiers: fusion and selection (Kuncheva, 2004). In the first one, each component classifier has access to the whole feature space. In classifier selection, each classifier employs a part of the feature space and is responsible for objects in this part.

As far as the training of the multiple classifier systems is concerned, there are three approaches:

a) The system does not need training, once the individual classifiers have been trained;

b) Additional training is performed at the classifier combiner level;

c) The combiner is developed during the training of the individual classifiers.

In the first approach, a voting scheme applied on classifier outputs or prior probabilities for individual classes (e.g., naïve Bayes combination) is frequently used to obtain the final classification decision. The second approach typically uses the decision profile $DP(\mathbf{x})$ obtained from L classifier outputs. If we denote by $d_{i,j}(\mathbf{x})$ the support that classifier D_i gives the hypothesis that $\mathbf{x}$ comes from class c_j, the decision profile can be presented as matrix:

$$DP(\mathbf{x}) = \begin{bmatrix} d_{1,1}(\mathbf{x}) & ... & d_{1,j}(\mathbf{x}) & ... & d_{1,c}(\mathbf{x}) \\ d_{i,1}(\mathbf{x}) & ... & d_{i,j}(\mathbf{x}) & ... & d_{i,c}(\mathbf{x}) \\ d_{L,1}(\mathbf{x}) & ... & d_{L,j}(\mathbf{x}) & ... & d_{L,c}(\mathbf{x}) \end{bmatrix} \tag{9}$$

where rows correspond to outputs of classifier $D_i(\mathbf{x})$, and columns to the classifier support for class c_j. The classifier combiner calculates the overall support for each class c_j, finds the class with the largest support, and labels the input accordingly. With sufficient amount of data available, the designer can make use of the whole $DP(\mathbf{x})$, i.e. of all the $L \times c$ degrees of support, rather then using the L supports for each class separately. Methods such as Decision Templates (DT) or Dempster-Shafer theory can be used for that purpose. In the third approach, a powerful method that enhances the classification process by aggregating weak classifiers into a strong combination of models (the so-called committees) is boosting. The most widely used boosting algorithm is AdaBoost (Freund and Shapire, 1997), short for "adaptive boosting". The interpretation of boosting as the sequential optimization of an additive model under an exponential error opens the door to a variety of boosting-like algorithms by altering the form of the error function, including multiclass and regression problems (Friedman, 2001).

7. System integration

An essential feature of a majority of modern IMSs that has to be accounted for during their design and development is the distributed nature of their operation, both in terms of the geographical distances and the computing architectures. An example of a large-scale, geographically and functionally distributed architecture is the Technoinfra (Technological Information Infrastructure) architecture for virtual enterprises investigated in the VIPNET IMS project (Seki, 2004) and shown in Fig. 7. The internal knowledge representation model used in the project was the General Process Model (GPM), a semantic network model developed in Japan (Yoon et al., 2002). The GPM class library was further investigated for the design of a nuclear power plant (Cho and Park, 2005). The GPM class library is composed of two object libraries: the class object library, and the association object library. Two types of class objects are defined, in order to represent geometry (*Location_base_definition*, *Topology*, etc.) and non-geometry product data. Relationships between classes in the GPM class library are represented by several association objects, such as *is_assembled_from*, *is_classified_as*, *is_placed_on*, *possesses*, *is_represented_by*, *has_property_of*, *is_qualified_as*, and *is_connected_to*.

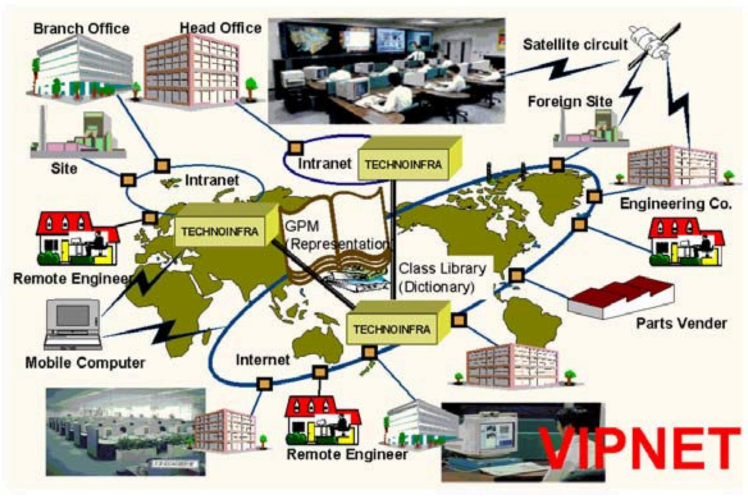
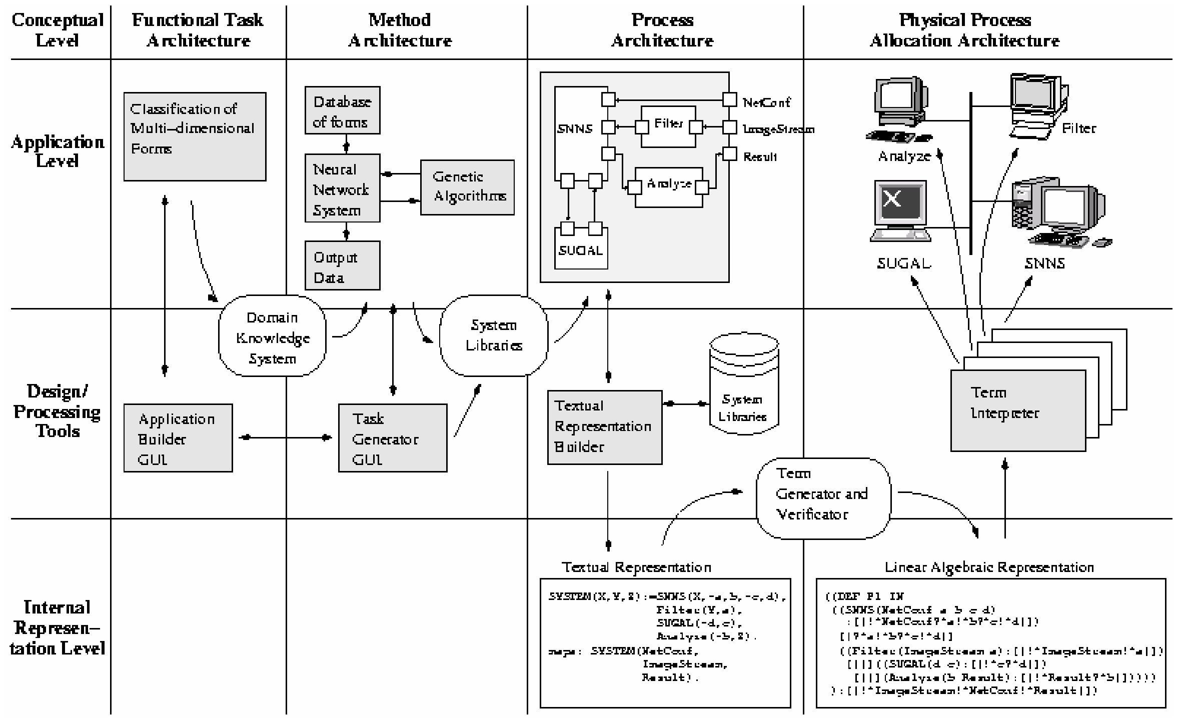


Figure 7. Distributed architecture of the Visual Production Enterprise Network (VIPNET) using Technoinfra

Distributed manufacturing architectures deal with two categories of industry data: geography-related data and resource data. In the case of IMS, the resource data of interest are the resources related to the intellectual capacity of the system. An example of the design process, which illustrates the main phases of creating and implementing a concurrent task system, as well as the architecture of the programming environment geared to a wide-area distributed system, is illustrated in Fig. 8.

Different stages in the process of design and implementation of a system of concurrently running tasks are shown using the example of a heterogeneous form recognition and classification application incorporating CI procedures. The Functional Task Architecture defines the tasks at the higher functional level, such as classification or optimization. This architecture is decomposed and defined in terms of specific, predominantly machine intelligence methods (neural network, fuzzy logic, SVM, k-means classification, etc.), at the Method Architecture Level. The rows depict different levels of the generalization of the design



```
SYSTEM(X,Y,Z):=SNNS(X,-a,b,-c,d),
               Filter(Y,a),
               SUGAL(-d,c),
               Analyze(-b,Z).
maps: SYSTEM(NetConf,
             ImageStream,
             Result).
```

```
((DEF P1 IN
 ((SNNS(NetConf a b c d)
   :[|!^NetConf?^a!^b?^c!^d|])
  [|?^a!^b?^c!^d|]
  ((Filter(ImageStream a):[|!^ImageStream!^a|])
   [||]((SUGAL(d c):[|!^c?^d|])
    [||](Analyze(b Result):[|!^Result?^b|]))))
 ):[|!^ImageStream!^NetConf!^Result|])
```

Figure 8. Architecture of a distributed system design process

system and tools. The columns correspond to successive stages of system design and implementation, from the definition of the functional task architecture to the physical process allocation architecture.

The essential capability that distinguishes this technological approach from other types of communication schemes is the inherent support for entirely dynamic communications, where concurrent agents can establish multi-channel, multi-directional communication without a priori restrictions on their location (Zaremba and Fraczak, 2001). The support for this intelligent process communication and synchronization is implemented as the Internal Representation Level applying the mechanisms of process algebra (Bergstra, Ponse and Smolka, 2001).

The illustrated design methodology and the tool set were developed to implement the design of distributed, heterogeneous intelligent systems. The task architecture incorporates processes that execute both general-purpose tasks requiring the use of a communication library and dedicated programs with built-in communication support.

8. Conclusions

The issue of intelligence embedded in IMS design and operation was addressed from the standpoint of pattern recognition methodology. It was argued that it is not sufficient that an IMS be defined simply as a system designed or operated using an Artificial Intelligence technique. An essential component of the design of an intelligent system should be a link with cognition, given that – from the perspective advocated in this paper – system behaviour can be predicted based on knowledge of its goals and content of its representations. All key elements of intelligence: the ability to predict, the ability to adapt, and the ability to take appropriate action require the detection and interpretation of patterns. A hierarchical model for intelligent system architecture with components at five levels of abstraction was adopted as a framework for IMS design. Pattern recognition techniques are situated mainly at the level of components and blocks. Criteria for selection of these techniques come from the system and integration levels.

The main components of the pattern recognition process, i.e., data preprocessing, feature selection, and classification are addressed in more detail in the paper. The specific issues concerning the preliminary stages in the design of a pattern recognition system for manufacturing systems were identified as high dimensionality and nonlinearity of the feature space, and the sparseness of data. The MIMF method, which provides an efficient tool for isotropic analysis of sparse data at different scales, was presented in more detail. Selection of both features and classification method was cast against the theoretical background formalized by the UDT and NFL theorems. The importance of knowledge about the goal of the system is particularly evident in the problem of classifier selection. Methods for increasing classification accuracy by combining pattern classifiers were discussed. Finally, an example of the design of a distributed intelligence system was provided.

References

Abe, S. (2005) *Support Vector Machines for Pattern Classification.* Springer-Verlag, London.

Aizerman, M.A., Braverman, E.M. and Rozonoer, L.I. (1964) The probability problem of pattern recognition learning and the method of potential functions. *Automation and Remote Control* **25**, 1175-1190.

Alt, H., Mehlhorn, H., Wegener, H. and Welzl, E. (1988) Congruence, similarity and symmetries of geometric objects. *Discrete Comput. Geometry* **3**, 237-256,

Balic, J., Kovacic, M. and Vaupotic, B. (2006) Intelligent programming of CNC turning operations using genetic algorithm. *J. Intelligent Manuf.* **17**, 331-340.

Bergstra, J.A., Ponse, A. and Smolka, S.A., eds. (2001) *Handbook of Process Algebra.* North-Holland, Amsterdam.

Bezdek, J.C, Keller, J., Krisnapuram, R. and Pal, N.R. (1999) *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing.* Kluwer Academic Publishers, Norwell, MA.

Bhanu, B. and Lin, Y. (2003) Genetic algorithm based feature selection for target detection in SAR images. *Image and Vision Computing* **21** (7), 591-608.

Bhatt, R.B. and Gopal, M. (2005) On fuzzy-rough sets approach to feature selection. *Pattern Recognition Letters* **26** (7), 965-975.

Bishop, C.M. (2006) *Pattern Recognition and Machine Learning.* Springer Science, Singapore.

Chang, P.C., Chen, L.Y. and Fan, C.Y. (2008) A case-based evolutionary model for defect classification of printed circuit board images. *J. Intelligent Manufacturing* **19** (2), 203-214.

Chen, C.H. (1999) Pattern recognition in non-destructive evaluation of materials. In: *Handbook of Pattern Recognition & Computer Vision.* World Scientific, Hackensack, NJ, 455-472.

Chumakov, R. (2008) An artificial neural network for fault detection in the assembly of thread-forming screws. *J. Intelligent Manufacturing* **19**(3), 327-333.

Cho, K.J. and Park, C.C. (2005) Implementation of the KNGR class library based on the GPM and semantic networks for co-design. *Computer-Aided Design & Applications* **2** (1-4), 165-172.

Cooley, W.W. and Lohnes, P.R. (1971) *Multivariate Data Analysis.* John Wiley & Sons, Inc., New York.

Deb, S., Ghosh, K. and Paul, S. (2006) A neural network based methodology for machining operations selection in Computer-Aided Process Planning for rotationally symmetrical parts. *J. Intelligent Manufacturing* **17** (5), 557-569.

Devedzic, V. and Radovic, D. (1999) A framework for building Intelligent Manufacturing Systems. *IEEE Trans. Systems, Man, and Cybernetics – Part C* **29** (3), 422-439.

Ding, Y., Ceglarek, D. and Shi, J. (2002) Fault Diagnosis of Multistage Manufacturing Processes by Using State Space Approach. *J. of Manufacturing Science and Engineering* **124** (2), 313-322.

Evangelista, P.F., Embrechts, M.J. and Szymanski, B.K. (2006) Taming the curse of dimensionality in kernels and novelty detection. In: A. Abraham et al., eds., *Applied Soft Computing Technologies: The Challenge of Complexity.* Springer Verlag, Berlin.

Freund, Y. and Schapire, R.E. (1997) A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* **55** (1), 119-139.

Friedman, J.H. (2001) Greedy function approximation: A gradient boosting machine. *Annals of Statistics.* **29** (5), 1189-1232.

Hastie, T., Tibshirani, R. and Friedman, J. (2001) *The Elements of Statistical Learning.* Springer-Verlag, New York.

Hu, M., Chen, Y. and Kwok, J.T.Y. (2009) Building sparse multiple-kernel SVM classifiers. *IEEE Trans. on Neural Networks* **20** (5), 827-839.

Hughes, G.F. (1968) On the mean accuracy of statistical pattern recognizers. *IEEE Trans. on Information Theory* **14** (1), 55-63.

Jähne, B. (2002) *Digital Image Processing.* Springer-Verlag, Berlin.

Jack, L.B. and Nandi, A.K. (2002) Fault detection using support vector machines and artificial neural networks, augmented by genetic algorithms. *Mechanical Systems and Signal Processing*, **16** (2-3), 373-390.

Kohonen, T., Oja, E., Simula, O., Visa, A. and Kangas, J. (1996) Engineering applications of the self-organizing map. *Proc. of the IEEE*, **84**, 1358-1384.

Kuncheva, L.I. (2004) *Combining Pattern Classifiers.* John Wiley & Sons, Hoboken, NJ.

Kusiak, A. and Smith M. (2007) Data mining in design of products and production systems. *IFAC Annual Reviews in Control* **31** (1), 147-156.

Liu, H. and Yu, L. (2005) Towards integrating feature selection algorithms for classification and clustering. *IEEE Trans. on Knowledge and Data Engineering* **17** (4), 491-502.

Markopoulos, A.P., Manolakos, D.E. and Vaxevanidis, N. (2008) Artificial neural network models for the prediction of surface roughness in electrical discharge machining. *J. Intelligent Manufacturing* **19** (3), 283-292.

Newell, A. (1990) *Unified Theories of Cognition.* Harvard University Press, Cambridge, Massachussets.

Oh, I.S., Lee, J.S. and Moon, B.R. (2004) Hybrid genetic algorithms for feature selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **26** (11), 1424-1437.

Palenichka, R.M., Zaremba, M.B. and Missaoui, R. (2006) Multi-scale model-based feature extraction in structural texture images. *Journal of Electronic Imaging* **15** (2), 1-15.

Palenichka, R.M. and Zaremba, M.B. (2007) Multiscale isotropic matched filtering for individual tree detection in LiDAR images. *IEEE Trans. on Geoscience and Remote Sensing* **45** (12), 3944-3956.

Prakash, A., Tiwari, M. and Shankar, R. (2008) Optimal job sequence determination and operation machine allocation in flexible manufacturing systems: an approach using adaptive hierarchical ant colony algorithm. *J. Intelligent Manufacturing* **19** (2), 161-173.

Repaka, S.R. and Truax, D.D. (2004) Comparing spectral and object based approaches for classification and transportation feature extraction from high resolution multispectral imagery. *Proc. ASPRS Annual Conference*, Denver, Colorado, May 2004, 11-22.

Revesz, P.Z. (1993) On the semantics of theory change: Arbitration between old and new information. *Proc. 12th ACM SIGACT Symp. on Principles of Database Systems*, 71-79.

Rokach, L. and Maimon, O. (2006) Data mining for improving the quality of manufacturing: a feature set decomposition approach. *J. Intelligent Manufacturing* **17** (3), 285-299.

Roweis, T. and Saul, L. (2000) Nonlinear dimensionality reduction by locally linear embedding. *Science* **290**, 2323-2326.

Seki, H. et al. (2005) *Virtual Production Enterprise Network (VIPNET).* IMS 0431 Summary Report, IMS Promotion Center, Seoul, Korea.

Shawe-Taylor, J. and Cristianini, N. (2004) *Kernel Methods for Pattern Analysis.* Cambridge University Press, Cambridge, UK.

Schölkopf, B. and Smola, A.J. (2002) *Learning with Kernels.* The MIT Press, Cambridge, Mass.

Starck, J.L., Murtagh, F. and Bijaoui, A. (1998) *Image Processing and Data Analysis: The Multiscale Approach*, Cambridge University Press, Cambridge, UK.

Tagare, H.D., Toyama, K. and Wang, J.G. (2001) A maximum-likelihood strategy for directing attention during visual search. *IEEE Trans. Pattern Analysis and Machine Intelligence* **23** (5), 490-500.

Tasan, S.O. and Tunali, S. (2008) A review of the current applications of genetic algorithms in assembly line balancing. *J. Intelligent Manufacturing* **19** (1), 49-69.

Tenenbaum, J.B., de Silva, V. and Langford, J.C. (2000) A global geometric framework for nonlinear dimensionality reduction. *Science* **290**, 2319-2323.

Tsang, I.W. and Kwok, J.T. (2006) Efficient hyperkernel learning using second-order cone programming. *IEEE Trans. on Neural Networks* **17** (1), 48-58.

Vapnik, V. (1998) *Statistical Learning Theory.* John Wiley & Sons, Inc., New York.

Verikas, A. and Bacauskiene, M. (2002) Feature selection with neural networks. *Pattern Recognition Letters* **23** (11), 1323-1335.

Wang, K. (2007) Applying data mining to manufacturing: The nature and implications. *J. Intelligent Manufacturing* **18** (4), 487-495.

Watanabe, S. (1969) Knowing and Guessing: A Quantitative Study of Inference and Information. John Wiley & Sons, New York.

Wolpert, D.H. and Macready, W.G. (1995) *No free lunch theorems for search.* Technical Report SFI-TR-05-010, Santa Fe Institute. Santa Fe, NM.

Yoon, T., Oota, Y., Naka, Y., Yoshinaga, T., Shibao, K., Igoshi, M., Matsushima, K. and Suzuki, T. (2002) Knowledge fusion among the Virtual Production Enterprises within the Technology Information Infrastructure Environment. *Proc. IEEE Engineering Management Conference*, Cambridge, UK, **1**, 35-40.

Zaremba, M.B. and Fraczak, W. (2001) Dynamic task communication for concurrent processing in distributed systems. *Concurrent Engineering: Research and Applications* **9** (2), 155-165.

Zaremba, M.B. (2008) Remote sensing applications – New vistas for measurement and control. *Journal of Automation, Mobile Robotics & Intelligent Systems* **2** (3), 3-12.

Zhang, P., Verma, B. and Kumar, K. (2005) Neural vs. statistical classifier in conjunction with genetic algorithm based feature selection. *Pattern Recognition Letters* **26** (7), 909-919.

Zien, A. and Ong, C.S. (2007) Multiclass multiple kernel learning. *Proc. 24th Int. Conf. Machine Learning.* Cornvallis, OR, 1191-1198.