Pattern Synthesis for Nonparametric Pattern Recognition

P. Viswanath
Indian Institute of Technology-Guwahati, India

M. Narasimha Murty
Indian Institute of Science, India

Shalabh Bhatnagar
Indian Institute of Science, India

INTRODUCTION

Parametric methods first choose the form of the model or hypotheses and estimates the necessary parameters from the given dataset. The form, which is chosen, based on experience or domain knowledge, often, need not be the same thing as that which actually exists (Duda, Hart & Stork, 2000). Further, apart from being highly error-prone, this type of methods shows very poor adaptability for dynamically changing datasets. On the other hand, non-parametric pattern recognition methods are attractive because they do not derive any model, but works with the given dataset directly. These methods are highly adaptive for dynamically changing datasets. Two widely used non-parametric pattern recognition methods are (a) the nearest neighbor based classification and (b) the Parzen-Window based density estimation (Duda, Hart & Stork, 2000). Two major problems in applying the non-parametric methods, especially, with large and high dimensional datasets are (a) the high computational requirements and (b) the curse of dimensionality (Duda, Hart & Stork, 2000). Algorithmic improvements, approximate methods can solve the first problem whereas feature selection (Isabelle Guyon & André Elisseeff, 2003), feature extraction (Terabe, Washio, Motoda, Katai & Sawaragi, 2002) and bootstrapping techniques (Efron, 1979; Hamamoto, Uchimura & Tomita, 1997) can tackle the second problem. We propose a novel and unified solution for these problems by deriving a *compact and generalized abstraction* of the data. By this term, we mean a compact representation of the given patterns from which one can retrieve not only the original patterns but also some artificial patterns. The compactness of the abstraction reduces the computational requirements, and its generalization reduces the curse of dimensionality effect. Pattern synthesis techniques accompanied with compact representations attempt to derive compact and generalized abstractions of the data. These techniques are applied with (a) the nearest neighbor classifier (NNC) which is a popular non-parametric classifier used in many fields including data mining since its conception in the early fifties (Dasarathy, 2002) and (b) the Parzen-Window based density estimation which is a well known non-parametric density estimation method (Duda, Hart & Stork, 2000).

BACKGROUND

Pattern synthesis techniques, compact representations and its application with NNC and Parzen-Window based density estimation are based on more established fields:

- **Pattern recognition:** Statistical techniques, parametric and non-parametric methods, classifier design, nearest neighbor classification, probability density estimation, curse of dimensionality, similarity measures, feature selection, feature extraction, prototype selection, and clustering techniques.
- **Data structures and algorithms:** Computational requirements, compact storage structures, efficient nearest neighbor search techniques, approximate search methods, algorithmic paradigms, and divide-and-conquer approaches.
- **Database management:** Relational operators, projection, cartesian product, data structures, data management, queries, and indexing techniques.

MAIN FOCUS

Pattern synthesis, compact representations followed by its application with NNC and Parzen-Window density estimation are described below.

Pattern Synthesis

Generation of artificial new patterns using the given set of patterns is called pattern synthesis. Instance based pattern synthesis uses the given training patterns and some of the properties of the data. It can generate a finite number of new patterns. Computationally this can be less expensive than deriving a model from which the new patterns can be extracted. This is especially useful for non-parametric methods like NNC and Parzen-Window based density estimation (Duda, Hart and Stork, 2000) which directly use the training instances. It is argued that using a larger synthetic set can reduce the bias of the density estimation or classification (Viswanath, Murty & Bhatnagar, 2006). Further, the usage of the respective compact representations can result in reduction of the computational requirements.

This chapter presents two instance based pattern synthesis techniques called *overlap based pattern synthesis* and *partition based pattern synthesis* and their corresponding compact representations.

Overlap Based Pattern Synthesis

Let F be the set of features (or attributes). There may exist a three-block partition of F, say, $\{A, B, C\}$ with the following properties. For a given class, there is a dependency (probabilistic) among features in A U B. Similarly, features in B U C have a dependency. However, features in A (or C) can affect those in C (or A) only through features in B. That is, to state more formally, A and C are statistically independent given B. Suppose that this is the case and we are given two patterns $X = (a_1, b, c_1)$ and Y = (a_2, b, c_2) such that a_1 is a feature-vector that can be assigned to the features in A, b to the features in B and c_1 to the features in C, respectively. Similarly, a_2, b and c_2 are feature-vectors that can be assigned to features in A, B, and C, respectively. Then, our argument is that the two patterns (a_1, b, c_2) and (a_2, b, c_1) are also valid patterns in the same class or category as X and Y. If these two new patterns are not already in the class of patterns then it is only because of the finite nature of the set. We call this type of generation of additional patterns as *overlap based pattern synthesis*, because this kind of synthesis is possible only if the two given patterns have the same feature-values for features in B. In the given example, feature-vector b is common between X and Y and therefore is called the *overlap*. This method is suitable only with discrete valued features (can be of symbolic or categorical types also). If more than one such partition exists then the synthesis technique is applied sequentially with respect to the partitions in some order.

One simple example to illustrate this concept is as follows. Consider a supermarket sales database where two records, *(bread, milk, sugar)* and *(coffee, milk, biscuits)* are given. Let us assume, it is known that there is a dependency between (i) *bread* and *milk*, (ii) *milk* and *sugar*, (iii) *coffee* and *milk*, and (iv) *milk* and *biscuits*. Then the two new records that can be synthesized are *(bread, milk, biscuits)* and *(coffee, milk, sugar)*. Here *milk* is the overlap. A compact representation in this case is shown in Figure 1 where a path from left to right ends denotes a data item or pattern. So we get four patterns in total from the graph shown in Figure 1 (two original and two synthetic patterns). Association rules derived from association rule mining (Han & Kamber, 2001) can be used to find these kinds of dependencies. Generalization of this concept and its compact representation for large datasets are described below.

If the set of features, F can be arranged in an order such that $F = \{f_1, f_2, ..., f_d\}$ is an ordered set with f_k being the k^{th} feature and all possible three-block partitions can be represented as $P_i = \{A_i, B_i, C_i\}$ such that $A_i = (f_1, ..., f_a)$, $B_i = (f_{a+1}, ..., f_b)$ and $C_i = (f_{b+1}, ..., f_d)$ then the compact representation called *overlap pattern graph* is described with the help of an example.

Figure 1.