Pattern Synthesis for Large-Scale Pattern Recognition

P. Viswanath
Indian Institute of Science, India

M. Narasimha Murty
Indian Institute of Science, India

Shalabh Bhatnagar
Indian Institute of Science, India

INTRODUCTION

Two major problems in applying any pattern recognition technique for large and high-dimensional data are (a) high computational requirements and (b) curse of dimensionality (Duda, Hart, & Stork, 2000). Algorithmic improvements and approximate methods can solve the first problem, whereas feature selection (Guyon & 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 *nearest neighbor classifier (NNC),* which is a popular nonparametric classifier used in many fields, including data mining, since its conception in the early 1950s (Dasarathy, 2002).

BACKGROUND

Pattern synthesis techniques, compact representations and its application with NNC are based on more established fields:

- **Pattern Recognition:** Statistical techniques, parametric and nonparametric methods, classifier design, nearest neighbor classification, 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 THRUST

Pattern synthesis, compact representations followed by its application with NNC, are described in this section.

Pattern Synthesis

Generation of artificial new patterns by using the given set of patterns is called *pattern synthesis.* There are two broad ways of doing pattern synthesis: *model-based pattern synthesis* and *instance-based pattern synthesis*.

In model-based pattern synthesis, a model (such as the Hidden Markov model) or description (such as probability distribution) of the data is derived first and is then used to generate new patterns. This method can be used to generate as many patterns as needed, but it has two drawbacks. First, any model depends on the underlying assumptions; hence, the synthetic patterns generated can be erroneous. Second, deriving the model might be computationally expensive. Another argument against this method is that if pattern classification is the purpose, then the model itself can be used without generating any patterns at all!

Instance-based pattern synthesis, on the other hand, uses the given training patterns and some of the properties of the data. It can generate only a finite number of

new patterns. Computationally, this method can be less expensive than deriving a model. It is especially useful for nonparametric methods, such as NNC- and Parzen-window-based density estimation (Duda et al., 2000), which directly use the training instances. Further, this method can also result in reduction of the computational requirements.

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

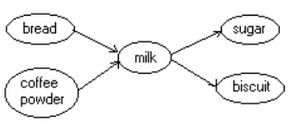
Overlap-based Pattern Synthesis

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 it more formally, A and C are statistically independent, given B. Suppose that this is the case and you 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. Similarly, a_2, b, and c_2 are feature-vectors that can be assigned to features in A, B, and C, respectively. Our argument, then, 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, it is only because of the finite nature of the set. We call this generation of additional patterns an *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 also be of symbolic or categorical types). 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. Assume that a known dependency exists between (a) bread and milk, (b) milk and sugar, (c) coffee and milk, and (d) milk and biscuits. The two new records that can then 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 denotes a data item or pattern. So you get four patterns total (two original and two synthetic patterns) from the graph shown in Figure 1. Association rules derived from association rule mining (Han & Kamber, 2000) can be used to find these kinds of dependencies. Generalization of this concept and its compact representation for large datasets are described in the paragraphs that follow.

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. A compact representation



Overlap Pattern Graph (OLP-graph)

Let $F = (f_1, f_2, f_3, f_4, f_5)$. Let two partitions satisfying the conditional independence requirement be $P_1 = \{ \{f_1\}, \{f_2, f_3\}, \{f_4, f_5\}\}$ and $P_2 = \{ \{f_1, f_2\}, \{f_3, f_4\}, \{f_5\}\}$. Let three given patterns be *(a,b,c,d,e), (p,b,c,q,r),* and *(u,v,c,q,w)*, respectively. Because *(b,c)* is common between the1st and 2nd patterns, two synthetic patterns that can be generated are *(a,b,c,q,r)* and *(p,b,c,d,e)*. Likewise, three other synthetic patterns that can be generated are *(p,b,c,d,e)*, *(p,b,c,q,w),* and *(a,b,c,q,w)*. (Note that the last synthetic pattern is derived from two earlier synthetic patterns.) A compact representation called *overlap pattern graph (OLP-graph)* for the entire set (including both given and synthetic patterns) is shown in Figure 2, where a path from left to right represents a pattern. The graph is constructed by inserting the given patterns, whereas the patterns that can be extracted out of the graph form the entire synthetic set consisting of both original and synthetic patterns. Thus, from the graph in Figure 2, a total of eight patterns can be extracted, five of which are new synthetic patterns.

OLP-graph can be constructed by scanning the given dataset only once and is independent of the order in which the given patterns are considered. An approximate method for finding partitions, a method for con-

Figure 2. OLP-graph