A Unified and Flexible Framework for Comparing Simple and Complex Patterns*

Ilaria Bartolini[1], Paolo Ciaccia[1], Irene Ntoutsi[2], Marco Patella[1], and Yannis Theodoridis[2]

[1] DEIS – IEIIT/BO-CNR, University of Bologna, Italy
{ibartolini,pciaccia,mpatella}@deis.unibo.it
[2] Research Academic Computer Technology Institute, Athens, Greece
and Department of Informatics, University of Piraeus, Greece
{ntoutsi,ytheod}@cti.gr

Abstract. One of the most important operations involving Data Mining patterns is computing their similarity. In this paper we present a general framework for comparing both simple and complex patterns, i.e., patterns built up from other patterns. Major features of our framework include the notion of structure and measure similarity, the possibility of managing multiple coupling types and aggregation logics, and the recursive definition of similarity for complex patterns.

1 Introduction

Data Mining and Knowledge Discovery techniques are commonly used to extract condensed artifacts, like association rules, clusters, keywords, etc., from huge datasets. Among the several interesting operations on such patterns (modeling, storage, retrieval), one of the most important is that of comparison, i.e., establishing whether two patterns are similar or not [1]. Such operation could be of valuable use whenever we have to measure differences of patterns describing evolving data or data extracted from different sources, and to measure the different behavior of Data Mining algorithms over a same dataset. A similarity operator between patterns could also be used to express similarity queries over pattern bases [5].

In the following we present a general framework for the assessment of similarity between both simple and complex patterns. Major features of our framework include the notion of structure and measure similarity, the possibility of managing multiple coupling types and aggregation logics, and the recursive definition of similarity for complex patterns, i.e., patterns whose structure consists of other patterns. This considerably extends FOCUS [1], the only existing framework for the comparison of patterns, which does not consider complex patterns, neither it allows different matching criteria (i.e., coupling types), since it limits itself to a fixed form of matching (based on a so-called greatest common refinement).

2 A Framework for the Evaluation of Pattern Similarity

We approach the problem of defining a general framework able to guarantee flexibility with respect to pattern types and their similarity criteria and, at the same time, to

* This work was supported by the European Commission under the IST-2001-33058 Thematic Network. PANDA "PAtterns for Next-generation DAtabase systems" (2001-04).

J.-F. Boulicaut et al. (Eds.): PKDD 2004, LNAI 3202, pp. 496–499, 2004.
© Springer-Verlag Berlin Heidelberg 2004

exploit common aspects of the comparison problem, by starting from the logical model proposed in [3], where each pattern type includes a structure schema *ss*, defining the pattern space, and a measure schema *ms*, describing the measures that quantify the quality of the source data representation achieved by each pattern. A pattern *p* of type *pt* instantiates the structure schema and the measure schema, thus leading to a structure, *p.s*, and a measure, *p.m*. In the basic case, the similarity between patterns is computed by means of a similarity operator, *sim*, which has to take into account both the similarity between the patterns' structures and the similarity between the measures. A pattern is called *simple* if its structure does not include other patterns, otherwise it is called a *complex pattern*. For instance, an Euclidean cluster in a *D*-dimensional space is a simple pattern whose structure is represented by the center (a *D*-dimensional vector) and radius (a real value) of the cluster. Measures for a cluster might include, for example, the average intra-cluster distance and its *support* (fraction of the data points represented by the cluster).

The similarity between two *simple patterns* of the same type *pt* can be computed by combining, by means of an *aggregation function* f_{aggr}, the similarity between both the structure and the measure components:

$$sim(p_1, p_2) = f_{aggr}(sim_{struct}(p_1.s, p_2.s), sim_{meas}(p_1.m, p_2.m)) \quad (1)$$

If the two patterns have the same structural component, then $sim_{struct}(p_1.s, p_2.s) = 1$, and the measure of similarity naturally corresponds to a comparison of the patterns' measures, e.g., by aggregating differences between each measure. In the general case, however, the patterns to be compared have different structural components, thus a preliminary step is needed to "reconcile" the two structures to make them comparable.

The computation of similarity between simple patterns is summarized in Fig. 1. It has to be remarked that the sim_{struct} block could also encompass the use of an underlying domain knowledge. For instance, if we are comparing keywords extracted from textual documents (i.e., the pattern is a keyword), the similarity between them can be computed by exploiting the presence of an ontology, such as WordNet [2].

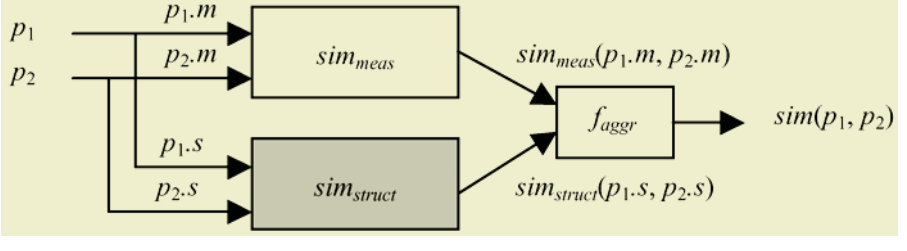

Fig. 1. Assessment of similarity between patterns.

The case of *complex patterns*, i.e., patterns whose structure includes other patterns, is particularly challenging, because the similarity between structures of complex patterns depends in turn on the similarity between component patterns. For instance, a *clustering* pattern is the composition of *cluster* patterns.

Evaluation of similarity between complex pattern follows the same basic rationale shown in Fig. 1 of aggregating similarities between measure and structure components. However, the structure of complex patterns now consists of several other pat-

terns. In our framework, the similarity between the structure of complex patterns is conceptually evaluated in a bottom-up fashion, and can be adapted to specific needs/constraints by acting on two fundamental abstractions, namely the *coupling type*, which is used to establish how component patterns can be *matched*, and the *aggregation logic*, which is used to combine the similarity scores obtained for coupled component patterns into a single overall score representing the similarity between the complex patterns.

Coupling type: Since every complex pattern can be eventually decomposed into a number of component patterns, in comparing two complex patterns, cp_1 and cp_2, we need a way to associate component patterns of cp_1 to component patterns of cp_2. To this end, the *coupling type* just establishes the way component patterns can be matched. Assume without loss of generality that component patterns are given an ordinal number, thus the structure of each complex pattern can be represented as $cp.s = (p^1, p^2, \ldots, p^N)$. Each coupling between cp_1 and cp_2 can be represented by a *matching matrix* $\mathbf{X}_{N\times M} = (x_{ij})$, where each $x_{ij} \in [0,1]$ ($i = 1, \ldots, N$; $j = 1, \ldots, M$) represents the (amount of) matching between ${p_1}^i$ and ${p_2}^j$. Different coupling types essentially introduce a number of constraints on the x_{ij} coefficients, e.g.:

- 1–1 *matching*: In this case we accept at most one matching for each component pattern ${p_1}^i$ or ${p_2}^j$.
- *EMD matching*: The Earth Mover's Distance (EMD) [4] is used to compare two distributions and is a particular *N–M* matching. Computing EMD is based on solving the well-known *transportation problem*. EMD has been applied, among others, to compare images by taking into account existing inter-color similarities [4].

Aggregation logic: Among all the feasible matchings, the rationale is to pick the "best" one. To this end, the overall similarity between complex patterns is computed by aggregating similarity scores obtained for matched component patterns, and then taking the maximum over all legal matchings. Formally, each pairing $({p_1}^i, {p_2}^j)$ contributes to the overall score, as evaluated by the *matching aggregation function* g_{aggr}, with the similarity, $sim({p_1}^i, {p_2}^j)$, between its matched component patterns:

$$sim_{struct}(cp_1.s, cp_2.s) = \max_{\mathbf{X}}(g_{aggr}(({p_1}^1, {p_1}^2, \ldots, {p_1}^N), ({p_2}^1, {p_2}^2, \ldots, {p_2}^M), \mathbf{X})) \quad (2)$$

The process of computing the structure similarity between complex patterns can be conceptually summarized as follows. Given a coupling type, any possible legal matching is generated, and the similarity scores between pairs of matched patterns are computed as in Fig. 1. Then, such similarity scores are combined together by means of the matching aggregation function. Finally, the matching attaining the highest overall similarity score is determined. In case of multi-level aggregations, this process has to be recursively applied to component sub-patterns.

The above-described scheme can turn to be highly inefficient. For this reason, when efficient evaluation of $sim_{struct}(cp_1.s, cp_2.s)$ is an issue, we provide efficient algorithms for the solution of the best-coupling problem which do not require going through all possible matchings.

References

1. V. Ganti, J. Gehrke, R. Ramakrishnan, and W.-Y. Loh. A framework for measuring changes in data characteristics. PODS'99, pp. 126-137.
2. G. A. Miller: WordNet: A lexical database for English. CACM, 38(11), 39-41, 1995.
3. S. Rizzi, E. Bertino, B. Catania, M. Golfarelli, M. Halkidi, M. Terrovitis, P. Vassiliadis, M. Vazirgiannis, and E. Vrachnos. Towards a logical model for patterns. ER 2003, pp. 77-90.
4. Y. Rubner, C. Tomasi, and L. J. Guibas. A metric for distributions with applications to image databases. ICCV'98, pp. 59-66.
5. Y. Theodoridis, M. Vazirgiannis, P. Vassiliadis, B. Catania, and S. Rizzi. A manifesto for pattern bases. PANDA Technical Report TR-2003-03, 2003.