Discovering Structural Patterns in Telecommunications Data

Andi Baritchi, Diane J. Cook, and Lawrence B. Holder

Department of Computer Science and Engineering
University of Texas at Arlington
Box 19015 (416 Yates St.), Arlington, TX 76019-0015
E-mail: {andi | cook | holder}@cse.uta.edu
Phone: (817) 272-3606
Fax: (817) 272-3784

Abstract

With the increasing amount and complexity of data being collected, there is an urgent need to create automated techniques for mining the data. In particular, data being generated and stored by telecom companies overwhelms scientists' ability to manually discover patterns in the data. Because much of this data is structural in nature, or composed of parts and relations between the parts, linear attribute-value based algorithms will not capture all of the intricacies of the data. Hence, there exists a need to develop scalable tools to analyze and discover concepts in structural databases.

Introduction

New technology and new laws are changing the telecommunications industry at a blinding rate. New methods of mining the data are needed to understand and control these changes. Because the amount of collected data far exceeds the ability to manually search for and interpret patterns in the data, there is a need to improve the discovery of knowledge in these large databases.

Equipment and service costs for wireless telephones have plummeted in the past decade, attracting millions of new customers. Many people who originally purchased these devices just for emergencies are now using them as their primary phones, which can translate to usage in excess of 400 calls per month for a good customer. Wireless service carriers keep extensive logs of calls made, and this data may contain items such as cell saturation, signal strength, handoffs between cells, call times, call source, and call destination. Mining this data could help wireless service carriers learn call trends, find areas where new cell sites are needed, identify strategic cells with high paying customers, and more.

Numerous approaches have been developed for discovering knowledge in databases using a linear, attribute-value representation. Although much of the data collected today has an explicit or implicit structural component (e.g., spatial or temporal), few discovery systems are designed to handle this type of data [4]. One reported method for dealing specifically with structural data is with the SUBDUE system [3]. SUBDUE provides a method for discovering substructures in structural databases using the minimum description length (MDL) principle introduced by Rissanen.

In this paper, we provide an overview of the knowledge discovery capabilities of the SUBDUE data mining system and demonstrate the ability of this system to discover structural patterns in telecommunications data. Furthermore, we present some enhancements that were made to Subdue that allow it to better represent numeric discoveries.

The Subdue System

The SUBDUE system is a structural discovery tool that finds substructures in a graph-based representation of structural databases using the minimum description length (MDL) principle introduced by Rissanen [5]. SUBDUE discovers substructures that compress the original data and represent structural concepts in the data. Once a substructure is discovered, the substructure is used to simplify the data by replacing instances of the substructure with a pointer to the newly discovered substructure. The discovered substructures allow abstraction over detailed structures in the original data. Iteration of the substructure discovery and replacement process constructs a hierarchical description of the structural data in terms of the discovered substructures. This hierarchy provides varying levels of interpretation that can be accessed based on the specific goals of the data analysis.

SUBDUE represents structural data as a labeled graph. Objects in the data map to vertices or small subgraphs in the graph, and relationships between objects map to directed or undirected edges in the graph. A substructure is a connected subgraph within the graphical representation. This graphical representation serves as input to the substructure discovery system. Figure 1 shows a geometric example of such an input graph. The objects in the figure become labeled vertices in the graph, and the relationships become labeled edges in the graph. The graphical representation of the substructure discovered by SUBDUE from this data is also shown in Figure 1. One of the four instances of the substructure is highlighted in the input graph. An instance of a substructure in an input graph is a set of vertices and edges from the input graph that match, graph theoretically, to the graphical representation of the substructure.

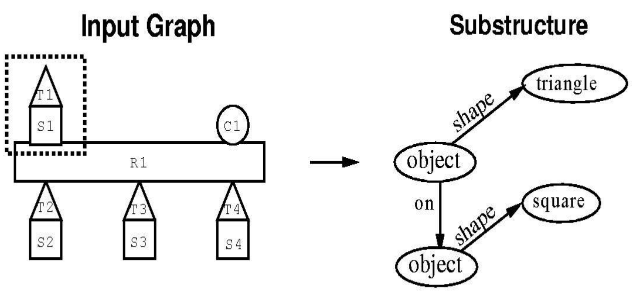

Figure 1. Example substructure in graph form.

The substructure discovery algorithm used by SUBDUE is a computationally-constrained beam search. That is, SUBDUE keeps a limited-size list of the current best discoveries as it searches for better ones. The algorithm begins with the substructure matching a single vertex in the graph. Each iteration the algorithm selects the best substructure and incrementally expands the instances of the substructure. The algorithm searches for the best substructure until all possible substructures have been considered or the total amount of computation exceeds a given limit. Evaluation of each substructure is determined by how well the substructure compresses the description length of the database.

Because instances of a substructure can appear in different forms throughout the database, an inexact graph match is used to identify substructure instances. In this inexact match approach, each distortion of a graph is assigned a cost. A distortion is described in terms of basic transformations such as deletion, insertion, and substitution of vertices and edges. The distortion costs can be determined by the user to bias the match for or against particular types of distortions. SUBDUE's run time is constrained to be polynomial by user-defined limits on the beam width, the total number of substructure definitions to consider, and computational constraints on the inexact graph match.

SUBDUE has been successfully applied with and without domain knowledge to databases in domains including image analysis, CAD circuit analysis, Chinese character databases, program source code, chemical reaction chains, Brookhaven protein databases, and artificially-generated databases [2,3].

A variety of approaches to unsupervised discovery using structural data have been proposed (e.g., [1,6]). Many of these approaches use a knowledge base of concepts to classify the structural data. These systems perform concept learning over examples and categorization of observed data. While the above methods represent examples as distinct objects and process individual objects one at a time, our method stores the entire database (with embedded objects) as one graph and processes the graph as a whole.

Scientific discovery systems that use domain knowledge have also been developed. However, these systems are targeted for a single application domain. One example is MECHEM [7], which relies on domain knowledge to constrain the discovery of credible explanatory hypotheses specific to the domain of chemistry. In contrast, SUBDUE is devised for general-purpose automated discovery with or without domain knowledge. Hence, the method can be applied to many structural domains.

Enhancing the Numeric Capabilities of SUBDUE

SUBDUE was designed initially to discover patterns in symbolic data. However, much of the data collected for applications such as telecommunications is numeric. Thus, the capabilities of SUBDUE have been extended to process numeric data.

Inexact Numeric Label Match

In particular, since vertex and edge labels may actually represent numeric values, the graph match algorithm is modified to allow some variation in these values for corresponding vertices and edges.

In the revised version of the system, the input graph specifies, with each numeric label n, a match type for that label. The three allowed options are:

- Exact Match: Label n matches another label y if x = y.
- Tolerance Match: Label matches y if the absolute value of their difference is less than t.
- Difference Match: MatchCost(x,y) is defined as the probability that y is drawn from a probability distribution with mean of x and standard deviation defined in the input file.

When SUBDUE is run on input graphs with numeric labels, these parameters are used to decide when two numeric labels can be considered a match. Hence, it is important to wisely choose the types of numeric match for the different labels, and the amount of inexactness allowed.

Enhanced Post-processing of Numeric Patterns

Due to the inexact numeric match presented in the previous section, a range of values may match a vertex label or edge label. Since SUBDUE creates a substructure definition for each discovered pattern, these substructure definitions need to provide the range of values covered by instances of the substructure in the graph.

Originally, when printing the discovered substructures at the end of the run, SUBDUE would print the values of only one instance of each discovered substructures. Although this method was ideal when all labels in matched substructures were identical, the inexact numeric match presents the need to collect and print out information about the matched numeric labels at output time.

We decided to collect the data concerning numeric labels and their matched counterparts in a post-processing step when printing out the substructures. Since SUBDUE maintains pointers from each discovered substructure to each of its matched instances, the post processing step involved iterating through these instances and deciphering the mappings between the labels in the instances and the labels in the discovered substructure. Once these mappings were determined, statistics were collected to describe the range of values discovered between the instances of each numeric label in the discovered substructure. The statistics, which are printed out with the definitions of the discovered substructures, include numeric ranges, means, bucketizations, and standard probability distributions of the numeric labels in those substructures.

Experiments

We generated synthetic call records to evaluate the ability of the SUBDUE system to discover trends in telecommunications data. The strategy applied here was to embed intentional patterns in the synthetic call records and see if the SUBDUE system could discover these patterns. These experiments will later be replicated using data provided by local telecom companies.

Testing Methodology

To facilitate graph generation from actual call records in the future, we implemented a graph generator for call records. When given call records in comma-separated-values (CSV) format, the program generates SUBDUE-readable graphs representing these call records. To create synthetic data with embedded patterns, we generated data corresponding to 60 wireless transactions. The graph representation is a star configuration centered around a hub node labeled ``call". Additional nodes are created for each transaction with data for call features such as caller city, callee city, call initialization time, call duration, and distance type (long distance or local). Each feature node is connected by a directed edge to the hub node, and the edge is labeled with the feature name.

We chose two patterns for embedding into the synthetic data, both of which mimic patterns in telecommunications data. The patterns are:

1. Long distance calls are generally short whereas local calls are usually long, and
2. Local calls occur more often during business hours whereas long distance calls are usually placed in the late evening.

Results

Upon running the SUBDUE system on our synthetic data in graph form, we analyzed the six substructures output by the system. Figure 2 shows two of these six substructures - the ones we found most interesting. The substructures in this figure show that the following patterns are common:

1. Local calls originating between 11:00am and 2:30pm with 30 to 45 minute durations, and
2. Long distance calls originating between 6:00pm and 10:00pm with 5 to 20 minute durations.

These substructures accurately describe the intentional patterns we embedded into our synthetic data. Furthermore, these results demonstrate SUBDUE's capability to report ranges of discovered numeric labels.

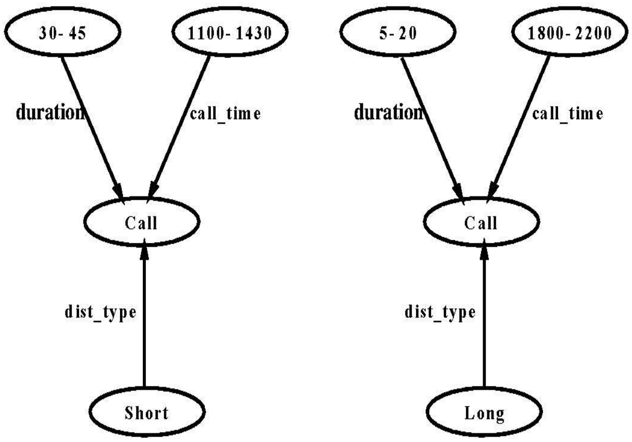

Figure 2. Telecom patterns discovered by Subdue.

Conclusions

The increasing structural component of databases such as those containing telecom data requires data mining algorithms capable of handling structural information. The SUBDUE system is specifically designed to discover concepts in structural databases. In this paper, we have described how SUBDUE can be used to discover interesting patterns in a sample database of wireless calls. We are continuing work in this direction to find patterns of interest in data provided by several telecom companies. Future work will also make use of additional concept learning and clustering capabilities within SUBDUE to perform a variety of data mining operations on data for this dynamic and data-intensive application.

References

1. D. Conklin. Machine discovery of protein motifs. Machine Learning, 21:125-150, 1995.

2. D. J. Cook and L. B. Holder. Graph-based data mining. To appear in IEEE Intelligent Systems, 2000.

3. D. J. Cook, L. B. Holder, and S. Djoko. Scalable discovery of informative structural concepts using domain knowledge. IEEE Expert, 11(5), 1996.

4. U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: An overview. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, Advances in Knowledge Discovery and Data Mining, chapter 1, pages 1-34. MIT Press, 1996.

5. J. Rissanen. Stochastic Complexity in Statistical Inquiry. World Scientific Publishing Company, 1989.

6. K. Thompson and P. Langley. Concept formation in structured domains. In D. H. Fisher and M. Pazzani, editors, Concept Formation: Knowledge and Experience in Unsupervised Learning, chapter 5. Morgan Kaufmann Publishers, 1991.

7. R. E. Valdes-Perez. Conjecturing hidden entities by means of simplicity and conservation laws: Machine discovery in chemistry. Artificial Intelligence, 65:247-280, 1994.