

# Graph-Based Relational Learning: Current and Future Directions

Lawrence B. Holder  
Computer Science and Engineering  
University of Texas at Arlington  
Box 19015, Arlington, TX 76019  
holder@cse.uta.edu

Diane J. Cook  
Computer Science and Engineering  
University of Texas at Arlington  
Box 19015, Arlington, TX 76019  
cook@cse.uta.edu

## ABSTRACT

Graph-based relational learning (GBRL) differs from logic-based relational learning, as addressed by inductive logic programming techniques, and differs from frequent subgraph discovery, as addressed by many graph-based data mining techniques. Learning from graphs, rather than logic, presents representational issues both in input data preparation and output pattern language. While a form of graph-based data mining, GBRL focuses on identifying novel, not necessarily most frequent, patterns in a graph-theoretic representation of data. This approach to graph-based data mining provides both simplifications and challenges over frequency-based approaches. In this paper we discuss these issues and future directions of graph-based relational learning.

## General Terms

Graph-based relational learning

## Keywords

Graph, relational, structural, learning, discovery

## 1. INTRODUCTION

Graph-based relational learning (GBRL) is the task of finding novel, useful, and understandable graph-theoretic patterns in a graph representation of data. While data mining approaches in general address the same task, most graph-based data mining approaches focus only on the frequency of the pattern. Therefore, we view GBRL as a subfield of graph-based data mining (GBDM), because the novelty of a pattern typically involves more than just the frequency of the pattern in the data. This distinguishes GBRL from the many GBDM approaches focused on finding frequent subgraphs [7; 10; 17], i.e., all subgraphs in the data whose number of instances above some minimum support.

We also distinguish GBRL from inductive logic programming (ILP) approaches to relation learning. Obviously, the underlying representations (graphs vs. logic) are the primary distinction. While graphs are extremely flexible in terms of the data they can encode, the semantics are not well defined. Conceptual graphs (CGs) [16] represent a body of work aimed at defining a graph semantics similar to that of first-order logic. CGs attach a semantics to the graph by

distinguishing between relations and entities or attributes. This semantics allow conversion between CGs and restricted forms of first-order logic. For this reason CG equivalents of logic provide a reasonably unbiased mechanism for comparing graph-based and logic-based relational learners.

These equivalencies between graphs and logic raise the question of whether GBRL and ILP are performing basically the same type of relational learning, i.e., searching equivalent spaces. This is not true for two reasons. First, and we view this as an advantage for GBRL, ILP approaches rely on the prior identification of the predicate or predicates to be defined by the learned pattern. GBRL approaches are more data-driven, identifying any portion of the graph that is able to distinguish between classes. Second, and we view this as an advantage for ILP, the logic-based representation allows the expression of more complicated patterns involving, e.g., recursion, variables, and constraints among variables. Graphically speaking, variables and variable constraints would imply that a portion of a graphical pattern matches any arbitrary subgraph or that two parts of the graphical pattern must be identical without specifying the parts' structure. These representational limitations of graphs can be overcome, but at a computational cost.

## 2. GBRL

Only a few GBRL approaches have been developed to date. Two specific approaches, Subdue [2] and GBI [18], take a greedy approach to finding subgraphs maximizing an information theoretic measure. Subdue searches the space of subgraphs by extending candidate subgraphs by one edge. Each candidate is evaluated using a minimum description length metric [15], which measures how well the subgraph compresses the input graph if each instance of the subgraph were replaced by a single vertex. GBI continually compresses the input graph by identifying frequent triples of vertices, some of which may represent previously-compressed portions of the input graph. Candidate triples are evaluated using a measure similar to information gain. Kernel-based methods have also been used for supervised GBRL [9].

We describe two extensions to the basic GBRL approach that take particular advantage of an information theoretic evaluation measure combined with an iterative application. These extensions are *supervised learning* and *graph grammar induction*. We describe these extensions in the context of the Subdue GBRL system, but they can be implemented fairly easily with similar GBRL approaches.

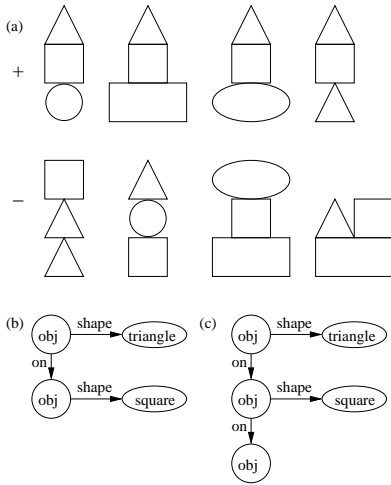


Figure 1: Graph-based supervised learning example with (a) four positive and four negative examples, (b) one possible graph concept, and (c) another possible graph concept.

## 2.1 Supervised Learning

Extending a GBRL approach to perform supervised learning involves, of course, the need to handle negative examples (focusing on the two-class scenario). In the case of a graph the negative information can come in three forms. First, the data may be in the form of numerous small graphs, or graph transactions, each labeled either positive or negative. Second, data may be composed of two large graphs: one positive and one negative. Third, the data may be one large graph in which the positive and negative labeling occurs throughout. We will consider the third scenario in section 3. The first scenario is closest to the standard supervised learning problem in that we have a set of clearly defined examples. Figure 1a depicts a simple set of positive and negative examples. Let  $G^+$  represent the set of positive graphs, and  $G^-$  represent the set of negative graphs. Then, one approach to supervised learning is to find a subgraph that appears often in the positive graphs, but not in the negative graphs. This amounts to replacing the information-theoretic measure with simply an error-based measure. For example, we would find a subgraph  $S$  that minimizes

$$\frac{|\{g \in G^+ | S \not\subseteq g\}| + |\{g \in G^- | S \subseteq g\}|}{|G^+| + |G^-|},$$

where  $S \subseteq g$  means  $S$  is isomorphic to a subgraph of  $g$ . The first term of the numerator is the number of false negatives, and the second term is the number of false positives.

This approach will lead the search toward a small subgraph that discriminates well, e.g., the subgraph in Figure 1b. However, such a subgraph does not necessarily compress well, nor represent a characteristic description of the target concept. We can bias the search toward a more characteristic description by using the information-theoretic measure to look for a subgraph that compresses the positive examples, but not the negative examples. If  $I(G)$  represents the description length (in bits) of the graph  $G$ , and  $I(G|S)$  represents the description length of graph  $G$  compressed by subgraph  $S$ , then we can look for an  $S$  that minimizes  $I(G^+|S) + I(S) + I(G^-) - I(G^-|S)$ , where the

last two terms represent the portion of the negative graph incorrectly compressed by the subgraph. This approach will lead the search toward a larger subgraph that characterizes the positive examples, but not the negative examples, e.g., the subgraph in Figure 1c.

Finally, this process can be iterated in a set-covering approach to learn a disjunctive hypothesis. If using the error measure, then any positive example containing the learned subgraph would be removed from subsequent iterations. If using the information-theoretic measure, then instances of the learned subgraph in both the positive and negative examples (even multiple instances per example) are compressed to a single vertex. We should note that the compression is a lossy one, i.e, we do not keep enough information in the compressed graph to know how the instance was connected to the rest of the graph. This approach is consistent with our goal of learning general patterns, rather than mere compression. For more information on graph-based supervised learning, see [6].

## 2.2 Graph Grammar Induction

As mentioned earlier, two of the advantages of an ILP approach to relational learning are the ability to learn recursive hypotheses and constraints among variables. Graph grammars offer the ability to represent recursive graphical hypotheses [5]. Graph grammars are similar to string grammars except that terminals can be arbitrary graphs rather than symbols from an alphabet. Graph grammars can be divided into two types: node-replacement grammars and hyperedge-replacement grammars. Node-replacement grammars allow non-terminals on vertices, and hyperedge-replacement grammars allow non-terminals on edges. Figure 2b shows an example of a context-free, node-replacement graph grammar. Recent research has begun to develop techniques for learning graph grammars [8; 4].

A variant of graph grammars called stochastic graph grammars [11] has been developed to represent uncertainty. Each production has an associated probability such that all the productions involving a particular non-terminal on the left-hand side sum to one. This induces a distribution over the graphs in the language accepted by the graph grammar. A related ILP formalism is stochastic logic programs (SLPs) [13], although SLPs achieve Turing equivalence, while stochastic graph grammars are limited to context-free languages.

A GBRL approach can be extended to consider graph grammar productions by analyzing the instances of a subgraph to see how they relate to each other. If two or more instances are connected to each other by an edge, then a recursive production rule generating an infinite sequence of such connected subgraphs can be constructed. A slight modification to the information-theoretic measure taking into account the extra information needed to describe the recursive component of the production is all that is needed to allow such a hypothesis to compete along side simple subgraphs (i.e., terminal productions) for maximizing compression. The above constraint that the subgraphs be connected by a single edge limits the grammar to be context free. More than one connection between subgraph instances can be considered, and would allow learning context-sensitive grammars, but the algorithm is exponential in the number of connections.

Figure 2b shows an example of a recursive, node-replacement graph grammar production rule learned from the graph in Figure 2a. These productions can be disjunctive, as in

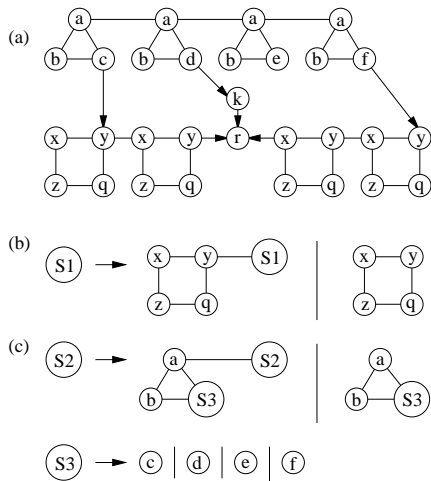


Figure 2: Graph grammar learning example with (a) the input graph, (b) the first grammar rule learned, and (c) the second and third grammar rules learned.

Figure 2c, which represents the final production learned from Figure 2a using this approach. The disjunctive rule is learned by looking for similar, but not identical, extensions to the instances of a subgraph. A new rule is constructed that captures the variability of the extensions, and is included in the pool of production rules competing based on their ability to compress the input graph. With a proper encoding of this disjunction information, the MDL criterion will tradeoff the complexity of the rule with the amount of compression it affords in the input graph.

An alternative to defining these disjunctive non-terminals is to construct a variable whose range consists of the different values of the production. In this way we can introduce constraints among variables contained in a subgraph by adding a constraint edge to the subgraph. For example, if the four instances of the triangle structure in Figure 2a each had another edge to a  $c$ ,  $d$ ,  $e$  and  $f$  vertex respectively, then we could propose a new subgraph, where these two vertices are represented by variables, and an equality constraint is introduced between them. If the range of the variable is numeric, then we can also consider inequality constraints between variables and other vertices or variables in the subgraph pattern.

A form of relational grammar induction takes place in ILP approaches, in that the learned theory can be viewed as a grammar for generating positive examples. In fact, the Duce system [12] used six transformation operators to search the space of propositional logic grammars guided by a simplicity measure. However, the transformations were not data driven as in the above graph grammar induction, and recursive productions were not learned. As we mentioned earlier, the starting symbol of an ILP theory is fixed by the input predicate to be learned. The graph grammar approach is not restricted to include a particular relation, but essentially invents relations that compress the relational data. Some earlier ILP systems did achieve this ability to invent relations in the context of learning a particular predicate theory. CIGOL [14] invented predicates using inverse resolution. These predicates were also evaluated based on their ability to compress the learned theory.

### 3. FUTURE DIRECTIONS

The need for practical GBRL algorithms is growing fast. Therefore, we need to address several challenging scalability issues, including incremental learning in dynamic graphs. Another issue regarding practical applications involves the blurring of positive and negative examples in a supervised learning task, that is, the graph has many positive and negative parts, not easily separated, and with varying degrees of class membership. We discuss these issues below.

#### 3.1 Scalability

Scaling GBRL approaches to very large graphs, graphs too big to fit in main memory, is an ever-growing challenge. We have investigated two approaches to address this challenge. One approach involves partitioning the graph into smaller graphs that can be processed in parallel [3]. A second approach involves implementing GBRL within a relational database management system, taking advantage of user-defined functions and the optimized storage capabilities of the RDBMS. These approaches have shown promise in allowing GBRL systems such as Subdue to process arbitrarily large graphs.

A newer issue regarding scalability is what we call *dynamic graphs*. With the advent of real-time streaming data, many data mining systems must mine incrementally, rather than off-line from scratch. The same is true for GBRL systems. Many of the domains we wish to mine in graph form are dynamic domains, e.g., spatio-temporal NASA remote sensing data. We do not have the time to periodically rebuild graphs of all the data to date and run a GBRL system from scratch. We must develop methods to incrementally update the graph and the patterns currently prevalent in the graph. The approach we are currently investigating is similar to the graph partitioning approach for distributed processing. New data can be stored in an increasing number of partitions. Information within partitions can be exchanged, or a re-partitioning can be performed if the information loss exceeds some threshold. GBRL can be used to search the new partitions, suggesting new subgraph patterns as they evaluate highly in new and old partitions.

Similar approaches have been developed to scale ILP systems. For example, the *learning from interpretations* approach [1] takes advantage of the typical disconnected nature of examples, even in relational domains, to make tractable the relational learning and accompanying coverage testing. Comparison of these GBRL and ILP techniques for scalability is a fruitful area of future work.

#### 3.2 Supervised Graphs

Most graph-based data mining approaches (and data mining approaches in general) assume the input data is in the form of transactions, i.e., a set of small, disconnected graphs. At the other extreme is the assumption that the input is one large interconnected graph. The transactional representation allows reduced matching complexity and more straightforward assignment of graphs to classes for supervised learning. However, some data is more naturally represented as one large graph, where the class assignments are made throughout the graph and possibly to varying degrees. We call such a graph a *supervised graph*, in that the graph as a whole contains class information, but is not easily divided into individual classified components. For example, consider a social network in which we seek to find relational patterns

distinguishing various income levels. Individuals of a particular income level can appear anywhere in the graph, and we cannot easily partition the graph into transactions without potentially severing the target relationships. Also, some entities in the graph may have memberships in multiple classes. Such a scenario presents a difficult challenge for future work in graph-based relational learning.

We are investigating two approaches to this task. The first involves modifying the MDL encoding to take into account the amount of information necessary to describe the class membership of compressed portions of the graph. The second approach involves treating the class membership of a vertex or edge as a cost, which can vary from -1 for clearly negative members to +1 for clearly positive members. The information-theoretic value of the subgraph patterns can be weighted by the costs of the instances of the pattern. The ability to learn from supervised graphs will also allow the user more flexibility in indicating class membership where known, and to varying degrees, without having to clearly separate the graph into disjoint examples.

## 4. CONCLUSIONS

Graph-based relational learning is a fast-growing field of data mining due to the increasing interest in mining the relational aspects of graph-oriented data. GBRL is distinct from frequent subgraph mining approaches, because it attempts to identify a small number of subgraph patterns that maximize an information-theoretic metric, rather than finding all subgraphs appearing in a certain percentage of the input graph. GBRL methods differ from ILP methods by leveraging properties unique to graphs versus logic. With over ten years of development, our Subdue approach has become an effective method for learning from graphs. Recent advances in supervised learning and graph-grammar induction have given Subdue capabilities seen in ILP and other GBRL approaches.

However, much work in GBRL remains to be done. Because many of the graph-theoretic operations inherent in GBRL are NP-complete or definitely not in P, scalability is a constant challenge. With the increased need for mining streaming data, the development of new methods for incremental learning from dynamic graphs is important. Also, the blurring of example boundaries in a supervised learning scenario gives rise to a supervised graph, where the class membership of even nearby vertices and edges can vary considerably. We need to develop better methods for learning in these scenarios.

As more and more domains realize the increased predictive power of patterns involving relationships between entities, rather than just attributes of entities, graph-based relational learning and data mining will become foundational to our ability to better understand the ever-increasing amount of data in our world.

## 5. REFERENCES

- [1] H. Blockeel, L. D. Raedt, N. Jacobs, and B. Demoen. Scaling up inductive logic programming by learning from interpretations. *Data Mining and Knowledge Discovery*, 3(1), 1999.
- [2] D. Cook and L. Holder. Graph-based data mining. *IEEE Intelligent Systems*, 15(2):32–41, 2000.
- [3] D. Cook, L. Holder, G. Galal, and R. Maglothin. Approaches to parallel graph-based knowledge discovery. *Journal of Parallel and Distributed Computing*, 61(3):427–446, 2001.
- [4] S. Doshi, F. Huang, and T. Oates. Inferring the structure of graph grammars from data. In *Proceedings of the International Conference on Knowledge-Based Computer Systems*, 2002.
- [5] H. Ehrig, G. Engels, H. Kreowski, and G. Rozenberg, editors. *Handbook of Graph Grammars and Computing by Graph Transformation: Applications, Languages and Tools*. World Scientific, 1999.
- [6] J. Gonzalez, L. Holder, and D. Cook. Graph-based relational concept learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, 2002.
- [7] A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *Proceedings of the European Conference on Principles and Practice of Knowledge Discovery in Databases*, 2000.
- [8] I. Jonyer, L. Holder, and D. Cook. Concept formation using graph grammars. In *Proceedings of the KDD Workshop on Multi-Relational Data Mining*, 2002.
- [9] H. Kashima and A. Inokuchi. Kernels for graph classification. In *Proceedings of the International Workshop on Active Mining*, 2002.
- [10] M. Kuramochi and G. Karypis. Frequent subgraph discovery. In *Proceedings of the First IEEE Conference on Data Mining*, 2001.
- [11] M. Mosbah. Properties of random graphs generated by probabilistic graph grammars. In *Proceedings of the Fifth International Workshop on Graph Grammars and their Application to Computer Science*, 1994.
- [12] S. Muggleton. Duce: An oracle based approach to constructive induction. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, 1987.
- [13] S. Muggleton. Stochastic logic programs. In L. de Raedt, editor, *Advances in Inductive Logic Programming*. IOS Press, 1996.
- [14] S. Muggleton and W. Buntine. Machine invention of first-order predicates by inverting resolution. In *Proceedings of the Fifth International Conference on Machine Learning*, pages 339–352, 1988.
- [15] J. Rissanen. *Stochastic Complexity in Statistical Inquiry*. World Scientific, 1989.
- [16] J. Sowa. *Conceptual Structures: Information in Mind and Machine*. Addison-Wesley, 1984.
- [17] X. Yan and J. Han. gSpan: Graph-based substructure pattern mining. In *Proceedings of the International Conference on Data Mining*, 2002.
- [18] K. Yoshida, H. Motoda, and N. Indurkha. Graph-based induction as a unified learning framework. *Journal of Applied Intelligence*, 4:297–328, 1994.