Comparison of Graph-based and Logic-based Multi-relational Data Mining

Nikhil S. Ketkar
University of Texas at Arlington
ketkar@cse.uta.edu

Lawrence B. Holder
University of Texas at Arlington
holder@cse.uta.edu

Diane J. Cook
University of Texas at Arlington
cook@cse.uta.edu

ABSTRACT

We perform an experimental comparison of the graph-based multi-relational data mining system, Subdue, and the inductive logic programming system, CProgol, on the Mutagenesis dataset and various artificially generated Bongard problems. Experimental results indicate that Subdue can significantly outperform CProgol while discovering structurally large multi-relational concepts. It is also observed that CProgol is better at learning semantically complicated concepts and it tends to use background knowledge more effectively than Subdue. An analysis of the results indicates that the differences in the performance of the systems are a result of the difference in the expressiveness of the logic-based and the graph-based representations. The ability of graph-based systems to learn structurally large concepts comes from the use of a weaker representation whose expressiveness is intermediate between propositional and first-order logic. The use of this weaker representation is advantageous while learning structurally large concepts but it limits the learning of semantically complicated concepts and the utilization background knowledge.

1. INTRODUCTION

Multi-relational data mining (MRDM)[7] is a subfield of data mining which focuses on knowledge discovery from relational databases comprising multiple tables. Representation is a fundamental as well as a critical aspect in the process of discovery and two forms of representation, namely the graph-based representation and the logic-based representation, have been used for MRDM.

Logic-based MRDM popularly known as Inductive Logic Programming (ILP)[16], is the intersection of Machine Learning and Logic Programming. ILP is characterized by the use of logic for the representation of multi-relational data. ILP systems represent examples, background knowledge, hypotheses and target concepts in Horn clause logic. ILP systems such as FOIL [20], CProgol[17], Golem[18], SMART+[4], G-Net[1], CHILLIN[26], TILDE[2] and WARMR[6] have been extensively applied to supervised learning and to a certain extent to unsupervised learning. The core of ILP is the use of logic for representation and the search for syntactically legal hypotheses constructed from predicates provided by the background knowledge. The ILP process is basically a search wherein the states are hypotheses and the goal is the hypothesis that is frequent or which distinguishes positive and negative examples.

An ILP system can be characterized by the way the hypothesis space is structured and the search strategy used to explore the hypothesis space. ILP systems may be classified on the basis of four key factors. ILP systems may learn a single concept or multiple concepts. ILP systems may be batch or incremental depending on how they accept examples. ILP systems may be interactive or non-interactive depending on whether they use human advice in the process of learning. Lastly, ILP systems may revise a theory or may learn concepts from scratch, the former being known as theory revision systems. Although the factors are independent, most ILP systems are either non-interactive, single concept batch learners that build concepts from scratch or are incremental, interactive theory revisers that learn multiple concepts. The former are known as empirical ILP systems while the later are called incremental ILP systems.

Graph-based approaches are characterized by representation of multi- relational data in the form of graphs. Graph-based MRDM systems have been extensively applied to the task of unsupervised learning, popularly known as frequent subgraph mining and to a certain extent to supervised learning. Graph-based approaches represent examples, background knowledge, hypotheses and target concepts as graphs. These approaches include mathematical graph theory based approaches like FSG[14] and gSpan[24], greedy search based approaches like Subdue [5] or GBI[15], and kernel function based approaches[12]. The core of all these approaches is the use of a graph-based representation and the search for graph patterns which are frequent or which compress the input graphs or which distinguish positive and negative examples.

Mathematical graph theory based approaches mine a complete set of subgraphs mainly using a support or frequency measure. The initial work in this area was the AGM[11] system which uses the Apriori level-wise approach. FSG takes a similar approach and further optimizes the algorithm for improved running times. gFSG [13] is a variant of FSG which enumerates all geometric subgraphs from the database. gSpan uses DFS codes for canonical labeling and is much more memory and computationally efficient than previous approaches. Instead of mining all subgraphs, CloseGraph[25] only mines closed subgraphs. A graph G is closed in a dataset if there exists no supergraph of G that has the same support as G. Gaston[19] efficiently mines graph datasets by considering frequent paths. These frequent paths are first transformed to trees and these trees are then transformed to graphs. FFSM[9] is a graph mining system which uses an algebraic graph framework to address the underlying problem of subgraph isomorphism.

In comparison to mathematical graph theory based approaches which are complete, greedy search based approaches use heuristics to evaluate the solution. The two pioneering works in the field are Subdue and GBI. Subdue uses MDL-based compression heuristics, and GBI uses an empirical graph size-based heuristic. The empirical graph size definition depends on the size of the extracted patterns and the size of the compressed graph.

Lastly, the kernel function based approaches have been used to a certain extent for mining graph datasets. The kernel function defines a similarity between two graphs. When high dimensional data is represented in linear space, the function to learn is difficult in that space. We can map the linear data to nonlinear space and the problem of learning in that high dimensional space becomes the learning of scalar products. Kernel functions make computation of such scalar products very efficient. The key to applying the kernel function based approach to mining graph data is finding efficient mapping functions and good feature vectors. The pioneering work that applied kernel functions to graph structures is the diffusion kernel [12].

We perform an experimental comparison of graph-based and logic-based MRDM. We identify three key factors for comparing graph-based and logic-based multi-relational data mining; namely, the ability to discover structurally large concepts, the ability to discover semantically complicated concepts and the ability to effectively utilize background knowledge. CProgol is selected as a representative of logic-based approaches and Subdue is selected as a representative of graph-based approaches. Experiments are performed on the Mutagenesis dataset which is a benchmark dataset for MRDM. In most of the experiments, transformations are applied to the Mutagenesis dataset or distinct types of background knowledge are provided to Subdue and CProgol. The rationale behind doing so is to perform lesion studies and gain insight on the specific abilities of the approaches. Additional experiments are performed on artificially generated Bongard problems to reinforce the findings from the experiments on the Mutagenesis dataset. We analyze the experimental results and present the insights and future directions drawn from them.

Our analysis of the experimental results indicates that the differences in the performance of the systems are a result of the difference in the expressiveness of the logic-based and the graph-based representations. The insights from this comparison are pertinent to various tasks involving learning from multi-relational data like inductive databases[10] and link mining[8].

The rest of the paper is organized as follows. In Section 2 we describe the experimental setup, comprising of the logic-based MRDM system CProgol, the graph-based MRDM system Subdue, the Mutagenesis dataset and the Bongard problems used for artificial experiments. In Section 3, we identify the factors for comparing graph-based and logic-based MRDM. Section 4 describes the experiments with the Mutagenesis dataset. Sections 5 describes the artificial domain experiments with Bongard problems. In Section 6, we analyze the experimental results. Conclusions and future work are presented in Section 7.

2. EXPERIMENTAL SETUP

In this section we describe the experimental setup, comprising of the logic-based MRDM system CProgol, the graph-based MRDM system Subdue, the Mutagenesis dataset and the Bongard problems used for artificial experiments.

2.1 CProgol

CProgol[17] is an ILP system, characterized by the use of mode-directed inverse entailment and a hybrid search mechanism. Inverse entailment is a procedure which generates a single, most specific clause that, together with the background knowledge, entails the observed data. The inverse entailment in CProgol is mode-directed that is, it uses mode definitions. A mode declaration is a constraint which imposes restrictions on the atoms and their arguments appearing in a hypothesis clause by,

1. Determining which atoms can occur in the head and the body of hypotheses.
2. Determining which arguments can be input variables, output variables or constants.
3. Determining the number of alternative solutions for instantiating the atom.

The user-defined mode declarations aid the generation of the most specific clause. CProgol first computes the most specific clause which covers the seed example and belongs to the hypothesis language. The most specific clause can be used to bound the search from below. The search is now bounded between the empty clause and the most specific clause. The search proceeds within the bounded θ-subsumption lattice in a general-to-specific manner. The search is a hybrid search, because it is a general-to-specific search bounded from below with respect to the most specific clause. The search strategy is an A* algorithm which is guided by a weighted compression and accuracy measure. The A* search returns a clause which covers the most positive examples and maximally compresses the data.

Any arbitrary Prolog program can serve as background knowledge for CProgol. The mode definitions and the background knowledge together define a hypothesis language. The hypothesis space explored by CProgol consists of every hypothesis defined by the hypothesis language.

2.2 Subdue

Subdue[5] is a graph-based MRDM system capable of unsupervised and supervised learning. When operating as a supervised learner, Subdue accepts a set of example graphs labeled as positive or negative and finds subgraphs distinguishing the positive graphs from the negative graphs. The hypothesis space of Subdue consists of all the connected subgraphs of all the example graphs labeled positive.

Subdue performs a beam search which begins from substructures consisting of all vertices with unique labels. The substructures are extended by one vertex and one edge or one edge in all possible ways, as guided by the input graph, to generate candidate substructures. Subdue maintains the instances of substructures (in order to avoid subgraph isomorphism) and uses graph isomorphism to determine the instances of the candidate substructure in the input graph. Candidate substructures are evaluated according to classification accuracy or the minimum description length principle

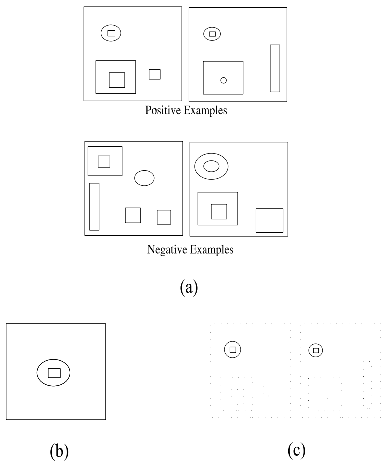


Figure 1: Bongard Problems. (a) Example of a Bongard Problem. (b) Concept to be learned in the Bongard problem. (c) Instances of the concept in the Bongard problem.

[22]. The length of the search beam determines the number of candidate substructures retained for further expansion. This procedure repeats until all substructures are considered or the user imposed computational constraints are exceeded. At the end of this procedure the positive examples covered by the best substructure are removed. The process of finding substructures and removing positive examples continues until all the positive examples are covered.

Subdue can be provided with predefined substructures as background knowledge. Subdue uses this background knowledge by preprocessing the examples and compressing each of the user defined substructures which form the background knowledge into a single vertex.

2.3 Mutagenesis Dataset

The Mutagenesis dataset[23] has been collected to identify mutagenic activity in a compound based on its molecular structure and is considered to be a benchmark dataset for MRDM. The Mutagenesis dataset consists of the molecular structure of 230 compounds, of which 138 are labeled as mutagenic and 92 as non-mutagenic. The mutagenicity of the compounds has been determined by the Ames Test. The task is to distinguish mutagenic compounds from non-mutagenic ones based on their molecular structure. The Mutagenesis dataset basically consists of atoms, bonds, atom types, bond types and partial charges on atoms. The dataset also consists of the hydrophobicity of the compound (logP), the energy level of the compound's lowest unoccupied molecular orbital (LUMO), a boolean attribute identifying compounds with 3 or more benzyl rings (I1), and a boolean attribute identifying compounds which are acenthryles (Ia). Ia, I1, logP and LUMO are relevant properties in determining mutagenicity.

2.4 Bongard Problems

Bongard problems[3] were introduced as an artificial domain in the field of pattern recognition. A simplified form of Bongard problems has been used as an artificial domain in the field of ILP[21]. We use a similar form of Bongard problems for our artificial domain experiments. We use a Bongard problem generator to generate datasets. Each dataset consists of a set of positive and negative examples. Each example consists of a number of simple geometrical objects placed inside one another. The task is to determine the particular set of objects, their shapes and their placement which can correctly distinguish the positive examples from the negative ones. Figure 1(a) shows a Bongard problem, while (b) and (c) show the concept to be learned and their instances in the positive examples respectively.

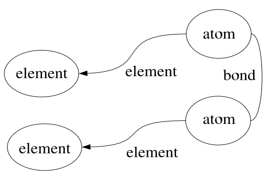
3. FACTORS FOR COMPARISON

By performing a comparison of the graph-based and logic-based approaches to MRDM, we intended to analyze the ability of the approaches to efficiently discover complex multi-relational concepts and to effectively utilize background knowledge. For doing so it is essential to establish some intuitive notions on the complexity of a multi-relational concept and to identify the types of background knowledge generally available in the task of MRDM.

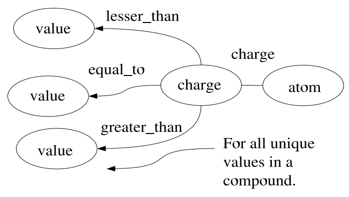
The complexity of a multi-relational concept is a direct consequence of the number of relations in the concept. A multi-relational concept is more complicated to learn than some other multi-relational concept if learning that concept involves learning more relations than the other concept. For example learning the concept of arene (six member ring as in benzene) which comprises learning six relations, involves the exploration of a larger hypothesis space than learning the concept of hydroxyl (oxygen connected to hydrogen as in methanol), which comprises learning one relation. The concept of arene is thus more complicated than that of hydroxyl. Although the number of relations in the multi-relational concept is a key factor in the complexity of the multi-relational concept, there are also other factors such as the number of relations in the examples from which the concept is to be learned. For example, learning the concept of hydroxyl from a set of large molecules (e.g., phenols, etc.) involves the exploration of a larger hypothesis space than learning the same hydroxyl concept from a set of small molecules (e.g., methanol, etc.). The concept of hydroxyl group is thus more complicated to learn from phenols than it is from a set of alcohols. We identify this complexity as *structural complexity*.

In order to learn a particular concept, it is essential that the representation used by a multi-relational data mining system is able to express that particular concept. For a representation to express a particular concept, it is beneficial to have both the syntax which expresses the concept and the semantics which associates meaning to the syntax. The concepts which cannot be represented by the representation used by the MRDM system can be explicitly instantiated in the examples . A relational concept can be said to have a higher complexity than some other relational concept if representing that concept requires a more expressive representation. For example to learn numerical ranges, it is essential to have the syntax and the semantics for representing notions like 'lesser than', 'greater than' and 'equal to'. We identify this complexity as *semantic complexity.*

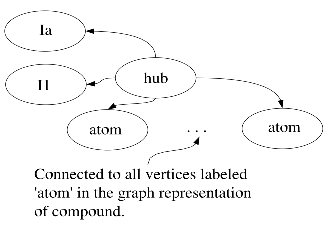
A relational learner can be provided background knowledge which condenses the hypothesis space. For example if the concept to be learned is 'compounds with three arene rings' (six member ring as in benzene) and the concept of an arene ring is provided as a part of the background knowledge, then



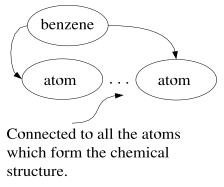
(a)



(b)


Ia
I1
hub
atom
. . .
atom
Connected to all vertices labeled 'atom' in the graph representation of compound.

(c)


benzene
atom
. . .
atom
Connected to all the atoms which form the chemical structure.

(d)

```
atom(example_id,atom_id,element).
bond(example_id,atom_id,atom_id).
```

(e)

```
gteq(X,Y):-                  lteq(X,Y):-
    not(var(X)),                 not(var(X)),
    not(var(Y)),                 not(var(Y)),
    float(X), float(Y),          float(X), float(Y),
    X >= Y, !.                   X =< Y, !.
gteq(X,X):-                  lteq(X,X):-
    not(var(X)),                 not(var(X)),
    float(X).                    float(X).
```

(f)

```
ind1(example_id,boolean).
inda(example_id,boolean).
```

(g)

```
benzene(example_id,[atom_id,atom_id,...]).
```

(h)

Figure 2: Representations of the Mutagenesis dataset. (a) Graph-based representation while comparing the ability to learn structurally large concepts. (b) Graph-based representation while comparing the ability to learn semantically complicated concepts. (c) Graph-based representation while comparing the ability to utilize background knowledge in the form of indicator variables. (d) Graph-based representation while comparing the ability to utilize background knowledge in the form of generic chemical concepts. (e) Logic-based representation while comparing the ability to learn structurally large concepts. (f) Logic-based representation while comparing the ability to learn semantically complicated concepts. (g) Logic-based representation while comparing the ability to utilize background knowledge in the form of indicator variables. (h) Logic-based representation while comparing the ability to utilize background knowledge in the form of generic chemical concepts.

the arene rings in examples could be condensed to a single entity. This would cause a massive reduction in the hypothesis space required to be explored to learn the concept and the relational learner would perform more efficiently than without the background knowledge. We identify such background knowledge as *background knowledge intended to condense the hypothesis space.*

A relational learner can be provided background knowledge which augments the hypothesis space. For example consider that the relational learner is provided with background knowledge which allows it to learn concepts like 'lesser than','greater than' and 'equal to'. In this case, the relational learner would explore a hypothesis space larger than what it would explore without the background knowledge. Thus introducing background knowledge has augmented the hypothesis space and has facilitated the learning of concepts which would not be learned without the background knowledge. We identify such background knowledge as *background knowledge intended to augment the hypothesis space.*

Using these notions, we can identify three key factors for comparing graph-based and logic-based multi-relational data mining,

1. The ability to discover structurally large concepts.
2. The ability to discover semantically complicated concepts (or the ability to utilize background knowledge which augments the hypothesis space).
3. The ability to effectively utilize background knowledge (which condenses the hypothesis space).

4. EXPERIMENTS ON THE MUTAGENESIS DATASET

In this section we present our experiments on the Mutagenesis Dataset. We compared the performance of Subdue and CProgol on four different representations of the Mutagenesis Dataset. Each representation was selected in order to analyze a specific ability of the systems.

In first case, we compare the ability of the approaches to learn large structural concepts. Both the relational learners were provided only with the basic information of the atoms, the elements and the bonds without any other information or background knowledge. The relational learners are not provided with any additional information or any form of background knowledge, because we intended to compare the ability to learn large structural concepts. The partial charge information was not provided to either system because this information would contribute to the accuracy and make it difficult to analyze how the approaches compare while learning structurally large concepts. The atom type and bond type information was also not provided to either system. The reasoning behind doing so is that we view the atom type and bond type information as a propositional representation of relational data. Such information allows the relational learners to learn propositional representations of relational concepts instead of the true relational concept.

Consider for example the rule found by CProgol on the Mutagenesis dataset[23],atom(A,B,c,195,C). This rule denotes that compounds with a carbon atom of type 195 are mutagenic. The atom type 195 occurs as the atom shared by 3 fused rings 6 member rings. Therefore all compounds with 3 fused 6 member rings are labeled active. It is interest-

Table 1: Results on experiments with the Mutagenesis dataset.

	Structurally Complicated Concepts		Semantically Complicated Concepts		Indicator Variables		Generic Chemical Concepts	
	CProgol	Subdue	CProgol	Subdue	CProgol	Subdue	CProgol	Subdue
Training Set Accuracy	60.00%	86.00%	67.00%	64.00%	82.00%	80.00%	62.00%	64.00%
Training Set Runtime	2010s	1876s	1180s	2876s	960s	848s	2130s	1910s
10-fold CV Accuracy	61.74%	81.58%	66.53%	63.91%	78.91%	77.39%	61.74%	63.84%
10-fold CV Runtime (average)	1940s	2100s	1330s	2900s	810s	878s	2212s	2010s
CProgol - Subdue, ΔError$\pm\sigma$	20.84%$\pm$12.78%		2.16%$\pm$3.5%		1.52%$\pm$11.54%		1.74%$\pm$25.12%	
CProgol - Subdue, Confidence	99.94%		95.77%		31.38%		16.86%	

ing to note that a rule involving 15 relations (3 fused 6 member rings) has been learned by learning a single relation. Learning such a rule has allowed CProgol to learn a propositional representation of a relational concept rather that the true relational concept. Providing atom type and bond type information would allow both systems to learn propositional representations of structurally large relational concepts rather than the true relational concepts. We do not consider the learning of such concepts equivalent to the learning of structurally large relational concepts. We therefore do not provide either system with the atom type and bond type information. This is depicted in Figure 2(a) and (e).

In the second case we compare the performance of the systems while learning semantically complicated concepts, we ran Subdue and CProgol on the Mutagenesis dataset. Each system was provided with background knowledge so that numerical ranges could be learned. For CProgol this was achieved by introducing Prolog based background knowledge. For Subdue this was achieved by explicitly instantiating the background knowledge, i.e., additional structure was added to the training examples. This is depicted in Figure 2(b) and (f).

In the third case we provide each system with the background knowledge indicating the presence of benzyl rings (I1) and identifying compounds which are acenthryles (Ia). This is depicted in Figure 2(c) and (g).

In the fourth case, each system was provided with the background knowledge indicating certain generic chemical concepts like benzene rings, nitro groups, etc. This is depicted in Figure 2(d) and (h).

The results of these experiments are shown in Table 1. For the training set, the accuracy for one run on the entire dataset and the learning time are shown. For 10-fold cross validation (CV), average learning time over 10 folds is shown.

From these experiments we can observe the following.

1. Subdue performs significantly better than CProgol in the first case, i.e. while learning structurally large concepts.

2. CProgol outperformed Subdue in the second case, i.e. while learning semantically complicated concepts.

3. In the third and the fourth cases i.e. while utilizing background knowledge in the form of indicator variables and generic chemical concepts respectively, the performance the systems is found to be comparable.

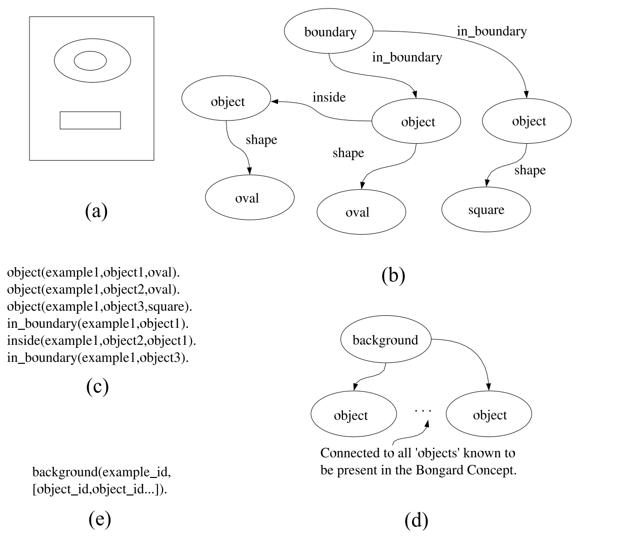
It must be noted that in the second case i.e while learning semantically complicated concepts, Subdue has achieved an accuracy lower than what was achieved without the background knowledge to process ranges. Manual inspection of the concepts learned by Subdue and CProgol (not shown here) indicate that Subdue did not learn any concept involving ranges as learned by CProgol. We performed additional experiments (not reported here) with different graph-based representations for learning ranges. In all these experiments, Subdue had similar performance.

5. ARTIFICIAL DOMAIN EXPERIMENTS

We performed additional experiments using artificially generated Bongard problems to reinforce the insights from the experiments on the Mutagenesis dataset. We now discuss the results of these experiments.

We systematically analyzed the performance of Subdue and CProgol on artificially generated Bongard problems with increasing numbers of objects in the concept and increasing numbers of objects in the examples in order to compare the abilities of the systems to learn structurally large concepts. Figure 3(a) shows a Bongard example, while (b) and (c) show the graph-based and logic-based representations of the example, respectively. In this experiment, the number of objects in the Bongard concept was varied from 5 to 35. The number of additional objects in each example (objects which are not a part of the concept) were kept constant at 5. For every concept size from 5 to 35, 10 different concepts were generated. For each of the 10 concepts a training set and a test set of 100 positive and 100 negative examples was generated.

Figure 4(a) shows the average accuracy achieved by CProgol and Subdue on 10 datasets for every concept size ranging from 5 to 35. In order to further analyze the performance of the systems we reran the same experiment but in this case the systems were iteratively given increased resources (this was achieved by varying the nodes parameter in CProgol and the limit parameter in Subdue) so that we could determine the number of hypotheses each system explored before it learned the concept (a cutoff accuracy of 80% was decided). Figure 4(b) shows the number of hypotheses explored by each system so as to achieve an accuracy of 80% (this experiment was only performed for concept size varying from 5 to 18 as a significantly large amount of time was required). A snapshot of the experiment (Accuracy vs. Number of Explored Hypotheses) for concept size 10 is shown in Figure 4(c). A similar experiment for increased example size was performed where the concept size was kept constant at 5 and the example size was varied from 10 to 35. Figure 4(d)



```
object(example1,object1,oval).
object(example1,object2,oval).
object(example1,object3,square).
in_boundary(example1,object1).
inside(example1,object2,object1).
in_boundary(example1,object3).
```

```
background(example_id,
[object_id,object_id...]).
```

Figure 3: Representation for Bongard Problems. (a) A Bongard example. (b) Graph-based representation of the Bongard problem. (c) Logic-based representation of the Bongard problem. (d) Graph-based representation for the introduction of background knowledge in Bongard problems. (e) Logic-based representation for the introduction of background knowledge in Bongard problems.

shows the average accuracy achieved by CProgol and Subdue on 10 datasets for every example size ranging from 10 to 35. Figure 4(e) shows the hypotheses required to be explored to learn the concept (a cutoff accuracy of 80% was decided) determined by iteratively increasing the resources for each system. A snapshot of the experiment (Accuracy vs. Number of Explored Hypotheses) for example size 15 is shown in Figure 4(f).

Experimental results indicate that,

1. Subdue achieves increasingly higher accuracy than CProgol as the size of the concept (number of relations in the concept) grows.
2. CProgol has to explore an increasingly larger number of hypotheses than Subdue as the size of the concept grows.
3. Subdue achieves increasing higher accuracy than CProgol as more hypotheses are explored, on large concept sizes.
4. Subdue achieves increasingly higher accuracy than CProgol as the size of the examples (number of relations in the example) grows.
5. CProgol has to explore an increasingly larger number of hypotheses than Subdue while learning from increasingly larger examples.
6. Subdue achieves increasing higher accuracy than CProgol as more hypotheses are explored, on large example sizes.

Experiments were also performed to analyze the ability to utilize background knowledge. Figure 3(d) and (e) show the representations used for CProgol and Subdue respectively. We systematically analyzed the performance of Subdue and CProgol on artificially generated Bongard problems with increasing amounts of background knowledge while learning a large concept (more objects in the concept) and with increasing amounts of background knowledge while learning a concept from a large example (more objects in each example). In this experiment, for a concept of size 10 and additional objects in each example equal to 5, 10 concepts were generated. For each of these concepts a training set and test set of 100 positive and 100 negative examples were generated. Figure 4(g) shows the accuracies achieved by Subdue and CProgol (Note that both the systems were given less resources than the experiments which analyzed the ability to learn structurally large concepts so that the effect of background knowledge could be analyzed). Figure 4(h) shows the hypotheses required by each system to learn the concept (a cutoff accuracy of 80% was decided). A similar experiment for a concept of size 5 and example size of 15 was performed. Figure 4(i) shows the accuracies achieved by Subdue and CProgol (Note that both the systems were given less resources than the experiments which analyzed the ability to learn structurally large concepts so that the effect of background knowledge could be analyzed). Figure 4(j) shows the hypotheses required by each system to learn the concept (a cutoff accuracy of 80% was decided).
Experimental results indicate that when increasing amounts of background knowledge are introduced,

1. CProgol achieves increasingly higher accuracy than Subdue in the case of large concepts,(number of relations in the concept).
2. Subdue has to explore an increasingly larger number of hypotheses than CProgol in the case of large concepts.
3. CProgol achieves increasingly higher accuracy than Subdue in the case of large examples (number of relations in the example).
4. Subdue has to explore an increasingly larger number of hypotheses than CProgol in the case of large examples.

6. ANALYSIS

The results of the experiments performed in Sections 4 and 5 indicate that Subdue significantly outperforms CProgol while learning structurally large concepts. It is also observed that CProgol performs better than Subdue while learning semantically complicated concepts and utilizing background knowledge. Here we first attempt to explain the empirical results based on the representational and algorithmic differences between the two systems and then analyze whether these findings are applicable, in general to all graph-based and logic-based approaches.
An analysis of the representations used by Subdue and CProgol indicates that the graph-based representation used by Subdue has much less expressiveness than that of CProgol which can accept any Prolog program as background knowledge. The expressiveness of the graph-based representation used by Subdue is intermediate between propositional and first-order logic. This difference can explain the differences between the performances of the two systems. Learning structurally large concepts involves learning

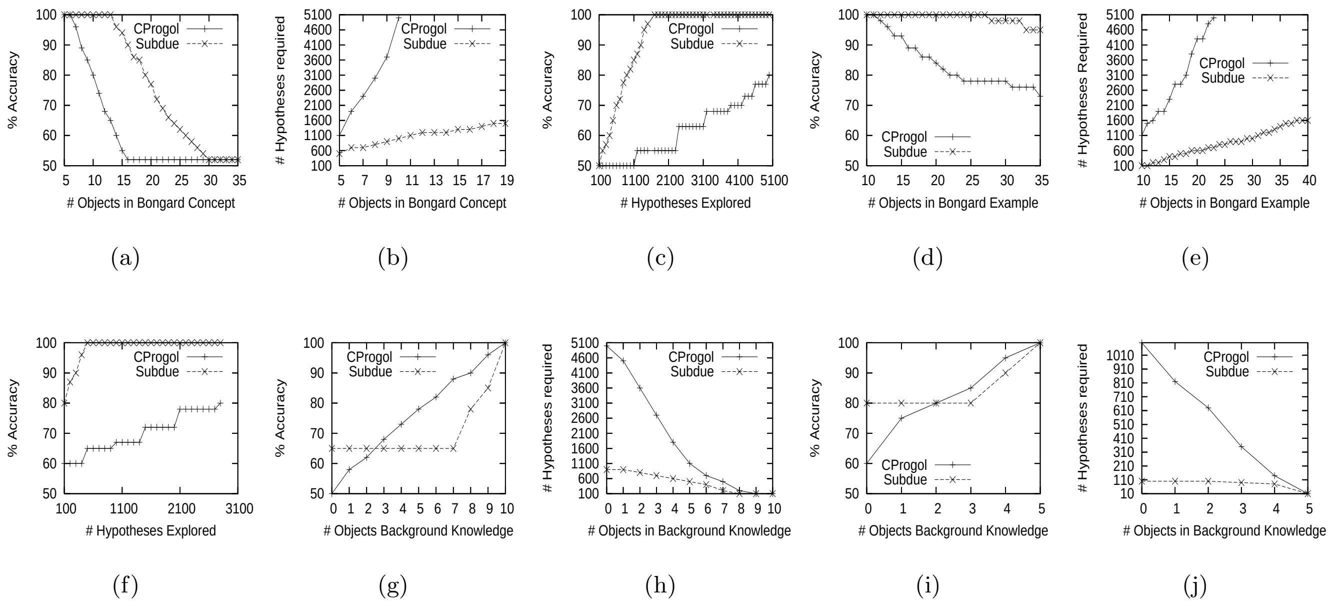


Figure 4: Results on experiments with artificially generated Bongard problems. (a) Accuracy vs Concept Size. (b) Hypotheses required to reach accuracy of 80% vs Concept size. (c) Accuracy vs Number of hypotheses explored, for fixed concept size. (d) Accuracy vs Example Size. (e) Hypotheses required to reach accuracy of 80% vs Example Size. (f) Accuracy vs Number of hypotheses explored, for fixed example size. (g) Accuracy vs Amount of background knowledge, for large concept size. (h) Hypotheses required to reach accuracy of 80% vs Amount of background knowledge, for large concept size. (i) Accuracy vs Amount of background knowledge,for large example size. (j) Hypotheses required to reach accuracy of 80% vs Amount of background knowledge, for large example size.

a large number of relations explicitly present in the examples. In this case, the less expressive representation used by Subdue leads to an efficient exploration of the hypothesis space. An example of this is the massive number of redundant hypothesis which are not generated and evaluated by Subdue.

Learning semantically complicated concepts involves learning relations which are not explicitly present in the examples but which must be implicitly derived from the examples. In this case CProgol which uses a more expressive representation not only performs more efficiently but also can learn concepts which may not be expressed by Subdue's mechanism of explicit instantiation. An example of this is Subdue's explicit instantiation to learn ranges in Figure 2(b). Explicit instantiation is cumbersome in most cases and also not a generalized methodology to learn complicated semantic concepts. For example, suppose a domain expert were to suggest that the ratio of the number of carbon atoms to the number of hydrogen atoms in a molecule has an effect on the mutagenicity. CProgol with some added background knowledge could use this information to classify the molecules. Subdue on the other hand would require making changes to the representation such that the pattern would be found in terms of a graph. CProgol allows the exploration of hypotheses through implicitly defined background knowledge rather than explicit instantiation in the examples.

The difference between the expressiveness of the representations can also explain the difference in the abilities of the systems to utilize background knowledge. In Subdue, background knowledge is introduced as a vertex which is connected to all the entities which comprise the background knowledge for example,in the Mutagenesis dataset as in Figure 2(d) and Bongard problems as in Figure 3(d). In the case of CProgol, similar background knowledge is introduced in the form of a predicate, as in Figure 2(h). Subdue's graph-based representation does not allow any specific mechanism to express that a set of entities (objects and atoms in the case of Bongard problems and Mutagenesis respectively) are known to be a part of a group which is relevant in classifying the examples. Such a group of entities is to be added in a single refinement step to generate candidate hypothesis, and benefit from the background knowledge. Subdue generates candidate hypotheses by extending the sub-graph by an edge and a vertex or just an edge in all possible ways as in the examples. Subdue's candidate hypothesis generation adds the entities known to be a part of the background knowledge one at a time and this leads to a massive increase in the hypothesis space. This may cause the introduction of background knowledge to deteriorate performance in some cases.

Subdue does provide an alternate way of introducing background knowledge by preprocessing the examples and compressing each of the user defined substructures which form the background knowledge into a single vertex. This technique has the drawback of information loss and Subdue will not be able to learn a concepts that contain only a partial portion of the background knowledge substructure.

The differences between the performance of the two systems can thus be explained by the difference in the expressiveness of the representations used by the two systems. Any graph-based and logic-based system will tend to behave similarly due to this underlying difference in expressiveness. The use of a less expressive representation will facilitate an efficient search and lead to a superior performance while learning structurally large concepts. The use of a weaker representation would also limit the learning of semantically complicated concepts and the effective use of background knowledge. It is important to note that these characteristics are the result of the difference in the expressiveness of the two representations and are not inherent to graph-based and logic-based representations in general. It would be possible to introduce the syntax and semantics in graph-based representations to express semantically complicated concepts using ordered graphs, hyper-graphs or graph rewriting rules. In this case, a graph-based system would tend to have a performance similar to a logic-based system.

7. CONCLUSIONS AND FUTURE WORK

We performed an experimental comparison of the graph-based multi-relational data mining system, Subdue, and the inductive logic programming system, CProgol. From this comparison we conclude that the use of a less expressive representation, like the present graph-based representation can achieve superior performance while learning structurally large multi-relational concepts. The use of a less expressive representation limits the learning of semantically complicated multi-relational concepts and the utilization of background knowledge.

Developing methodologies for using the less expressive graph-based systems and more expressive logic-based systems in combination may achieve the learning of structurally large concepts and semantically complicated concepts. It may also allow better utilization of background knowledge. For example, the structurally complicated hypotheses learned by a graph-based system could then be used as background knowledge by a logic-based system. We plan to pursue this as part of our future work.

8. ACKNOWLEDGMENTS

This research is sponsored by the Air Force Research Laboratory (AFRL) under contract F30602-01-2-0570. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of AFRL or the United States Government.

9. REFERENCES

[1] C. Anglano, A. Giordana, G. L. Bello, and L. Saitta. An experimental evaluation of coevolutive concept learning. In *ICML*, pages 19–27, 1998.

[2] H. Blockeel and L. D. Raedt. Top-down induction of first-order logical decision trees. *Artif. Intell.*, 101(1-2):285–297, 1998.

[3] M. Bongard. *Pattern Recognition.* Spartan Books, 1970.

[4] M. Botta and A. Giordana. Smart+: A multi-strategy learning tool. In *IJCAI*, pages 937–945, 1993.

[5] D. J. Cook and L. B. Holder. Substructure discovery using minimum description length and background knowledge. *J. Artif. Intell. Res. (JAIR)*, 1:231–255, 1994.

[6] L. Dehaspe and H. Toivonen. Discovery of frequent datalog patterns. *Data Min. Knowl. Discov.*, 3(1):7–36, 1999.

[7] S. Dzeroski. Multi-relational data mining: an introduction. *SIGKDD Explorations*, 5(1):1–16, 2003.

[8] L. Getoor. Link mining: a new data mining challenge. *SIGKDD Explorations*, 5(1):84–89, 2003.

[9] J. Huan, W. Wang, and J. Prins. Efficient mining of frequent subgraphs in the presence of isomorphism. In *ICDM*, pages 549–552, 2003.

[10] T. Imielinski and H. Mannila. A database perspective on knowledge discovery. *Commun. ACM*, 39(11):58–64, 1996.

[11] A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *PKDD*, pages 13–23, 2000.

[12] R. I. Kondor and J. D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *ICML*, pages 315–322, 2002.

[13] M. Kuramochi and G. Karypis. Discovering frequent geometric subgraphs. In *ICDM*, pages 258–265, 2002.

[14] M. Kuramochi and G. Karypis. An efficient algorithm for discovering frequent subgraphs. *IEEE Trans. Knowl. Data Eng.*, 16(9):1038–1051, 2004.

[15] T. Matsuda, T. Horiuchi, H. Motoda, and T. Washio. Extension of graph-based induction for general graph structured data. In *PAKDD*, pages 420–431, 2000.

[16] S. Muggleton. Inductive logic programming. *New Generation Comput.*, 8(4):295–, 1991.

[17] S. Muggleton. Inverse entailment and progol. *New Generation Comput.*, 13(3&4):245–286, 1995.

[18] S. Muggleton and C. Feng. Efficient induction of logic programs. In *ALT*, pages 368–381, 1990.

[19] S. Nijssen and J. N. Kok. A quickstart in frequent structure mining can make a difference. In *KDD*, pages 647–652, 2004.

[20] J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.

[21] L. D. Raedt and W. V. Laer. Inductive constraint logic. In *ALT*, pages 80–94, 1995.

[22] J. Rissanen. *Sochastic Complexity in Statistical Inquiry.* Singapore: World Scientific Publishing, 1989.

[23] A. Srinivasan, S. Muggleton, M. J. E. Sternberg, and R. D. King. Theories for mutagenicity: A study in first-order and feature-based induction. *Artif. Intell.*, 85(1-2):277–299, 1996.

[24] X. Yan and J. Han. gspan: Graph-based substructure pattern mining. In *ICDM*, pages 721–724, 2002.

[25] X. Yan and J. Han. Closegraph: mining closed frequent graph patterns. In *KDD*, pages 286–295, 2003.

[26] J. M. Zelle, R. J. Mooney, and J. B. Konvisser. Combining top-down and bottom-up techniques in inductive logic programming. In *ICML*, pages 343–351, 1994.