Issues of Verification for Unsupervised Discovery Systems

Shou-de Lin
Information Science Institute
University of Southern California
sdlin@isi.edu

Hans Chalupsky
Information Science Institute
University of Southern California
hans@isi.edu

ABSTRACT

Developing systems that perform discovery presents unique challenges, e.g., when compared to learning programs, since in general there is no teacher or example library available to train or evaluate a discovery system. In particular, the lack of gold-standard methods or examples makes the verification of discovered results and evaluation of system performance a very difficult problem. In this paper we address several issues of verifying a machine discovery system and discuss what one should and should not expect from the evaluation report of a discovery system. Recognizing that there is no direct way to verify the validity of a true discovery system, this paper proposes several indirect strategies one can adopt to assess the validity of discovered results. We then present our own set of novel link discovery tools as a case study to show how the proposed concepts can be applied to verify a real-world discovery system. In a time where computer science research has become extremely evaluation driven, researchers sometimes shy away from areas where results are difficult to evaluate. By discussing these issues with respect to discovery systems, we hope to provide a useful overview as well as – hopefully - a positive and encouraging signal to attract more researchers to work on machine discovery problems.

1. INTRODUCTION

Discovery is, by definition[1], a procedure of finding out or ascertaining something previously unknown or unrecognized. Since the products of discovery are previously unknown or unrecognized, it is usually non-trivial to convince ourselves and others whether the discoveries are truly correct. In fact, many scientific discoveries (e.g. Copernicus' statement that earth is not the center of the universe, or Fermat's Last Theorem) do take years or even centuries to be proven true or false.

Machine discovery has been an important research area of artificial intelligence (AI) for more than twenty years. Herbert Simon described it as "gradual problem-solving processes of searching large problem spaces for **incompletely defined goal objects**" [27]. The majority of traditional machine discovery programs (e.g. AM [15] and BACON [14]) focus on discovering (or rediscovering) theories and laws in natural science. These programs usually rely on some pre-requisite knowledge in a specific domain and some more general knowledge to guide the search (e.g. heuristics).

More recently, researchers have encountered another problem: there is more and more data available to us and we do not know how to make use of it. This started a new type of discovery research called knowledge discovery and data mining (KDD) that mainly focuses on discovering and extracting previously unknown, valid, novel, potentially useful and understandable patterns from lower-level data [8].

[1] *The American Heritage® Dictionary of the English Language*

Researchers have developed various kinds of discovery systems that utilize many different approaches. Nevertheless, one can divide them into two general categories: "supervised discovery systems" and "unsupervised discovery systems". Most systems that aim at predicting a future trend based on history data belong to the first category [2]. These systems are closer to supervised learning systems, since they have (historic) examples available for training. On the other hand, there are some discovery systems that naturally do not (and cannot) have training examples to use. For example, ARROWSMITH [31] is a literature-based discovery tool that hypothesizes the possible treatments of medical diseases. This type of system needs to be an unsupervised discovery system, since it targets diseases whose treatments are not developed yet.

In general, machine discovery research faces the same verification problems as human discovery. Due to its incompletely defined goal objects, there is no universal way or "gold standard" to determine whether or how well a goal has been achieved. Nevertheless, the verification of supervised discovery systems is relatively easier than that of unsupervised systems. To verify a supervised system, one usually performs the same cross-validation as for a typical learning system. That is, a certain amount of history training data is held out and the system is then evaluated by seeing how well it performs on the held-out data. This means one can evaluate the system by standard recall and precision and related measures. Therefore, verifying a supervised discovery system (or a prediction system) is very similar to verifying a typical supervised learning system. The basic assumption underlying this verification strategy is that the distribution of future unseen data is the same or very similar to the distribution of the history [2]. Said differently, if we can use a part of the history to predict the other part of the history, then we should also be able to use the history to predict the future.

This paper, however, focuses on the verification of unsupervised discovery systems where no training examples or historic data are available. The rest of the paper is organized as follows: in Section 2 we describe several indirect strategies to evaluate discovery systems. These are rediscovery, explanation-based discovery, exploiting independent resources, minimum description length and unexpectedness measures. In Section 3 we show how recall and precision measures play a different role in machine discovery compared with machine learning. In Section 4 we use the verification of a set of novel link discovery tools as a case study to demonstrate how the methodologies described here can be applied to verify a real-world unsupervised discovery system. Section 5 presents concluding remarks.

2. VERIFYING DISCOVERY SYSTEMS

In this section we describe several independent strategies to verify an unsupervised discovery system. For each strategy, we will address its strength and as well as some potential weaknesses.

2.1 Rediscovery

Since for a true discovery system there is no "gold standard" available to check the validity of discovered results, one strategy we can use in some cases is to test whether the system works on some domain that we already have some knowledge about. Rediscovery is an *indirect* verification procedure in the sense that we are testing the validity and generality of the *program* instead of the discovered results. Using rediscovery as a verification strategy is supported by the following reasoning: if the program is general enough to perform discovery successfully in certain rediscovery domains, and these are reasonably similar to the original or target domain, then there is a good chance that the program will also generate good results in the target domain. Often the rediscovery and target domains are the same and rediscovery simply means – as its name implies – automatic discovery of previously human-discovered results.

Most AI researchers working on scientific discovery adopt this strategy to verify the value of their discovery system. Lenat justifies his heuristic discovery system AM by showing that it can rediscover various mathematic laws and concepts such as natural numbers and Goldbach's conjecture [15]. Pericliev and Valdes-Perez justify their maximally parsimonious discrimination program by rediscovering several linguistic phenomena such as the structure of kinship-related terms in Seneca (originally found by Lounsbury in 1964) [32] as well as Greenburg's language universal rules [22].

The most important concern we have with rediscovery is with the amount, representation and use of background knowledge in the rediscovery domain provided to the program. The more domain-specific knowledge is encoded, the harder it is to make the claim that the discovery is a product of our program rather than already "built-in" via background knowledge which might be missing in the target domain. This is particularly difficult for systems that do need a significant amount of background knowledge to work successfully, such as, for example, scientific discovery systems.

Another concern is with the claim that "since the discovery program works for domain X, it should also work for domain Y" which is only valid if the two domains are fairly similar. The more different they are, the weaker this analogy becomes. Even if rediscovery and target domains are the same, say mathematics, it is not known whether past discovered results are in any way similar to any as yet undiscovered results to be found by the program.

2.2 Explanation-Based Discovery

The idea of explanation-based discovery is to use natural language (or some other easily understandable expressions) to explain how or why the discovery is made. By providing this information, the users are given more data to judge whether they should trust the discovered results or not. Explanation plays an important role in many learning systems [5, 11, 24, 25]. Haynes suggests that users would not accept recommendations that emerge from reasoning that they do not understand [11] and we believe the same reasoning applies to discovery results as well. To our knowledge, however, there are only very few KDD or scientific discovery programs that have explanation capabilities. Probably the best example is Yao's work using rough set theory as well as supervised classification technologies to explain discovered association rules [33, 34]. Knorr proposed a way to generate explanation for distance-based outlier mining by categorizing them into stronger and weaker outlier groups [13].

Similar to rediscovery, explanation-based discovery focuses on verifying the methodologies instead of the results, because it tries to describe how and why the results are generated not whether the results are "correct". Both methods assume that trust in a discovered result can be formed by trust in the underlying discovery methods.

2.3 Exploiting External Resources

In many complex, real-world systems, verifying whether a candidate solution is right or wrong takes much less time than generating the solution. For example, verifying the correctness of a proof is usually much simpler than generating the proof in the first place. The idea of exploiting external resources to evaluate a discovery system exploits this characteristic.

Knowledge can be represented in various forms as well as acquired from difference sources. For example, suppose our system somehow discovered that "Y's car will develop a problem today". One might discredit such a result using some external knowledge not available to the system such as "Y is a 3-year-old that cannot drive". Or one might find it supported by observing that Y on average has to go to the garage once per week to fix various problems or that he just posted an advertisement looking for another car on the Internet. This shows that logical inference, statistical analysis or external sources such as the Web could all serve as different methods to verify a particular piece of knowledge. Similarly, to verify a discovered result, one can try to find support for it from external sources and reasoning independent of the discovery methods. The basic idea stems from the observation that for a specific discovery problem some resources are more suitable for generating the results while others are more adequate to verify them. In our example, it might have been impossible to make the discovery or prediction using Web sources, but once the discovery was made a focused search might have revealed the car-for-sale advertisement in support of the discovery.

To exploit external resources for verification, we have to make sure that those resources were not explicitly or implicitly used to generate the discovery. In Section 4 we will describe an example on how the methods described above can be applied to verify discovered results.

2.4 Minimum Description Length

Minimum Description Length [29], minimal encoding length [16] and the principle of parsimony (Occam's razor) [30] all try to formalize the notion that knowledge should be represented in a concise way. Information theory researchers have studied these measures extensively [3]. As pointed out by Milosavljevic, the relativity of MDL is addressed by the first central theorem of algorithmic information theory, which states that there exists a language that gives encodings of the data that are as concise as the encodings in any other language [20]. He also shows that the shortest program can give the best predictions about unseen data in molecular evolution patterns [19]. This concept has not only been used for coding and communication problems, but also been applied widely in machine learning research for model selections. To verifying discovery results, we propose to exploit MDL in two different ways:

First, if a program is designed to discover rules or patterns from existing data or observations, then the MDL principle tells us that the fewer patterns needed to describe the data the better. Said differently, the pattern that has higher coverage implies higher validity [23, 28]. In mining of association rules such as X→Y, the concepts of *support* (the percentage of transactions that contains both X and Y items) and *confidence* (given the transactions which contain X, the probability that it also contains Y) [1] are used to quantify the coverage of rules. *Sufficiency* (the probability that the evidence occurs given the hypothesis is true divided by the probability of the evidence given the hypothesis is false) and *necessity* (the probability that the evidence does not occur given the hypothesis is true divided by the probability that the evidence does not occur given the hypothesis is false) can be used to measure the confidence of a classification rule [12] as well.

Secondly, the MDL principle also tells us that a pattern discovered by a program will be more convincing if it can be described in a concise way. This claim is similar to applying Occam's razor for learning because Occam's razor prefers simple models over complicated ones. So, not only the fewer patterns the better, but also the simpler the patterns the better.

History shows that scientific discovery can satisfy this MDL criterion. For example Maxwell's equations play a more important role in physics than Faradays' law and Ampere's law, because they are more general and can explain the entire phenomenon that is explained by the other two. Also the equations are very concise:

$$\nabla \cdot \vec{B} = 0$$

$$\nabla \times \vec{E} + \partial \vec{B} / \partial t = 0$$

$$\nabla \cdot \vec{D} = 0$$

$$\nabla \times \vec{H} - \partial \vec{D} / \partial t = 0$$

However, there are several issues we have to be aware of when applying MDL to verify discovery results. First, in many discovery systems the usage described above has been implemented (maybe implicitly) as a heuristic to generate the results. In this case it is not fair to also apply it for evaluation. Moreover, the MDL criterion is usually not a suitable measurement for an instance discovery system (e.g. a system that tries to identify a potential suspect in a police database or a network intrusion event from network traffic data). On the contrary, an instance discovery system might prefer rich, "informative" instances which will require longer description length.

The last concern with applying MDL as an evaluation strategy is that the description length is highly related to the language used for encoding [20]. That is, an apparently complex result with larger encoding length might not imply that the discovery strategy is poor, rather that the encoding for the knowledge or data might not be suitable.

2.5 Measuring Unexpectedness

Intuitively, unexpected findings have a higher chance to catch one's eye, but it is risky to utilize unexpectedness alone as an indicator for the validation of a discovery. Nevertheless, unexpectedness could be a sign for a discovery that a researcher might find interesting. Various methods have been proposed to measure unexpectedness either subjectively or objectively. Subjective measures in general require the representation of a user's beliefs or background knowledge. The surprising discoveries are then those that bring completely new knowledge about a domain or that contradict current belief [21, 26]. Objective measures, in a nutshell, regard discovered results to be surprising if they are different from other candidates based on certain distance metrics [6, 10]. Freitas points out a variety of alternative unexpectedness indicators that are worth noticing such as "small disjuncts" (rules whose coverage is small), rules whose antecedent contains attributes with low information-gain, and the occurrence of Simpson's paradox (that is, an association between a pair of variables can consistently be inverted in each subpopulation of a population when the population is partitioned) [9].

The major concern for this strategy is to distinguish between unexpected results and noise. In noisy domains, many unexpected findings might be due to noise. To make things worse, a collection of small exceptions could indicate a meaningful phenomenon [4] although each individual one looks irrelevant. The other issue is that most of the methods and systems we mentioned in this section are domain dependent. The subjective measures could carry bias and the distance metrics in objective measures might be hard to define in some domains. Finally we would like to emphasize that these measures, which quantify the discovered results based on certain criteria of unexpectedness, can only strengthen our beliefs about the interestingness but not necessary the validity of the discovery.

3. RECALL AND PRECISION

In this section we discuss why traditional recall and precision cannot and should not play a critical role when trying to verify a discovery system. Recall and precision are widely accepted ways to evaluate the performance of machine learning systems, for example, to measure how well a learned classifier classifies test instances. When applied to the discovery domain, recall measures how many things that are supposed to be discovered have been discovered. Thus, it is measurable only when one knows exactly what there was to be found. In this sense, asking an unsupervised discovery system to report its recall is similar to asking Isaac Newton what percentage of physical laws relating to gravity, etc. he has discovered so far. Trying to measure discovery recall unavoidably runs into a paradox: to measure recall one has to know what there is to be found, but if one already knows that then there is no need to build a discovery system at all. For unsupervised systems, the recall measurement is therefore only suitable when the "rediscovery strategy" is applied for verification.

Precision, on the other hand, focuses on measuring how accurate the discovered results are (i.e. are the discovered results among those that are supposed to found). Though there is still no systematic way for us to compute the precision for an unsupervised discovery system, the concept of precision is not as inapplicable as recall in terms of verification. It tells us that the discovered results have to be at least plausible. Explanation-based discovery, minimum description length, the methodology of applying external resources as well as the unexpectedness measures all try to assure some level of discovery precision, that is, was something found that was worth finding.

It is usually unacceptable for a learning system to produce results of low recall and precision. However, in this section we would like to point out that for a discovery system, having low recall and precision does not necessary imply the system is valueless. Let us assume for the moment that there is somehow a way for us to know the recall and precision of a discovery system. Given that, what would the value of a discovery system be? To answer this question, we introduce a "utility function of discovery" in terms of precision (P), recall (R), total number of outputs produces by the discovery system (D), the total cases to be discovered (N), the expected utility value of a true discovery (v_+), the cost of a false positive (wrong discovery) (c_{f+}), and cost of a false negative (missed discovery) (c_{f-}). The utility function of discovery will be equal to the total utility gain minus the penalty of wrong discovery minus the penalty of missed discovery:

$$U = P \bullet D \bullet v_+ - (1-P) \bullet D \bullet c_{f+} - (1-R) \bullet N \bullet c_{f-}$$

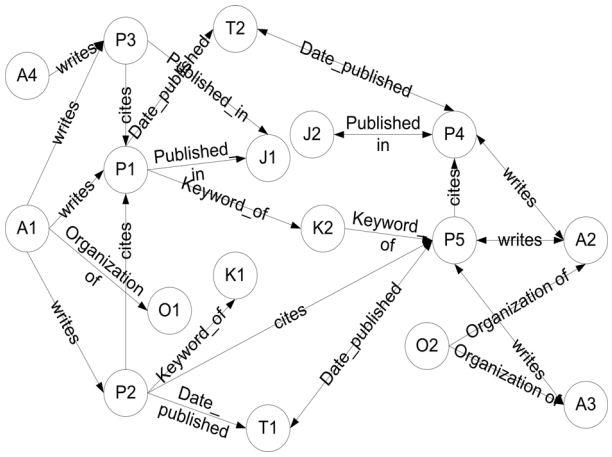
With this utility function of discovery, we can then answer the question what is the "value" of a discovery system. We claim that it is simply the difference of the utility value with and without the system or $U - U_0$. U_0 stands for the penalty of not being able to discover anything, which will be equal to the system with zero precision/recall.

$$U - U_0 = P \bullet D \bullet v_+ - (1-P) \bullet D \bullet c_{f+} - (1-R) \bullet N \bullet c_{f-} - (-N \bullet c_{f-})$$
$$= P \bullet D \bullet v_+ - (1-P) \bullet D \bullet c_{f+} + R \bullet N \bullet c_{f-}$$

From this equation we can see that low precision can still produce high utility if $v_+ >> c_{f+}$. Similarly, low recall can produce large amount of utility given c_{f-} is very large.

Let us illustrate this with an example: Assuming there is a discovery system aiming at discovering the threat events (e.g. bomb or hijacking) from some database. Saying that it predicts the location and time for five events (D=5) but only one of them is correct (P= 20%). Also assuming that there are in fact a total of 10 true threats all over the area (N=10, R=10%) and each of them causes on average 1 million dollars in damage (c_{f-}=1 million). Discovering a true event earns no financial profit (v_+=0) and each wrong finding will cost 10000 dollars (c_{f+}=10000) in wasted resources. According to the equation the value of this discovery system is 0.2*5*0 - (1-0.2)* 5 * 10000+ 0.1* 10 * 100000= 60000 dollars. The shows that in this situation it is still worth 60000 dollars to develop a discovery system achieving only 10% in recall and 20% in precision.

In general, a discovery system aims at finding something that has not yet been found, which in many cases leads to a very high v_+ (e.g. in many science discovery tasks such as gene decoding or the invention of new medicines) or c_{f-} (e.g. in the domain of homeland security, credit card fraud detection, law enforcement, and network intrusion detection), and, therefore, makes high recall and precision less important. A typical supervised learning system requires much higher precision and recall, since in general its utility is lower. The above observation is an encouraging message to researchers working on machine discovery problems because it tells us that it is still worthwhile and important to work on machine discovery problems, even though true precision and recall is hard or even impossible to measure.



4. CASE STUDY: VEFIFYING NOVEL LINK DISCOVERY TOOLS

In this section we present a case study on how some of the above strategies can be applied to verify a discovery tool suite developed by us which we call "novel link discovery (NLD) tools". We did not use the MDL strategy because they are more suitable for pattern discovery instead of instance discovery program such as our NLD tools. Note that the goal in this section is not to justify our specific approaches but to show how we can apply various indirect methods to check the validity of discovered results. We therefore focus mainly on the verification part. More detailed descriptions of our methodologies are published elsewhere[17, 18].

4.1 Novel Link Discovery (NLD) Tools

Our NLD tools are designed as unsupervised tools to discover interesting evidences and connections in multi-relational networks such as the bibliography network shown in Figure 1. In this case study, we will focus on how to verify three NLD tools that aim at discovering different type of interesting facts in the network:

1. Novel loop discovery [18]: given a multi-relational network, find the most interesting loop path (or type of loop) in it. For example, we might want to find the most interesting loop that goes through node A1 or find the most interesting type of loop in the whole network. Our basic approach to this problem (which we call rarity analysis) is that the loops that look different from most others have a higher chance to be interesting.
2. Significant node discovery [17]: given a pair of nodes, finding whether they are significantly connected to each other relative to other nodes. For example, for all pairs of nodes in Figure 1, find those that are the significantly connected to each other. The basic concept behind our solution is that two nodes are significantly connected to each other if there are many "rare paths" between them.
3. Interesting instance discovery [18]: given an arbitrary source entity in a network, find entities that are most interestingly connected to it. For example, find the most interesting organizations in Figure 1 or find organizations most interestingly connected to node A1. The solution we use for this problem is to characterize each node by its semantic profile

based on the nodes and paths surrounding it and extracting those nodes with abnormal profiles.

In terms of verification, those tools face exactly the problems outlined above. In other words, if there were gold-standard measures for interestingness or novelty, we could simply implement them as our discovery tools. Since there are no such measures, there is no perfect way to verify our system, thus, we need to exploit indirect methods for verification.

4.2 Verifying Significant Node Discovery

We applied the *rediscovery* method to verify our significant node discovery tool. To perform rediscovery, we need a dataset for which we already know the solutions to allow us to test whether our program can rediscover them. The data we used came from a suite of simulated data sets developed as part of DARPA's Evidence Extraction and Link Discovery program. It simulates a Russian Mafiya domain with a large number of entities involved in contract murders, gang wars and industry takeovers. For each dataset we are given an answer key, which describes higher-level information of interest that is not explicitly mentioned in the data but needs to be inferred from lower-level, incomplete and noisy simulated evidence. Using these answer keys we can test our program by checking if the significantly connected nodes it discovered are truly the ones that the simulator or the developers of the simulator deemed to be interesting.

4.2.1 The Russian Contract Kill Dataset

The Russian Contract Kill (or RCK) data sets were developed by Information Extraction & Transport, Inc. to serve as a challenge problem domain for different link discovery approaches. The data is generated by a simulator and describes activities of fictitious Russian Mafiya groups and the people and organizations they operate or come in contact with. The simulator has a model of high-level threat events such as contract murders, gang wars and industry takeovers and their decomposition into lower-level events (or tasks) such as observations, payments, wire transfers, information exchange, killings, etc. The hierarchy of event types used by the simulator is shown in Figure 2. The two highest-level (Lv5) threats are GangWar and IndustryTakeOver. Gang wars occur between mafiya groups and involve multiple contract murders. Industry takeovers are attempts by one mafiya group to take over an industry controlled by another, which also involves multiple contract murders. The simulator is plan-based and starts with an initial world state and some goals to generate a certain number of high-level threat events. It then hierarchically decomposes them into lower level subevents (or subtasks) until they bottom out into evidence producing actions. Most tasks can be achieved in multiple ways with various random choices along the way. Omission, corruption and noise can be used to obfuscate the generated evidence and correspondingly there are parameters one can choose to adjust the level of each.

The output of the simulator describes a set of typed entities such as mafiya groups, people, victims, hit men, banks, etc. and relationships between them, as well as evidence of events that occurred such as money transactions, meetings, killings, etc. We can build a semantic network similar to figure 1 base on the given information. The link discovery goal is to discover what high-level threat events occurred by looking at the lower-level evidence.

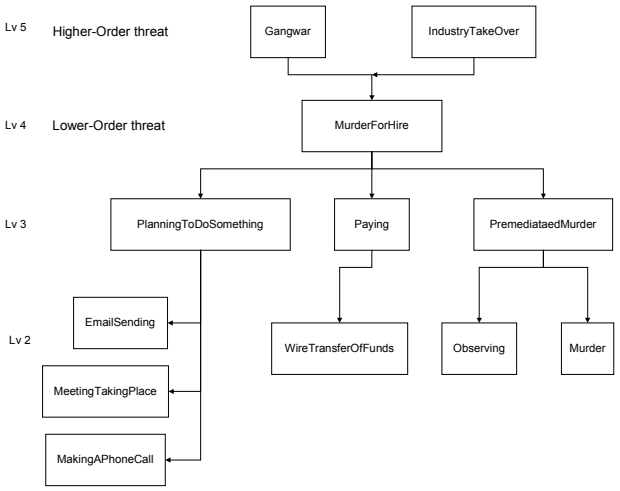


Figure 2: Russian Contract Kill data event type hierarchy

4.2.2 Experimental results

The RCK datasets where not designed to evaluate significant node discovery programs and our program was not designed to detect gang wars or contract murders in RCK datasets. Nevertheless, the information provided in answer keys can be used to evaluate our program. Answer keys give full descriptions of high-level cases such as gang wars and industry takeovers along with all participants such as the involved mafiya groups. Turning this around, we can view two mafiya groups involved in a high-level threat event as interestingly connected and then test how well our significant node discovery program can detect such interesting connections automatically looking only at low-level, incomplete evidence data. Since there is at most one gang war instance in each data set, it is reasonable to expect that the mafiya groups involved in it should have some interesting connections between them.

To do this we enumerate the set of all possible mafiya group pairs in a dataset, calculate the node-based n-rarity for each pair [17], and then rank the pairs according to the computed scores. For example, for one of the datasets there are 42 mafiya groups, hence, there are 42*41/2=861 candidates pairs to evaluate. Ideally, the mafiya group pair involved in the gang war should be among the top scoring pairs. Similarly, we can do the same for all pairs of mafiya groups and industries to detect pairs that are involved in an industry takeover.

We tested our program on six data sets turned up by different parameters and summarized the results in Table 1. Data set names are formatted as "*ObservabilitySizeNoise*" to indicate their simulator parameter settings. *Observability* describes how frequently the higher-level evidences are revealed, *Size* stands for the number of nodes and links in the data and *Noise* indicates how accurate the given evidences are. The results show that our program can perfectly predict the participants of GangWar and IndustryTakeOver events in data sets with high observability and no noise. In those cases the top-scoring pair was actually the one we were supposed to find. For data sets with lower observability and more noise, the relevant pair can still be found in the top 1-2% of ranked pairs.

In each dataset we have on average 42 mafiya groups and 21 industries connected to 8500 entities by 13500 evidence links; therefore, it would be very difficult for human beings to manually figure out the answers. Additionally, since our

program was not given any background knowledge about the event patterns nor the semantics of entities and links, the experiment also assures the generality of our discovery tool (e.g., the program does not even know what "mafiya" or "murder" means). This satisfies one of the important conditions discussed in section 3.1, namely, that rediscovery as a verification strategy is most convincing if very little domain-dependent background knowledge is used.

Table 1: Results for different data sets: Top 1 indicates the pair with highest rarity is the one we were supposed to find. Top x% indicates the pair we were supposed to find has the top x% score among all the possible pairs. There is no gang war in medium-size files.

	GangWar	IndustryTakeOver
veryHighLg0	Top 1	Top 1
veryHighMed0	n/a	Top 1
averageLg1	top1	top 0.5%
averageMed1	n/a	top 1.36%
veryLowLg2	top 2.50%	top1.6%
veryLowMed1	n/a	Top 0.4%

4.3 Verifying novel loop discovery and interesting instance discovery

Rediscovery is generally only applicable if the data is synthetic, historic with previously discovered results or small to allow manual discovery. Large real-world datasets that have known answers usable to evaluate a discovery program via rediscovery are, unfortunately, difficult to find. This section shows how we can apply some of the other strategies described in Section 3 to verify unsupervised discovery tools.

4.3.1 The high-energy physics bibliography dataset

The "High Energy Physics - Theory" (HEP-Th) bibliography dataset was provided as a test dataset for the 2003 KDD Cup. We translated the data into a semantic net similar to the one shown in Figure 1. We extracted six different types of nodes (entities) and six types of links (relations) from the dataset to generate the semantic network. Nodes represent paper IDs (29014), author names (12755), journal names (267), organization names (963), keywords (40) and the publication time encoded as year/season pairs (60). Numbers in parentheses indicate the number of different entities for each type in the dataset. We defined the following types of relationships to connect various types of nodes:

writes(a, p) : connects author a to one of his/her papers p.
date_published(p, d) : connects paper p to its publication date d.
organization_of(a, o) : connects author a to an organization o they belong to.
published_in(p, j) : connects paper p to journal j it appears in.
cites(p, r): connects paper p to a paper r it cites.
keyword_of(p,k) : connects paper p to keyword k in its title.

These links are viewed to be directional with an implicit inverse link. Thus, there are a total of 12 different relations. The network generated is similar to the one in Figure 1, only that there are 43095 different nodes and 477423 links overall.

4.3.2 Verifying interesting instance discovery

The goal of interesting instance discovery (IID) is to discover a set of interesting entities in the network. Our program does this by first generating the semantic profiles of the nodes. A semantic profile contains various path types (a path type can be seen as a specific event a node involved) as features and the node's contribution to each path type (the contribution can be regarded as a measurement of how deeply a node is involved in this event) as feature values. Then we extract the nodes of abnormal (different from others) profiles as interesting ones.

In our experiments we use two different ways to evaluate the discovered results: (1) Examine the original network to learn the reason why instances are chosen as abnormal ones. Since our program does not have any knowledge about the semantics of the nodes, manually inspecting which path types contributed the most together with our knowledge of what these path types mean is a good way of evaluating whether the program has indeed found something interesting. This verification method reflects the idea of *explanation-based discovery,* since we are examining the results by explaining how and why our program chooses them. In the current system we still need to inspect and explain results manually, but ongoing work aims at generating such explanations automatically. (2) Use the Web as an *external source* to find supporting evidence. Since the nodes represent real-world entities such as people, we can "verify" the computed results by investigating whether they reflected a real-world, semantically interesting profile or connection visible through the World-Wide Web. This method therefore uses the idea of using external resources to verify discovered results. It is fair to apply the Web information for verification, since that information was not used by our discovery tool to generate results.

The results show that C.N. Pope, Ashoke Sen, and Edward Witten are the top three interesting people discovered by our program. After looking into the data and feature distribution, we find that the reason why C.N.Pope is chosen is twofold: First, he contributed significantly in most of the path types. However this fact itself is not enough to distinguish him from other nodes that also contribute significantly. The second reason is that he contributes 0 to the path organization_of(x,$o1$) $\wedge$ organization_of(y,$o1$) $\wedge$ organization_of (y,$o2$) $\wedge$ organization_of(x,$o2$). That is, there is no other person in the data that has ever belonged to any two organizations he has ever worked in, which is abnormal for people who contribute significantly in most other dimensions. Ashoke Sen is chosen as abnormal because some features suggest he has very focused research directions (e.g. he contributes the most to the loop "a single paper cites multiple of his papers") while some suggests he has a broad research directions (e.g. he contributes relatively low to the loop "his papers are published in the same journal") which is not common at all in this data. The reason Edward Witten is chosen, in short, is because he did not contribute much for most events (e.g. he does not publish or co-author frequently with others in this data set), but also that a relatively large amount of people in this data cite more than one of his publications. After searching on the Web we found that Edward Witten is a famous mathematical physicist who has won the Fields Medal, the highest honor a mathematician can receive. This fact strengthens the validity of our discovery, since even though his research is not fully focused on high-energy physics, some of his contributions to the fundamental mathematics must be valuable to this community and thus attract many citations.

4.3.3 Verifying Novel Loop Discovery

The goal of novel loop discovery is to find interesting loops in multi-relational datasets. The program models interestingness via a rarity measure. It tries to determine which types of loops are rare compared with the whole dataset [18]. We verified the tool based on the high-energy physics data as well. The rarest, least frequent types of loops we found are listed in Table 2. The most rare loops represent papers citing themselves directly, which only occurs 28 times in the whole dataset. We do not have a real world explanation for this and can only attribute it to errors in the dataset. The second, third and fourth paths are citation loops of different length. The rationale behind this finding is that for a paper to cite another, the cited paper needs to be published earlier. In this sense a citation loop such as "P1 cites P2 cites P3 cites P1" is really a contradiction in time and should not occur at all. One explanation for such "contradictions" is that sometimes an author (or close colleague) might cite one of his/her own submitted but not yet published papers P2 (which has already cited P1) in a paper P1. The other explanation is that a journal might have a very long revision period and during that period other people can access the previous version. For both explanations we have found supporting instances (e.g. "0110099 cites 0110200 cites 0110186 cites 0110099" for the first case and "9912210 cites 9906151 cites 9509140 cites 9912210" for the second). However, there are still some other unexplainable citation loops (e.g. "9912288 cites 0004011 cites 9911183 cites 9912288") that might occur due to the difference between the true publication date and the SLAC-date. The fifth path shows a similar concept where it is rare for a paper to cite another paper that was published during the same time period. This type of loop could also be an indicator for authors that work closely with each other. Finally, the last path shows that people seldom publish multiple papers at the same time.

This case study exploits another *external resource* to verify the results which is our background knowledge of the scientific publication domain (e.g. for P1 to cite P2, P1 has to publish later than P2) as well as some simple inference capability (e.g. if A occurs earlier than B and B occurs earlier than C, then C can't occur earlier than A). The results of the program are verified to be "interesting" (at least to some extent), because they are against the common knowledge or expectations we have. Besides that, the discovery of citations loops is somehow unexpected, since they are rare despite the fact that citation paths (A cites B cites C…) themselves are very common.

Table 2 The rare loops

Top 6 rarest loops
1. PaperX cites PaperX
2. PaperX cites Paper1→Paper1 cites PaperX
3. PaperX cites Paper1→Paper1 cites Paper2 → Paper2 cites PaperX
4. PaperX cites Paper1→Paper1 cites Paper2 → Paper2 cites Paper3 →Paper3 cites PaperX
5. PaperX cites (or cited by) Paper1 → Paper 1 published at Time1→ At Time1, PaperX also published.
6. PaperX is written by Person1 → Person 1 has another Paper1→ Paper1 published at the same time period as PaperX

5. CONCLUSIONS

There is much less research being done in unsupervised machine discovery compared to, for example, machine learning. We believe this to be the case not only because of the difficulty of the problem, but also because of the difficulty to verify the results. However, this fact does not justify that unsupervised machine discovery should deserve less attention than other problems. In fact, as we motivated in Section 3, the utility value for discovery can be much higher than for learning which is a strong reason to engage in machine discovery research. Take the mathematical conjecture discovery program GRAFFITI [7] for example: its author Dr. Fajtlozicz published a fair amount of conjectures discovered by his program in a mathematical journal, but *without evaluating them or proving their correctness*. Despite that (or probably because of that), this then led to tens of publications (including Ph.D. theses) in mathematics centered on proving or disproving those conjectures. Had he waited until he was able to verify the conjectures before publishing them, GRAFFITI would probably still be an unknown program.

In applied sciences such as computer or information science, it is hard for research to receive credit without verifying its validity. The success story of GRAFFITI tells us, however, that the value of a discovery program is sometimes large enough to outweigh the problem of not having straightforward and convincing verification methods. To further help with this situation, this paper addresses various indirect verification methods for discovery systems and their application to a suite of link discovery tools. We hope that this overview will help and encourage more researchers to work on machine discovery problems.

6. REFERENCES

1. R. Agrawal, T. Imielinski, and A. Swami. *Mining association rules between sets of items in large databases*. in *ACM SIGMOD*. 1993. Washington D.C.
2. P.S. Bradley. *Data Mining as an Automated Service*. in *PAKDD 2003*. 2003: Springer-Verlag Heidelberg.
3. T. Cover and J. Thomas, *Elements of Information Theory*. 1991: Wiley.
4. A.P. Danyluk and F.J. Provost. *Small Disjuncts in Action: Learning to Diagnose Errors in the Local Loop of the Telephone Network*. in *In Proceedings of the Tenth International Conference on Machine Learning*. 1993.
5. J.S. Dhaliwal and I. Benbasat, *The use and effects of knowledge-based system explanations: theoretical foundations and a framework for empirical valuation.* Information Systems Research,, 1996. **7**: p. 342-362.
6. G. Dong and J. Li. *Interestingness of Discovered Association Rules in terms of Neighborhood-Based Unexpectedness*. in *Proceedings of Pacific Asia Conference on Knowledge Discovery in Databases*. 1998.
7. S. Fajtlowicz, *On conjectures of Graffiti.* Discrete Mathematics, 1988. **72**: p. 113-118.
8. U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, *The KDD Process for Extracting Useful Knowledge from Volumes of Data.* Communications of the ACM, 1996. **39(11)**: p. 27-34.

9. A.A. Freitas. *On objective measures of rule surprisingness*. in *Principles of Data Mining & Knowledge Discovery (PDDD)*. 1998.
10. P. Gago and C. Bento. *A Metric for Selection of the Most Promising Rules*. in *European Conference on Principles of Data Mining and Knowledge Discovery*. 1998.
11. S.R. Haynes, *Explanation in Information Systems: A Design Rationale Approach*, in *The London School of Economics*. 2001, University of London.
12. M. Kamber and R. Shinghal. *Evaluating the interestingness of characteristic rules*. in *Second Int'l Conference on Knowledge Discovery and Data Mining*. 1996.
13. E. Knorr and R. Ng. *Finding Intensional Knowledge of Distance Based Outliers*. in *VLDB*. 1999: Endiburgh Scotland.
14. P. Langley, et al., *Scientific discovery: computational explorations of the creative process.* Cambridge, MA: The MIT Press, 1987.
15. D. Lenat, *The Nature of Heuristics.* Artificial Intelligence, 1982. **19**: p. 189-249.
16. L. Li and P. Vitanyi, *An Introduction to Lolmogorov Complexity and its applications*. 1993: Springer Verlag.
17. S. Lin and H. Chalupsky. *Unsupervised Link Discovery in Multi-relational Data via Rarity Analysis*. in *Proceedings of IEEE International Conference on Data Mining (ICDM)*. 2003. Florida.
18. S. Lin and H. Chalupsky, *Using Unsupervised Link Discovery Methods to Find Interesting Facts and Connections in a Bibliography Dataset.* KDD Explorations, 2003. **5**(2): p. 173-179.
19. A. Milosavljevic, *Discovery by minimal length encoding: A case study in molecular evolution.* Machine Learning Journal, 1993. **12**: p. 69-87.
20. A. Milosavljevic, *Discovery Process as a Search for Concise Encoding of Observed Data*, in *Machine Discovery*, Jan Zytkow, Editor. 1997, Kluwer Academic Publishers.
21. B. Padmanabhan and A. Tuzhilin, *Unexpectedness as a measure of interestingness in knowledge discovery.* Decision Support Systems, 1999. **27**: p. 303--318.
22. V. Pericliev, *A linguistic discovery system that verbalizes its discoveries.* 19th International Conference on Computational Linguistics, 2002: p. 1258-62.
23. G. Piatesky-Shapiro, *Discovery, analysis and presentation of strong rules.* Knowledge Discovery in Databases, 1991: p. 229-248.
24. J. Pitt, *Theory of Explanation*. Oxford University Press. 1988: Oxford.
25. R. Schank and A. Kass, *Explanations, machine learning, and creativity.* Machine Learning: An Artificial Intelligence Approach, 1990. **3**: p. 31-48.
26. A. Silberschatz and A. Tuzhilin. *On subjective measures of interestingness in knowledge discovery*. in *First International Conference on Knowledge Discovery and Data Mining*. 1995.
27. H. Simon, *Machine Discovery.* Foundations of Science, 1995. **2**: p. 171-200.
28. P. Smyth and R.M. Goodman, *Rule Induction Using Information Theory.* In Knowledge Discovery in Databases, 1991: p. 159-176.
29. E. Sober, *Reconstructing the Past: Parsimony, Evolution, and Inference*. 1988: MIT Press.
30. R. Solomonoff, *A formal theory of inductive inference, Part I.* Information and control, 1964. **7**: p. 1-22.
31. D.R. Swanson, *Fish Oil, Raynaud's syndrome and undiscovered public knowledge.* Perspectives in Biology and Medicine, 1986.
32. R.E. Valdes-Perez and V. Pericliev, *Computer Enumeration of Significant Implicational Universals of Kinship Terminology.* Cross-Cultural Research: The Journal of Comparative Social Science, 1999. **33**(2): p. 162-174.
33. Y. Yao, Y. Zhao, and R.B. Maguire. *Explanation-Oriented Association Mining Using a Combination of Unsupervised and Supervised Learning Algorithms*. in *Canadian Conference on AI*. 2003.
34. Y.Y. Yao, Y. Zhao, and R.B. Maguire. *Explanation-oriented association mining using a combination of unsupervised and supervised learning algorithms*. in *Conference of the Canadian Society for Computational Studies of Intelligence*. 2003.