Research Description

The huge wealth of data accessible today is a source of tremendous potential benefits and also potential pitfalls. On the positive side, the accessibility of new integrated information sources has the potential to make our lives easier and more productive, if we can infer the information relationships of interest to us from the data. On the negative side, this information glut has the potential of overwhelming us with incorrect or noisy information, and may permit unauthorized users to infer private information. In order to make effective use of this data, we need to understand both how to mine it effectively, and how to ensure sensitive information cannot be inferred.

Much of the data easily available to us is best represented as a graph or network. Unfortunately, this does not fit neatly into the common format assumed by statistical machine learning algorithms; most machine learning algorithms assume that the data is generated by an independent, identically distributed (i.i.d) process, and can be represented by a fixed-length vector of attribute values. Within the machine learning community, there has been a growing interest in learning structured models from input data that is also structured. Statistical relational learning (SRL) is one of the research sub-communities studying methods for combining statistical modeling with rich declarative knowledge representations capable of learning from graph and network data.

In this research, I define the abstract problem of graph identification. Graph identification is the process of taking as input an observed "data network" and outputting a transformed "information network." The data network may be noisy, but will provide the evidence that is used to construct the information network containing the entities and relationships of interest.

There are at least three major components that go into graph identification:

1. **entity resolution**: determining when two nodes in the data graph refer to the same underlying entity in the information graph, and therefore must be merged in the information graph.

2. **link prediction**: inferring edges in the information graph, based on the information contained in the data graph.

3. **collective classification**: labeling the nodes in the output graph; the labeling is collective in the sense that the label of one node depends on the labels of neighboring nodes, so the labels must be determined jointly.

These correspond to the graph operations of partitioning nodes, adding edges and labeling nodes. These are not the only operations; group detection (inferring a hidden node) and community identification (finding cliques) are also important components. However, since these topics have received more research attention, we will not focus as intensely on them.

An example of graph identification is the inference of a personal social network from observed email communication data. The input data graph is the email communication network: nodes correspond to email addresses and links represent email exchanges between email addresses. In such a graph, entity resolution is the problem of grouping together all of the email addresses that correspond to the same underlying individual. An example of link prediction is the problem of identifying the relationships among the correspondents; we may be able to infer which relationships are manager-subordinate relationships, and which relationships are personal relationships. Based on the other relationships that have been inferred, we may even be able to predict relationships when there is no observed communication between individuals. Finally, an example of collective classification is inferring an employee's department, which is likely to be the same as their co-workers' department; furthermore, employees in one department may have common communication patterns with employees from other departments. (For example, marketing communicates frequently to sales and development; development doesn't interact much with sales.)
Graph identification problems are common in many different domains. Any domain consisting of noisy transaction or interaction data, among individuals that do not have unique identifiers in which one is trying to make inferences about structured outputs is a candidate. There are many examples in biological domains. The identification of gene regulatory networks from noisy protein-protein interactions, the discovery of disease transmission networks, and the reconstruction of lineage trees all can be seen as forms of graph identification. There are computer vision problems, such as scene understanding, that can be framed as graph identification. Finally, many information extraction problems can be thought of as graph identification.

The proposed work builds on my expertise in each of the subtasks, and is novel in that there have been few attempts to combine all these tasks into a single framework. My group has done extensive work on entity resolution [7, 8, 4, 18, 5, 3], link prediction [25, 24, 26, 55, 81], and collective classification [74, 23, 46, 47, 48, 23, 73, 72]. In the proposed effort, we will develop a generic tool for graph identification, which will support interleaving entity resolution, link prediction, and collective classification. A key feature of the framework will be a declarative input format that will allow users to describe their input data schema and the desired output graph characteristics. The framework will also support flexible plug-in components for each of the major processes. Where applicable, we will compare with existing approaches, such as Markov logic networks [66]. This framework will allow us to develop new strategies for collective inference where the inference of nodes, edges, and labels is done in concert. Such strategies will help us understand the trade-offs between exact and approximate inference in these problems and have the potential to help us understand how the underlying structure of the information network impacts its learnability. Such a comprehensive framework is particularly important in the study of privacy where the goal is to ensure that specified sensitive information cannot be inferred from a data network as a whole.

**Intellectual Merit:** The proposed work will develop new algorithms which integrate entity resolution, link prediction, and collective classification. In addition, an understanding of the theory of graph identification will help us understand the conditions under which a network cannot be re-identified, which has important privacy and security implications.

**Broader Impact:** The output of the graph identification system—the inferred information graph—has a number of uses. Knowledge discovery is an obvious one, but there are several others that may also be of benefit. One potential application is personal information management. For example, the use of an inferred social network for enhancing users' productivity by helping to manage information overload. Graph identification could be part of an underlying system that develops a model of a user's context, which might consist of their social environment, their information-seeking needs, and their current tasks. This model could be used to help with tasks such as organizing and prioritizing email, retrieving relevant messages, and performing auto-completion. There are a number of law enforcement and national security applications such as inferring a financial laundering network, uncovering the structure of terrorist networks, and assessing the impact of potential threats. A third long-range impact is privacy management. The dual of graph identification is understanding how to ensure that identification of particular individuals or relationships is not possible. By analyzing the process of graph identification, we can identify situations in which we can guarantee privacy of, for example, medical records and other sensitive information.

I have a strong track record in disseminating the results of my research. In the past five years, I have given more than 50 invited lectures, in eleven different countries, including an invited talk that I will be giving at AAAI this month. I have a commitment to mentoring and serving as a role model for underrepresented and non-traditional students. I will continue my work on interdisciplinary domains (including biological, physics and social science applications) to ensure
Figure 1: A series of resolutions on a portion of the InfoVis data set. (a) The original data graph, showing authors and co-authorship information for the InfoVis data set. It is apparent that there are a number of duplicates. (b) The information graph, after the following resolutions have been performed: "Hua Su" and "Hua Su," "Lisa Tweedie" and "L. Tweedie," "Huw Dawkes" and "H. Dawkes," "Bob Spencer" and "B. Spence," and "Robert Spence" and "Bob Spence." The final graph is much simpler (and more informative). http://www.cs.ucc.edu/%7Eer/er-animation.html shows an animation of the process.

the broad applicability and widespread adoption of the results of this work.

1 Prior Work

The next three sections review research in entity resolution, link prediction and collective classification, highlighting the results of my group.

1.1 Entity Resolution

Entity resolution is the problem of determining when two nodes in a data graph are referring to the same underlying real-world entity. Figure 1(a) shows a data graph for a small portion of a co-author network that was used for the 2004 InfoVis Visualization Contest [20]. The data graph describes publications from a ten-year history of the InfoVis Conference. In this graph, the nodes represent authors, and the links represent co-authorship relationships. Even though the data graph had been hand-cleaned by several experts from the community, it is easy to spot many duplicates. In fact, after going through the steps of merging all of the duplicates, the final graph, shown in Figure 1(b), is much simpler, more accurate, and more informative, in that it shows a small research clique, which is not apparent in the original information graph.

Entity resolution is a common problem that comes in different guises (and is given different names) in many computer science domains. Examples include computer vision, where we need to figure out when regions in two different images refer to the same underlying object (the correspondence problem); natural language processing when we would like to determine which noun phrases refer to the same underlying entity (co-reference resolution); and databases, where, when merging two databases or cleaning a database, we would like to determine when two records are referring to the same real-world object (deduplication and data integration). Deduplication [30, 52] is important for both accurate analysis, for example determining the number of customers, and for cost-effectiveness, for example removing duplicates from mailing lists. In information integration, determining approximate joins [11] is important for consolidating information from multiple sources; most often there will not be a common unique key that can be used to join tables in distributed databases, and we must infer when two records from different databases, possibly with
different structures, refer to the same entity. In many of these examples, co-occurrence information in the input can be naturally represented as a graph.

Traditional approaches to entity resolution and deduplication use a variety of attribute similarity measures, often based on approximate string matching criteria. These work well for correcting typographical errors and other types of noisy reference attributes. More sophisticated approaches make use of domain specific attribute similarity measures and often learn such mapping functions from resolved data. However, it is still difficult to decide when identical references are in fact distinct. For example, two people with name ‘J. Doe’ and living at the same address and of the same age may be brothers and not the same person.

More recent approaches take relational (i.e., graph structure) similarity into account [1, 3, 35, 19]. One approach simply looks at the attributes of neighboring references, and incorporates them into the attribute similarity score. For example, if we are comparing two nodes for ‘Jon Doe’ and ‘Jonathan Doe’, we should be more likely to match them if they are married to ‘Jean Doe’ and ‘Jeanette Doe’. The problem becomes even more interesting when we assume that the entity for a reference depends not on the attribute similarities of related references but instead on the entities to which they correspond. Then the references cannot be assigned to entities independently any more — the entities for related references depend on one another. In our example, we would not consider ‘Jon Doe’ and ‘Jonathan Doe’ to be the same person simply because their wives’ names are similar, since different people may have wives with similar names. But if we can determine that they link to the same node, i.e., they are married to the same person, this would provide significant evidence that these references refer to the same entity. Because the resolutions are no longer independent, the problem becomes one of collective entity resolution.

We have developed a relational clustering algorithm [3, 5, 7] for collective entity resolution which iteratively discovers entities by clustering references, taking into account the clusters of co-occurring references. We have applied our algorithms to a number of real world problems and found significant improvement over traditional approaches. We participated in an entity resolution challenge developed by the KDD program, and our algorithm was one of the top-performing unsupervised methods. In addition to the relational clustering approach to entity resolution, we have developed a second entity resolution approach based on non-parametric hierarchical Bayesian methods to build a probabilistic generative model for collective resolution that uncovers hidden group structures among the entities and uses the latent groups as evidence. The paper describing this work won the best paper award at the SIAM Data Mining conference in 2006 [4]. Beyond developing practical algorithms for entity resolution, we developed a theoretical analysis to determine when exploiting relational information will reduce referential ambiguity[7]. This led nicely into work on query-time entity resolution [6, 8], in which, rather than viewing entity resolution as a one-time, offline process, we consider the case where it must be done online at runtime. We developed an adaptive algorithm that uses collective resolution for answering queries by recursively exploring and resolving related references. This enables resolution at query-time, while preserving the performance benefits of collective resolution.

1.2 Collective Classification

Classification is the problem of assigning, or labeling, nodes based on the observed attributes of the instances. An example is determining the category or topic of a paper based upon the words that appear in it. In traditional statistical machine learning approaches, the instances are often assumed to be generated by an independent, identically distributed (i.i.d) process. Collective classification is the task of inferring the class labels of a network of objects simultaneously. The underlying assumption in collective classification is that the relationships between objects carry important
information for classifying the objects. In many cases, there is a simple auto-correlation between
the objects [34], which means that linked objects are likely to have the same labels. But other,
more complex correlations can be modeled and exploited, such as the fact that professor web pages
are more likely to link to student web pages and course web pages, and are less likely to link to
other professor web pages [13].

There has been a large amount of research in collective classification [10, 58, 60, 76, 46, 50, 51].
Collective classification algorithms build local predictive models for the class of a node based on
node attributes and the attributes of a node’s neighbors and use some inference procedure to
collectively classify the unlabeled objects. Techniques can differ in both the local models and the
inference methods used. Chakrabarti et al. [16] use naive Bayes models for the local attributes of
the object and the class labels of the related objects. They then use mean field relaxation labeling
for the inference [67]. Neville and Jensen [58] also use a naive Bayes model for the attributes,
but they use iterative classification (ICA) for inference. In later work, they investigate the use of
relational dependency networks and the inference algorithm is based on Gibbs sampling [59]. Lu
and Getoor [46] use logistic regression as a model and ICA for inference but they explore a set
of aggregates that can be used for the class labels of the related objects. Getoor et al. [26] use
probabilistic relational models as a (full joint) model and then use loopy belief propagation for
the inference [54]. Taskar et al. [76] use relational Markov networks as a (full joint) model and also use
loopy belief propagation. Mcskassy and Provost [49] propose a baseline where they use only the
class labels of objects for classification; they infer the class label of an object by taking a weighted
average of the potentially inferred class labels of the related objects.

My work with Ben Taskar, Eran Segal and Duphne Koller [26] was one of the early approaches
that attempted to combine attributes and links for the collective classification of objects in the
non-i.i.d setting. In later work, with my student Qing Lu, we developed a framework for link-based
classification in irregular graphs, explored the effect of different representations for the neighboring
nodes, explored how, in the relational context, unlabeled training data can help improve performance,
and investigated the effect of different inference strategies [46, 47, 48, 23, 74].

Most classification algorithms assume that misclassification costs are equally costly, but in many
problems, such as disease diagnosis, there are significant differences in cost between false positive
and false negative errors. Cost-sensitive learning refers to learning algorithms which take into
account the difference in misclassification costs; often, in addition to differences in costs, there
are differences in class distributions that must be taken into account as well. Together with my
student Prithviraj Sen, I developed the first methods which take structured costs into account
during collective classification [72, 73]. Our work on cost-sensitive Markov network classification
captures correlations in the link structure and handles structured cost functions. We developed
two cost-sensitive structured classifiers based on maximum entropy principles and demonstrated
the utility of our cost-sensitive structured classifiers experimentally on synthetic and real-world
data.

1.3 Link Prediction

Link prediction is the problem of identifying whether a link exists between two objects. The link
problem can be formulated either as the identification of unobserved links in a static network or as
a dynamic prediction problem where the task is to predict which links will be present in the network
at a time $t + 1$ given the state of a network at time $t$. In both cases, link prediction is treated as
a binary classification problem over the set of all possible links. Link prediction is difficult since it
involves a large class skew in the number of positive and negative cases. Often, negative cases are
quadratic in the number of objects while the number of positive cases is linear [65]. Despite this
issue, there have been a number of successful applications of link prediction in a variety of domains using a number of different attributes and approaches. Liben-Nowell and Kleinberg [45] look at link prediction specifically for social networks and apply a series of predictors using only structural properties of the network. Popescul and Ungar [63] predict new links in a bibliographic dataset by generating and searching a space of relational features to learn potential predictors. Sarrukai [71] proposed a Markov Chain approach as an adaptive and generative way to do link prediction in web sequence modeling, while Kashima and Abe [37] use only topological features of the network structures and propose a probabilistic network evolution model over a biological network data set. Yu et al. [80] use stochastic relational models and apply it to a user-movie preference link prediction task. There have also been studies on adapting link prediction approaches to a related task of anomalous link detection [31, 65].

2 Proposed Work

The proposed research is to develop GID, a graph identification system that combines entity resolution, collective classification, and link prediction into a single unified framework. This will require: 1) the development of a flexible model construction language that allows users to define the set of graph-based features that should be included or considered in the models; 2) the development and analysis of collective algorithms which combine all three subtasks; and 3) development of a user interface which allows users to navigate the data graph and information graph in a coordinated fashion, and to view, understand and potentially modify the resulting information graph. Our goal is to develop an end-to-end open source system which will be easily extensible. This will give other researchers a platform which has a set of basic collective inference algorithms, together with several local classifiers for entity resolution, collective classification and link prediction. They can use this platform as a baseline, or they can build and extend it with their own methods.

There have been approaches that perform some sort of global inference for inferring structured outputs. For example, there has been work that performs global inference by combining local classifiers [68, 69, 14, 75, 17]. Much of this work has been done in the natural language processing community where the input is unstructured text, and the information extraction task involves entity extraction and semantic role labeling. In some sense, because we are restricting ourselves to inputs which are already structured, we are studying a simpler problem. Yet at the same time, the transformations that we consider (partitioning, labeling, and linking) are more complex than simply joint labeling. There are other approaches which use a global, joint probabilistic model. Examples which combine collective classification and some sort of link prediction jointly include [25, 77, 66]. One of the major differences between our approach and these other approaches is that our approach is a relatively simple mechanism for which users can plug-in almost any collective classification, link prediction or entity resolution algorithms. Moreover, due to its simplicity, it is straightforward to analyze the interaction between entity resolution, object classification and link prediction.

2.1 Model Specification Language

An important contribution of this work will be a uniform representation for describing input to SRL algorithms. This will include a data definition language that describes the different node and link types, and their attributes. We plan to use a generic format, similar in spirit to the WEKA arff input format, that is expressive enough to describe multi-modal attributed input data. We will also provide simple translators for other formats such as SQL and some of the social network software input formats.
Algorithm GID Learning\((G_{DP}, G_{IR}, M_{CC}, M_{LP}, M_{ER})\)

**Input:** Data graph \(G_D\)

**Output:** Local classifiers \(M_{CC}, M_{LP}, M_{ER}\), for object classification, link prediction and entity resolution respectively.

1. Train \(M_{CC}\) on training pair \(G_{DP}\) and \(G_{IR}\).
2. Train \(M_{LP}\) on training pair \(G_{DP}\) and \(G_{IR}\).
3. Train \(M_{ER}\) on training pair \(G_{DP}\) and \(G_{IR}\).
4. Return \(M_{CC}, M_{LP}, M_{ER}\)

Figure 2: The main GID Learning algorithm

In addition to the data definition language, we will provide a declarative feature construction language which will allow users to easily specify the aggregates that should be considered within the model. Feature construction is an important, and sometimes overlooked element of SRL models. While there has been some work in this area [62, 64, 16, 15, 46, 33], still the most common approach to constructing and comparing alternate feature representations is to write a Perl script to transform the data. We propose a general feature construction language, and the machinery such that the appropriate feature values will be computed as needed within the learning and inference algorithms. While this has much appeal to users who will no longer need to write special purpose scripts, it is also required so that we can support a tight level of integration between the algorithms in which, for example, inferences made by the link prediction system affect the aggregate features used by the collective classification system.

Our language will support the common used aggregates such as \textit{mode}, \textit{count}, \textit{min}, and \textit{max}. It will also have primitives to support basic structural attributes common within the network analysis community such as \textit{centrality} and \textit{betweenness}. For richer features, we will use representation techniques from the inductive logic programming community [53, 42, 40]. We will also investigate feature construction methods from the graph mining community [70, 12, 32, 41, 78] and graph kernels community [21, 38, 22].

2.2 Collective Inference Algorithm

Our proposed collective inference algorithm allows for flexible integration of object classification, link prediction and entity resolution. Initially, for simplicity, we plan to separate the learning of the local classifiers from the collective inference using the classifiers. We assume, again for simplicity, that we have a training graph pair \(G_{DP}\) and \(G_{IR}\). This has advantages in terms of extensibility and ease of implementation. In later stages, we will explore more fully integrated learning and inference methods. Figure 2 shows a high-level template for the learning and Figure 3 shows the inference algorithms.

The algorithms described above are very high-level, and we plan to investigate a range of representations, interleaving strategies, and stopping criteria. One question to explore is whether the inferences at each stage should be "hard" assignments (e.g., MAP assignments) or "soft" assignments. There are interesting trade-offs in terms of computational efficiency and time complexity, and at this point we have little intuition about how best to make the trade-offs. For interleaving strategies, at one extreme we can simply do one step of inference in each of the models at each call, at the other extreme we can run each collective inference algorithm to completion. It is likely that some middle ground is best, but this needs to be verified on some real-world problems. Another challenge for the algorithms is efficient passing of information; in our earlier work on collective entity resolution, we constructed the appropriate index structures to make this efficient. We will need to do the same for collective classification and link prediction. This requires keeping track of
Algorithm GID Inference \((G_D, M_{CC}, M_{LP}, M_{ER}, G_I)\)

**Input:** Data graph \(G_D\), and local classifiers \(M_{CC}, M_{LP}, M_{ER}, G_I\)

**Output:** Information graph \(G_I\)

1. repeat
2. \(G_I \leftarrow \text{perform collective entity resolution using } M_{ER} \text{ and updated } G_I\).
3. \(G_I \leftarrow \text{perform collective link prediction using } M_{LP} \text{ and updated } G_I\).
4. \(G_I \leftarrow \text{perform collective object classification using } M_{CC} \text{ and updated } G_I\).
5. until no more additional inference required
6. return \(G_I\)

Figure 3: The main GID inference algorithm. After each collective inference algorithm is applied, \(G_I\) is updated with the latest resolutions, links and labels. This will affect the inputs to the next stage.

the values that aggregates depend on; for example if a new link is inferred, that may require the attributes of some of the local classifiers for entity resolution and object classification to be recomputed. Stopping criteria are, of course, also important considerations.

Once we have basic implementations for the algorithms in place, there is a very rich research space to explore. Beyond those mentioned above, there is work on semi-supervised approaches which integrate learning and inference, there is work to be done on active feature acquisition, and, while our focus is the complete graph identification process, we expect we may learn something more about the individual processes as well; e.g., we may gain some insight into link prediction algorithms that will help us construct better stand-alone link prediction algorithms.

### 2.3 User Interface for Graph Identification

In addition to the representation and algorithmic support for graph identification, we also plan to develop a user interface that will help users interact with the collective inference system and understand the output of the system. By its nature, graph identification is a complex process, and in many cases it makes sense to have a human in the loop. Our group has developed a tool, D-Dupe [9], which provides a novel user interface for entity resolution in graphs. The tool has been very well received, both by our sponsors and the research community. We believe the utility of the GID framework will be greatly enhanced by having a user interface which allows users to inspect the data graph and information graph in parallel. This tool would allow users to override the inferences and ordering decisions made by the system, and allow users to compare two information graphs inferred under different operating assumptions.

### 3 Graph Re-Identification and Privacy

A better theoretical and algorithmic understanding of graph identification has benefits not only for knowledge discovery, it also has implications for data privacy and security. The flip-side of graph identification is understanding the conditions under which a graph cannot be re-identified. This may help us build better data anonymization models for graph data and provide more appropriate privacy and security guarantees. If the identities of the nodes are sensitive, then we should take care in releasing relational information; if the links themselves are sensitive, then we should be aware that careless leaking of node information may be enough to infer links.

Most of the work in this area makes the assumption that the data is described by a single table with attribute information for each of the entries. However, real-world datasets often exhibit more complexity. Relational data, often represented as a multi-graph, can exhibit rich dependencies between entities. The challenge of anonymizing graph data lies in understanding these dependencies.
and removing sensitive information which can be inferred by direct or indirect means.

There has been a growing interest in privacy in multi-relational settings and to date very little work has been done in this direction. The existing work looks at the identifying structural properties of the graph nodes [2, 28], or considers relations to be attributes of nodes [57]. Our work assumes that the anonymized data will be useful only if it contains both structural properties and node attributes. We study anonymization techniques to match this assumption.

4 Preliminary Results

Next, we describe some preliminary results on combining object labeling and link prediction, privacy and the label re-identification problem, and a user interface for graph identification.

4.1 Combining Collective Classification and Link Prediction

As a first step toward our GID framework, we have built a system, Collective Object Classification and Link Prediction (COL), which combines object classification and link prediction. The local models for object classification and link prediction are logistic regression models (which we have found to work better than other classifiers, such as naive Bayes), and the collective inference algorithm used is the ICA algorithm.

In order to evaluate the COL system, we built a relatively sophisticated data generator that can generate data graphs with varying amounts of noise, auto-correlation among the object labels,
link density, etc. It models the dependence of the links and labels on object attributes and groups. In addition to making the code for the system we develop available, we plan to make our synthetic data generators available.

We compared our COL algorithm to collective object classification (CC) algorithms that were either given all link information (CC w/ Links) or no link information (CC w/o Links), and to link prediction (LP) algorithms that were either given all the labels (LP w/ Labels) or none of the labels (LP w/o Labels). COL, on the other hand, was not given any label or link information in the test graph; it was given only the group attributes and the class attributes (as were the other algorithms). We also tested robustness of COL over different network types by exploring four parameters: the amount of homophily, the amount of noise in the class attribute distribution, the amount of noise in link generation, and the link density of the graphs. Overall, our hypothesis was that COL algorithm would perform better than collective object classification without any links and would do comparably well with collective object classification given all the link information. We had the same hypothesis for the link prediction component of the COL given none or all links. We use accuracy as the performance metric for collective object classification and F1 for link prediction. We chose F1 over accuracy for link prediction because of the huge class skew. All results are averaged over ten runs and are presented in Figure 4.

In most of the results, COL-CC performed significantly better than collective object classification without links (i.e., flat classification). Remember that COL was not given any links to start with. Furthermore, COL-CC performed comparably well with collective classification with all links; the differences between the two is not statistically significant, confirming our hypothesis. However, even though COL-LP performed better than link prediction without any labels, the differences were not as significant. We think that the class label information may not be as informative as we initially expected for our data.

With the first parameter, homophily, we expect that as the amount of homophily increases in the graph, collective classification with all labels will increasingly perform better than collective object classification without any links. This result is apparent from our results when we compare the first and third rows of Figure 4(a). COL-CC does better than flat classification in all cases, and does significantly better for medium and high homophily. Its performance is also comparable with collective classification with all links observed. Our hypothesis was that link prediction with all labels known would perform similarly better in a more homophilic setting because the class labels would become more informative for link prediction. However, we did not observe any difference in link prediction performance over different homophily values.

The results for the next parameter, class attribute noise, are presented in Figure 4(b). As we increased the noise, we expected a drop in the flat classification and the difference between COL-CC and flat classification to increase. We did not expect any significant changes in link prediction results as the links and the link prediction algorithm do not depend on class attribute values directly. As before, COL performed consistently better than CC and LP without links and labels, and is comparable with the case where links and labels are given.

In the next two experiments, we varied the link parameters. First, we increased the noise in link generation (Figure 4(c)). As expected, the performance for link prediction decreased. One of the effects this had on COL-CC was that, even though with low and medium link noise COL-CC was performing comparably well with CC with all links, COL-CC performance degraded when the link noise was high. The reason for this result is that link prediction is not able to perform as well anymore, and thus is of less help to COL-CC.

Finally, we varied the link density of the graphs (Figure 4(c)). For low density, the link prediction algorithms performed poorly. We found that a large part of the reason for this poor performance is that given the class skew problem of link prediction, the logistic regression algorithm did not
have enough positive examples to learn a good model of the data. Similarly, the number of links was not enough for collective classification algorithms to perform significantly better than flat classification. We found, however, that as we increased the link density, link prediction algorithms performed better, and at the same time the collective object classification algorithms performed significantly better than flat classification.

4.2 Privacy and Link Re-identification

We have begun looking at privacy in graph data, and have first considered the case where relationships between entities are to be kept private. Finding out about the existence of these sensitive relationships leads to a privacy breach. We refer to the problem of inferring sensitive relationships from anonymized graph data as link re-identification.

We consider the node data to be anonymized using a known single-table definition such as k-anonymization [70] or the more recently proposed t-closeness [44]. For the edge data, we propose five different anonymization strategies. The most conservative approach is to remove the relationships altogether, thus preserving any privacy that these relationships may compromise. We assume that while all of the sensitive relationships are removed, all or a portion of the relationships of other types are left intact in the anonymized data. We propose a method which allows modeling the influence of data attributes on sensitive relationships, and studying how different anonymization techniques can preserve privacy. The process of anonymization involves taking the unanonymized graph data, making some modifications, and constructing a new released graph which will be made available to the adversary. The modifications include changes to both the nodes and edges of the graph. The privacy breach is measured by counting the number of sensitive relationships that can be inferred from the anonymized data. The utility of the data is measured by counting how many attributes or observations have to be deleted in the sanitization process.

The effectiveness of the anonymization approaches depends on the structural and statistical characteristics of the underlying graph. In order to study the influence of each anonymization approach on privacy preservation, we apply them to synthetic data generated under varying statistical and structural assumptions and compute the information disclosed. We show how many relationships are revealed at different probability thresholds. Figure 5 shows representative results. For full details, see [82].
4.3 DualNet

We have developed several tools for visual analytics applied to graph data [9, 36, 56], and as part of the proposed work will develop a novel user interface that allows users to navigate the data graph and information graph in a coordinated manner. Figure 6 shows a DualNet prototype that we have developed which provides coordinated views of a single network. DualNet is a Java application built using the open source Prefuse information visualization toolkit [29]. As seen in Figure 6, DualNet consists of two panels, showing dual views of the network. Each panel is identical, consisting of a display panel on the top and four control and information tabs on the bottom. The controls and filters for each panel are independent; options and filters applied on one panel will not be applied to the second panel. The four tabs (Network, Filters, Properties, Search) provide users with information and controls for manipulating the network. The In addition, the Graph Type option allows users to choose how to visually represent the network.

The novelty of DualNet is that, rather than treating the network data as a single indivisible collection, the tool allows the user to reduce the size of the network by selecting meaningful subnetworks of the overall collection. The subnetworks can be a natural subset of the data (i.e., subnetworks created by selecting nodes belonging to a given group defined in the network), a manually generated subset of interest for a task (i.e., the set of interesting nodes and edges users have selected explicitly), or the subnetworks resulting from merging or splitting data items (i.e., multiple email addresses merged to represent the person using them). The tool allows any combination of subnetworks to be represented in separate, interactive visualizations where each subnetwork can be displayed and manipulated with the appropriate representation and controls. The tool supports
linking between the different network views. Selection of nodes and edges in one view highlights the corresponding set of nodes and edges in the other views. This allows users to cross-reference visualizations to see how different subnetworks and attributes correlate with each other. For example, if one visualization groups people with the same manager as a single node and the other shows each person as a node, selection of a manager node in the first view will highlight all the people nodes, with that same manager, in the second view.

Figure 6 shows a DualNet view of the the well-studied Enron email collection [39]. For our evaluation, we focused on a subset of the collection containing email addresses from 2000-2001, selected specifically because we have documented information about the titles and positions for the individuals who used these email addresses [18, 55]. We also used counts of the email communications between all those individuals in 2000-2001. The network graph constructed from this data set resulted on a network consisting of 119 email addresses and 1140 directed edges representing the amount and direction of email communications between nodes.

5 Research Plan

Throughout the project, we will develop and maintain a site with code, datasets, and data generators available. We will also seek out collaborations with users in domains such as biology, vision, and computational linguistics.

Year 1: We will build a simple, end-to-end, implementation of GID in Java. The local classifiers that we will use will include logistic regression and naive Bayes, and the inference algorithms will include ICA and Gibbs sampling. We will support the simpler aggregation functions. We will do initial testing of various granularities for interleaving the basic operations.

Year 2: We will develop the appropriate indexing and maintenance structures to make the combined collective inference efficient. We will study the impact of various methods of propagating information, and will implement belief propagation. We will support more complex aggregation representations, and integrate them into the inference algorithms. We will work on the user interface, to help us evaluate the results.

Year 3: We will focus on how structure affects identification. We will begin looking at the information hiding capabilities of GID. We will release a beta version of the software, making it available open source to the research community.

Year 4: We will focus on applications areas, ensuring that the primitives we currently envision are adequate. We will focus on ways of extending the user interface to support users in different domains.

Year 5: We will release version 1.0 of the tool, together with the user interface, and sample domain datasets.

We have several collaborators that have expressed interest in the framework. We are currently collaborating with [redacted] and her group at Lawrence Livermore National Laboratories and expect them to be alpha testers for our code. We have also spoken with [redacted] and his group at Carnegie Mellon University, and hope that they will be early users of our framework.

Current Support The PI currently has NSF support for NSF Grant [redacted], [redacted], [redacted], [redacted], [redacted], and a collaborative proposal, [redacted], [redacted], [redacted]. All three of these are in their last year of funding. This Career proposal builds on earlier results, most notably from the first award, NSF Grant [redacted].
6. Educational Plan

Objectives  My objectives in education are consistent with my research philosophy: to promote students’ solid theoretical foundations and apply their research to interesting real-world problems.

6.1. Student Research Development

I believe it is important to provide students with an environment where they can become familiar with current research, where they can develop their skills in analyzing and evaluating research, where they have an opportunity to discuss new research ideas, and where they have an opportunity to present their research results. I plan to do this both on a small scale within my own research group (http://www.cs.____.edu/______) and on a larger scale, within the artificial intelligence group at the University of __________. I think it is very important to develop a local community of student researchers. At __________, my research work with other students was an incredibly rewarding and fulfilling aspect of my education.

Since joining the University of __________ in December 2001, I have organized several reading groups for seniors and graduate students on advanced topics in data mining and probabilistic models. The reading groups were lively and well attended. I have taught a seminar class on statistical relational learning, and next year will be teaching a seminar class on link mining. I also organized a series of talks, ____________, to give students greater exposure to the research being done in artificial intelligence at __________ and organized AI Day at __________, which gave students an opportunity to present their work, and was attended by over 100 students and faculty.

Further, I believe it is very important to encourage students to consider academic careers and research. I encourage students by providing resources, and encouragement wherever possible. Each year, I have given a one hour talk, “PhishingD: Now What?”, in the introductory graduate “How to Do Research” course. A number of students have expressed their appreciation for my talk and advice. I am also committed to encouraging students outside my institution. For example, I gave a talk at the Workshop for Women in Machine Learning, held at the Grace Hopper Conference in 2006, I have participated in the SIGART/AAAII Doctoral Consortium, which provides students feedback and advice on their research, and I plan to pursue further activities along these lines.

6.2. Support for Underrepresented Groups

I have a track record of involvement and support for underrepresented groups in computer science. As an undergraduate, I was an instructor for the Summer Transition Enrichment Program, a program designed to help incoming freshmen from disadvantaged backgrounds gain pre-college experience. I was a computer science peer counselor, providing academic advising to students. I also tutored students in computer science and math. As a graduate student, I was an active participant in the __________ Women in Computer Science program, securing funding for the group’s social activities. I also tutored students at a local low-income elementary school in math.

Since I am a woman and an example of a successful returning PhD student (I earned my PhD after working in industry for several years), I hope to provide a role model for other women and non-traditional students. I also plan to develop my skills as a mentor. The beneficial impact of mentors, especially for women in male-dominated fields, and for underrepresented groups, has been well documented [61, 43]. I had the good fortune to have several very influential mentors, so I appreciate the positive impact that they can have. I am currently involved in both the undergraduate and the graduate women in computer science programs at __________.
6.3. Course Development

**Link Mining** In spring 2008, I will be teaching a graduate seminar in link mining. My goal is to have the GID software platform at a stage in development where students in the course can use it as a testbed for understanding and empirically evaluating the algorithms we will discuss in class. In 2005, I taught a graduate seminar in Statistical Relational Learning, and from that produced a tutorial which I have given at several conferences and a book [27], which will be available in August, 2007 by MIT Press. For the link mining topic, I also plan to develop a tutorial and a book.

**Principles of Data Mining and Machine Learning** In spring of 2002, I developed and taught a new course in Principles of Data Mining, and in spring 2004 and spring 2006, I taught a revamped graduate machine learning course. The goal of the both courses was to give students a solid statistical foundation, an in-depth understanding of the major tasks and algorithms and a hands-on experience building and applying these tools. I developed a collection of course materials, including a series of powerpoint slides, that are available on the web. A major component of each course was a term project, in which I gave students experience with the submission and revision of a research paper, the process of reviewing others' research projects, and final project presentations. At least eight of the course projects have developed into papers that were accepted at workshops or conferences.

**Artificial Intelligence** I also regularly teach an upper-level undergraduate course in artificial intelligence. The emphasis in the course is two-fold. First, the course provides a solid background in knowledge representation, reasoning (including search, constraint satisfaction and planning) and learning. The field of artificial intelligence has matured to a state where a rigorous presentation of the algorithms and their analysis is possible, and I teach my AI course with this emphasis. Second, the course emphasizes interesting and topical applications of the algorithms to real-world problems such as information extraction, robot navigation, and web search. As with my data mining course, I have made the course materials available to other instructors.

**Other Courses** I have a commitment to, and track record of, creating tutorials and presentations that help disseminate the ideas in my research area. I have given tutorials on statistical relational learning at the International Machine Learning Conference (ICML), the Inductive Logic Programming Conference (ILP), the International Joint Conference on Artificial Intelligence (IJCAI), and I will be giving a tutorial at the European Conference on Machine Learning (ECML) this fall. I will be giving a tutorial on “Data and Meta-data Alignment” at the Workshop on Foundations of Databases and the Web, and will be giving an invited talk at the AAAI conference on July, a keynote talk at the Workshop on Mining and Learning from Graphs in August, and an invited talk at a workshop on probabilistic databases at the Conference on Very Large Databases in September.

6.4. Building Ties with the International Research Community

One of the exciting things about my research area is the number of different research communities that it impacts. In particular, there is an active European research community involved in relational learning. Historically this community has employed a logic-based approach to relational learning [53, 42]; however, recently there has been a surge of interest in more probabilistic approaches to the problem. More than many researchers in my area, I have made an effort to bridge and bring together U.S. and international scholars. This includes both my collaborations with international scholars and more general research efforts. When I organized the AAAI 2000 Workshop on Learning Statistical Models from Relational Data, I made an effort to recruit members from the (European) ILP community. I have participated in European workshops, served on the program committee for many international conferences including the main ILP conference, and given invited talks at many international universities.