Ranking-Based Classification of Heterogeneous Information Networks

Ming Ji
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL, USA
mingji1@illinois.edu

Jiawei Han
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL, USA
hanj@illinois.edu

Marina Danilevsky
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL, USA
danilev1@illinois.edu

ABSTRACT

It has been recently recognized that heterogeneous information networks composed of multiple types of nodes and links are prevalent in the real world. Both classification and ranking of the nodes (or data objects) in such networks are essential for network analysis. However, so far these approaches have generally been performed separately. In this paper, we combine ranking and classification in order to perform more accurate analysis of a heterogeneous information network. Our intuition is that highly ranked objects within a class should play more important roles in classification. On the other hand, class membership information is important for determining a quality ranking over a dataset. We believe it is therefore beneficial to integrate classification and ranking in a simultaneous, mutually enhancing process, and to this end, propose a novel ranking-based iterative classification framework, called RankClass. Specifically, we build a graph-based ranking model to iteratively compute the ranking distribution of the objects within each class. At each iteration, according to the current ranking results, the graph structure used in the ranking algorithm is adjusted so that the subnetwork corresponding to the specific class is emphasized, while the rest of the network is weakened. As our experiments show, integrating ranking with classification not only generates more accurate classes than the state-of-art classification methods on networked data, but also provides meaningful ranking of objects within each class, serving as a more informative view of the data than traditional classification.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data Mining*

General Terms

Algorithms

Keywords

heterogeneous information network, ranking, classification

KDD'11, August 21–24, 2011, San Diego, California, USA.

1. INTRODUCTION

Information networks have been found to play increasingly important roles in real life applications. Examples include friendship networks in Facebook[1], co-author networks extracted from bibliographic data, and webpages interconnected by hyperlinks on the Web. Networks and feature vectors are two alternatives to represent the data, and the former is often more natural than the latter in many data sets [8]. Even if the data is naturally represented in a feature space, it is usually helpful to transform the data into a network, or graph structure (for example, by constructing a nearest neighbor graph) to better exploit the intrinsic characteristics of the data. Therefore, learning on networked data is receiving growing attention in recent years [9, 22, 1, 24, 6]. Most of the existing studies [19, 16, 2, 12] about information networks mainly work with homogeneous networks, i.e., networks composed of a single type of object, as mentioned above. However, heterogeneous networks composed of multiple types of objects are more general and prevalent in many real world applications [21, 11, 3]. For example, beyond co-author networks, bibliographic data actually forms a heterogeneous information network consisting of multi-typed objects, such as papers, authors, venues and terms.

Example 1. Bibliographic Information Network. A bibliographic information network generally contains four types of objects: *papers*, *authors*, *venues* (conferences and journals) and *terms*. *Papers* and *authors* are linked by the relation of "written by" and "write". *Papers* and *venues* are linked by "published in" and "publish". *Papers* and *terms* are linked by "contain" and "contained in". ■

In this paper, we study the analysis of heterogeneous information networks. *Classification* and *ranking* are two of the most fundamental analytical techniques. When label information is available for some of the data objects, *classification* makes use of the labeled data as well as the network structure to predict the class membership of the unlabeled data [19, 16]. On the other hand, *ranking* gives a partial ordering to objects in the network by evaluating the node/link properties using some ranking scheme, such as PageRank [2] or HITS [12]. Both classification and ranking have been widely studied and found to be applicable in a wide range of problems.

Traditionally, classification and ranking are regarded as orthogonal approaches, computed independently. However, adhering to such a strict dichotomy has serious downsides. Consider, for instance, an information network of bibliographic data, consisting of some combination of published papers, authors, and conferences. As a concrete example, suppose we wish to classify the conferences in Table 1 into two research areas. We wish to minimize the chance

[1]http://www.facebook.com/

Table 1: Conferences from two research areas

Database	SIGMOD, VLDB, ICDE, EDBT, PODS, ...
Information Retrieval	SIGIR, ECIR, CIKM, WWW, WSDM, ...

Table 2: Top-5 ranked conferences in different settings

Rank	Global Ranking	Within DB	Within IR
1	VLDB	VLDB	SIGIR
2	SIGIR	SIGMOD	ECIR
3	SIGMOD	ICDE	WWW
4	ICDE	PODS	CIKM
5	ECIR	EDBT	WSDM

that the top conferences are misclassified, not only to improve our classification results overall, but also because misclassifying a top conference is very likely to increase errors on many other objects that link to that conference, and are therefore greatly influenced by its label. We would thus like to more heavily penalize classification mistakes made on highly ranked conferences, relative to a workshop of little influence. Providing a ranking of all conferences within a research area can give users a clearer understanding of that field, rather than simply grouping conferences into classes without noting their relative importance. On the other hand, the class membership of each conference is very valuable for characterizing that conference. Ranking all conferences globally without considering any class information can often lead to meaningless results and apples-to-oranges comparisons. For instance, ranking database and information retrieval conferences together may not make much sense since the top conferences in these two fields cannot be reasonably compared, as shown in the second column of Table 2. These kinds of nonsensical ranking results are not caused by the specific ranking approach, but are rather due to the inherent incomparability between the two classes of conferences. Thus we suppose that combining classification with ranking may generate more informative results. The third and fourth columns in Table 2 illustrate this combined approach, showing the more meaningful conference ranking within each class.

In this study, we propose RankClass, a new framework that groups objects into several pre-specified classes, while generating the ranking information for each type of object within each class simultaneously in a heterogeneous information network. More accurate classification of objects increases the quality of the ranking within each class, since there is a higher guarantee that the ranking algorithm used will be comparing only objects of the same class. On the other hand, better ranking scores improve the performance of the classifier, by correctly identifying which objects are more important, and should therefore have a higher influence on the classifier's decisions. We use the ranking distribution of objects to characterize each class, and we treat each object's label information as a prior. By building a graph-based ranking model, different types of objects are ranked simultaneously within each class. Based on these ranking results, we estimate the relative importance or visibility of different parts of the network with regard to each class. In order to generate better within-class ranking, the network structure employed by the ranking model is adjusted so that the sub-network composed of objects ranked high in each specific class is emphasized, while the sub-network of the rest of the class is gradually weakened. Thus, as the network structure of each class becomes clearer, the ranking quality improves. Finally, the posterior probability of each object belonging to each class is estimated to determine each object's optimal class membership. Instead of performing ranking after classification, as facet ranking does [4, 23], RankClass essentially integrates ranking and classification, allowing both approaches to mutually enhance each other. RankClass iterates over this process until converging to a stable state. Experimental results show that RankClass both boosts the overall classification accuracy and constructs within-class rankings, which may be interpreted as meaningful summaries of each class.

The rest of this paper is organized as follows. We briefly review the related work in Section 2. We introduce related concepts and formally define our problem in Section 3. Our ranking-based classification algorithm, RankClass, is introduced in Section 4. Section 5 presents the experimental results, and we conclude this work in Section 6.

2. RELATED WORK

As data sets with inherent network structures become increasingly prevalent, ranking networked objects has received substantial interest in recent years. Two important representative algorithms are PageRank [2] and HITS [12], both of which propagate information throughout the network to compute the ranking score of each object, using different propagation methods corresponding to different ranking rules. These methods mainly work on homogeneous information networks. Recently, PopRank [18] was proposed to rank the popularity of heterogeneous web objects via knowledge propagation throughout the heterogeneous network of web objects. This approach considers that different types of links in a network have different propagation factors, which are trained according to partial ranks given by experts. In contrast, we rank objects according to their importance within each class, rather than within the global set of all objects, and the ranking results in turn facilitate more accurate classification.

Classification is an essential tool in analyzing information networks when some object label information is available [25, 20]. Collective classification [14, 19, 17] has been proposed to employ both the network structure and the feature representation of objects in the classification task. Since local features may not be always available, Macskassy et al. [15] develop a relational neighbor classifier to classify network-only data by iteratively assigning an object to the majority class of its neighbors. This idea is similar to the label propagation scheme in graph-based classification [26]. However, existing algorithms mainly work on homogeneous networks and graphs, and therefore cannot easily distinguish between the type differences among objects in a heterogeneous information network. Recently, the graph-based classification framework has been extended to work on heterogeneous information networks [11]. In this paper, we build on this approach by providing within-class ranking for objects in the information network, which can improve classification results by providing informative summaries of each class.

To enhance the quality of classification, boosting, bagging and ensemble methods have been explored in various studies [10]. In particular, boosting methods such as AdaBoost [5] iteratively learn from their classification mistakes by assigning higher weights to objects which are misclassified in each previous round, until a stable classification state is reached. Like boosting, RankClass also adjusts the relative importance of objects in various rounds of classification. However, RankClass uses within-class ranking to measure the importance of each object with regard to each class, in contrast to boosting, which estimates the global importance of each object based on classification mistakes.

Another relevant algorithm is the newly proposed NetClus [21] method, which uses a ranking-clustering mutual enhancement methodology to cluster objects in heterogeneous information networks. Although this method effectively provides a ranking within each cluster, it has some limitations: (1) it can only work on heteroge-

neous information networks with a star schema; and (2) it requires a prior distribution specified by several labeled *representative* objects of each cluster, and does not work well with arbitrary labeled objects, which may not be representative. Thus, if we do not know which objects are representative in a data set, NetClus cannot be used. However, for heterogeneous information networks with arbitrary network schema, our proposed RankClass algorithm can make full use of label information available for any data objects to generate accurate classification results and informative rankings.

3. PROBLEM FORMALIZATION

In this section, we introduce several related concepts and notations, and then formally define the problem.

Definition 1. **Heterogeneous information network.** Given m types of data objects, denoted by $\mathcal{X}_1 = \{x_{11}, \ldots, x_{1n_1}\}, \ldots, \mathcal{X}_m = \{x_{m1}, \ldots, x_{mn_m}\}$, a graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E}, \mathcal{W} \rangle$ is called a heterogeneous information network if $\mathcal{V} = \bigcup_{i=1}^{m} \mathcal{X}_i$ and $m \geq 2$. $\mathcal{E}$ is the set of links between any two data objects of $\mathcal{V}$, and $\mathcal{W}$ is the set of weight values on the links. When $m = 1$, $\mathcal{G}$ reduces to a homogeneous information network. ■

In the following sections, for convenience, we use $\mathcal{X}_i$ to denote both the set of objects belonging to the i-th type and the type name. We let $\mathcal{W}_{x_{ip}x_{jq}}$ denote the weight of the link between any two objects x_{ip} and x_{jq}, which is represented by $\langle x_{ip}, x_{jq} \rangle$.

In a heterogeneous information network, each type of link relationship between two types of data objects $\mathcal{X}_i$ and $\mathcal{X}_j$ can be represented by a relation graph $\mathcal{G}_{ij}$, $i, j \in \{1, \ldots, m\}$. Note that it is possible for $i = j$. Let $\mathbf{R}_{ij}$ be an $n_i \times n_j$ relation matrix corresponding to graph $\mathcal{G}_{ij}$. The element at the p-th row and q-th column of $\mathbf{R}_{ij}$ is denoted as $R_{ij,pq}$, representing the weight on link $\langle x_{ip}, x_{jq} \rangle$. There are many ways to define the weights on the links, which can also incorporate domain knowledge. A simple definition is as follows:

$$R_{ij,pq} = \begin{cases} 1 & \text{if data objects } x_{ip} \text{ and } x_{jq} \text{ are linked together} \\ 0 & \text{otherwise.} \end{cases}$$

Here we consider undirected relation graphs such that $\mathbf{R}_{ij} = \mathbf{R}_{ji}^T$. In this way, each heterogeneous network $\mathcal{G}$ can be mathematically represented by a set of relation matrices $\mathcal{G} = \{\mathbf{R}_{ij}\}_{i,j=1}^{m}$.

To naturally generalize classification in homogeneous network data, we define a class in a heterogeneous information network to be a group of multi-typed objects sharing a common topic. For instance, a research community in a bibliographic information network contains not only authors, but also papers, venues and terms belonging to the same research area. Other examples include movie networks in which movies, directors, actors and keywords are tagged with the same genre, and E-commerce networks where sellers, customers, items and tags belong to the same shopping category. The formal definition of a *class* is given below:

Definition 2. **Class.** Given a heterogeneous information network $\mathcal{G} = \langle \mathcal{V}, \mathcal{E}, \mathcal{W} \rangle$, $\mathcal{V} = \bigcup_{i=1}^{m} \mathcal{X}_i$, a class is defined as $\mathcal{G}' = \langle \mathcal{V}', \mathcal{E}', \mathcal{W}' \rangle$, where $\mathcal{V}' \subseteq \mathcal{V}$, $\mathcal{E}' \subseteq \mathcal{E}$. $\forall e = \langle x_{ip}, x_{jq} \rangle \in \mathcal{E}'$, where $x_{ip} \in \mathcal{V}'$ and $x_{jq} \in \mathcal{V}'$, we have $\mathcal{W}'_{x_{ip}x_{jq}} = \mathcal{W}_{x_{ip}x_{jq}}$. Note here, $\mathcal{V}'$ also consists of multiple types of objects from $\mathcal{X}_1$ to $\mathcal{X}_m$. ■

Definition 2 follows [21] and [11]. Notice that a class in a heterogeneous information network is actually a sub-network containing multi-typed objects that are closely related to each other. In addition to grouping multi-typed objects into the pre-specified K classes, we also aim to generate the ranking distribution of objects within each class k, which can be denoted as $P(x|T(x), k)$, $k = 1, \ldots, K$. $T(x)$ denotes the type of object x. Note that different types of objects cannot be compared in a ranking. For example, it is not meaningful to create a ranking of conferences and authors together in a bibliographic information network. Therefore, each ranking distribution is restricted to a single object type, i.e., $\sum_{p=1}^{n_i} P(x_{ip}|\mathcal{X}_i, k) = 1$.

Now our problem can be formalized as follows: given a heterogeneous information network $\mathcal{G} = \langle \mathcal{V}, \mathcal{E}, \mathcal{W} \rangle$, a subset of data objects $\mathcal{V}' \subseteq \mathcal{V} = \bigcup_{i=1}^{m} \mathcal{X}_i$, which are labeled with values $\mathcal{Y}$ denoting which of the K pre-specified classes each object belongs to, predict the class labels for all the unlabeled objects $\mathcal{V} - \mathcal{V}'$ as well as the ranking distribution of objects within each class, $P(x|T(x), k)$, $x \in \mathcal{V}$, $k = 1, \ldots, K$.

4. THE RANKCLASS ALGORITHM

In this section we introduce our ranking-based iterative classification method, RankClass. There are two major challenges when working with heterogeneous information networks: (1) how to exploit the links representing the dependency relationships between data objects; and (2) how to model the type differences among objects and links. The intuition behind RankClass is to build a graph-based ranking model that ranks multi-typed objects simultaneously, according to the relative importance of objects within each class. The initial ranking distribution of each class is determined by the labeled data. During each iteration, the ranking results are used to modify the network structure to allow the ranking model to generate higher quality within-class ranking.

4.1 The Framework of RankClass

We first introduce the general framework of RankClass. We will explain each part of the algorithm in detail in the following subsections.

- Step 0: Initialize the ranking distribution within each class according to the labeled data, i.e., $\{P(x|T(x), k)^0\}_{k=1}^{K}$. Initialize the set of network structures employed in the ranking model, $\{\mathcal{G}_k^0\}_{k=1}^{K}$, as $\mathcal{G}_k^0 = \mathcal{G}$, $k = 1, \ldots, K$. Initialize $t = 1$.
- Step 1: Using the graph-based ranking model and the current set of network structures $\{\mathcal{G}_k^{t-1}\}_{k=1}^{K}$, update the ranking distribution within each class k, i.e., $\{P(x|T(x), k)^t\}_{k=1}^{K}$.
- Step 2: Based on $\{P(x|T(x), k)^t\}_{k=1}^{K}$, adjust the network structure to favor within-class ranking, i.e., $\{\mathcal{G}_k^t\}_{k=1}^{K}$.
- Step 3: Repeat steps 1 and 2, setting $t = t + 1$ until convergence, i.e., until $\{P(x|T(x), k)^*\}_{k=1}^{K} = \{P(x|T(x), k)^t\}_{k=1}^{K}$ do not change much for all $x \in \mathcal{V}$.
- Step 4: Based on $\{P(x|T(x), k)^*\}_{k=1}^{K}$, calculate the posterior probability for each object, i.e., $\{P(k|x, T(x))\}_{k=1}^{K}$. Assign the class label to object x as:

$$C(x) = \arg\max_{1 \leq k \leq K} P(k|x, T(x))$$

4.2 Graph-based Ranking

Ranking is often used to evaluate the relative importance of objects in a collection. In this paper, we propose to rank objects within their own type and within a specific class. The higher an object x is ranked within class k, the more important x is for class k, and the

more likely it is that x will be visited in class k. Clearly, within-class ranking is quite different from global ranking, and will vary throughout different classes.

The intuitive idea of our ranking scheme is authority propagation throughout the information network. Taking the bibliographic information network as an example, in a specific research area, it is natural to observe the following ranking rules [21]:

1. Highly ranked conferences publish many high quality papers.
2. High quality papers are often written by highly ranked authors.
3. High quality papers often contain keywords that are highly representative of the papers' areas.

The above authority ranking rules can be summarized as follows: objects which are linked together in a network are more likely to share similar ranking scores. Therefore, the ranking of each object can be iteratively updated by looking at the rankings of its neighbors. The initial ranking distribution within a class k can be specified by the user. When data objects are labeled without ranking information in a general classification scenario, we can initialize the ranking as a uniform distribution over only the labeled data objects:

$$P(x_{ip}|\mathcal{X}_i,k)^0 = \begin{cases} 1/l_{ik} & \text{if } x_{ip} \text{ is labeled to class } k \\ 0 & \text{otherwise.} \end{cases}$$

where l_{ik} denotes the total number of objects of type $\mathcal{X}_i$ labeled to class k.

Suppose the current network structure used to estimate the ranking within class k is mathematically represented by the set of relation matrices: $\mathcal{G}_k^{t-1} = \{\mathbf{R}_{ij}\}_{i,j=1}^m$. For each relation matrix $\mathbf{R}_{ij}$, we define a diagonal matrix $\mathbf{D}_{ij}$ of size $n_i \times n_i$. The (p,p)-th element of $\mathbf{D}_{ij}$ is the sum of the p-th row of $\mathbf{R}_{ij}$. Instead of using the original relation matrices in the authority propagation, we construct the normalized form of the relation matrices as follows:

$$\mathbf{S}_{ij} = \mathbf{D}_{ij}^{(-1/2)}\mathbf{R}_{ij}\mathbf{D}_{ji}^{(-1/2)}, i,j \in \{1,\ldots,m\} \quad (1)$$

This normalization technique is adopted in traditional graph-based learning [26] in order to reduce the impact of node popularity. In other words, we can suppress popular nodes to some extent, to keep them from completely dominating the authority propagation. Notice that the normalization is applied separately to each relation matrix corresponding to each type of link, rather than the whole network. In this way, the type differences between objects and links are well-preserved [11]. At the t-th iteration, the ranking distribution of object x_{ip} with regard to class k is updated as follows:

$$P(x_{ip}|\mathcal{X}_i,k)^t \propto \frac{\sum_{j=1}^m \lambda_{ij} S_{ij,pq} P(x_{jq}|\mathcal{X}_j,k)^{t-1} + \alpha_i P(x_{ip}|\mathcal{X}_i,k)^0}{\sum_{j=1}^m \lambda_{ij} + \alpha_i} \quad (2)$$

The first term of Equation (2) updates the ranking score of object x_{ip} by the summation of the ranking scores of its neighbors x_{jq}, weighted by the link strength $S_{ij,pq}$. The relative importance of neighbors of different types is controlled by $\lambda_{ij} \in [0,1]$. The larger the value of λ_{ij}, the more value is placed on the relationship between object types $\mathcal{X}_i$ and $\mathcal{X}_j$. For example, in a bibliographic information network, if a user believes that the links between *authors* and *papers* are more trustworthy and influential than the links between *conferences* and *papers*, then the λ_{ij} corresponding to the *author-paper* relationship should be set larger than that of *conference-paper*. As a result, the rank of a paper will rely more on the ranks of its authors than the rank of its publication venue. The parameters λ_{ij} can also be thought of as performing feature selection in the heterogeneous information network, i.e., selecting which types of links are important in the ranking process.

The second term learns from the initial ranking distribution encoded in the labels, whose contribution is weighted by $\alpha_i \in [0,1]$. A similar strategy has been adopted in [13, 11] to control the weights between different types of relations and objects. After each iteration, $P(x_{ip}|\mathcal{X}_i,k)^t$ is normalized such that $\sum_{p=1}^{n_i} P(x_{ip}|\mathcal{X}_i,k)^t = 1, \forall i = 1,\ldots,m, k = 1,\ldots,K$, in order to stay consistent with the mathematical definition of a ranking distribution.

We employ the authority propagation scheme in Equation (2) to estimate the ranking distribution instead of other simple measures computed according to the network topology (e.g., the degree of each object). This choice was made since we aim to rank objects with regard to each class by utilizing the current soft classification results. Therefore, if the ranking of an object were merely based on the network topology, it would be the same for all classes. By learning from the label information in the graph-based authority propagation method, the ranking of each object within different classes will be computed differently, which is more suitable for our setting.

Following a similar analysis to [11] and [27], the updating scheme in Equation (2) can be proven to converge to the closed form solution of minimizing the following objective function:

$$\begin{aligned} & J(P(x_{ip}|\mathcal{X}_i,k)) \\ = & \sum_{i,j=1}^m \lambda_{ij} \sum_{p=1}^{n_i}\sum_{q=1}^{n_j} S_{ij,pq}\left(P(x_{ip}|\mathcal{X}_i,k) - P(x_{jq}|\mathcal{X}_j,k)\right)^2 \\ & + \sum_{i=1}^m \alpha_i \sum_{p=1}^{n_i} (P(x_{ip}|\mathcal{X}_i,k) - P(x_{ip}|\mathcal{X}_i,k)^0)^2 \end{aligned} \quad (3)$$

which shares a similar theoretical foundation with the graph-based regularization framework on heterogeneous information networks [11] that preserves consistency over each relation graph corresponding separately to each link type. However, we extend the graph-based regularization framework to rank objects within each class, which is conceptually different from [11].

4.3 Adjusting the Network

Although graph-based ranking considers class information by incorporating the labeled data, it still ranks all object types in the global network. Instead, a within-class ranking should be performed over the sub-network corresponding to each specific class. The cleaner the network structure, the higher our ranking quality. Therefore, the ranking within each class should be performed over a different sub-network, rather than employing the same global network for every class. The network structure is mathematically represented by the weight values on the links. Thus, extracting the sub-network belonging to class k is equivalent to increasing the weight on the links within the corresponding sub-network, and decreasing the weight on the links in the rest of the network. It is straightforward to verify that multiplying $\mathbf{R}_{ij}$ by any positive constant c will not change the value of $\mathbf{S}_{ij}$. So increasing the weights on the links within a sub-network should be performed relative to the weight on the links of other parts of the network. In other words, we can increase or decrease the absolute values of the weights on the links in the whole network, as long as the weights on the links of the sub-network belonging to class k are larger than those on the links belonging to the rest of the network. Let $\mathcal{G}_k^t = \{\mathbf{R}_{ij}^t(k)\}_{i,j=1}^m$. We propose a simple scheme to update the network structure so as to favor the ranking within each class k, given the current ranking

distribution $P(x|T(x),k)^t$:

$$R^t_{ij,pq}(k) = R_{ij,pq} \times \left(r(t) + \sqrt{\frac{P(x_{ip}|\mathcal{X}_i,k)^t}{\max_p P(x_{ip}|\mathcal{X}_i,k)^t}\frac{P(x_{jq}|\mathcal{X}_j,k)^t}{\max_q P(x_{jq}|\mathcal{X}_j,k)^t}}\right) \quad (4)$$

Recall that $\mathbf{R}_{ij}$ is the relation matrix corresponding to the links between object types $\mathcal{X}_i$ and $\mathcal{X}_j$ in the original network. Using the above updating scheme, the weight of each link $\langle x_{ip}, x_{jq}\rangle$ is increased in proportion to the geometric mean of the ranking scores of x_{ip} and x_{jq}, which are scaled to the interval of $[0, 1]$. The higher the rankings of x_{ip} and x_{jq}, the more important the link between them $\langle x_{ip}, x_{jq}\rangle$ is in class k. The weight on that link should therefore be increased. Note that instead of creating hard partitions of the original network into classes, we simply increase the weights on the links that are important to classes k. This is because at any time in the iteration, the current classes represented by the ranking distributions are not very accurate, and the results will be more stable if we consider both the global network structure and the current ranking results. By gently increasing the weights of links in the sub-network of class k, we gradually extract the correct sub-network from the global network, since the weights of links in the rest of the network will decrease to very low values. Note that this adjustment of the network structure still respects the differences among the various types of objects and links.

$r(t)$ is a positive parameter that does not allow the weights of links to drop to 0 in the first several iterations, when the authority scores have not propagated very far throughout the network and $P(x|T(x),k)^t$ are close to 0 in value for many objects. As discussed above, multiplying $\mathbf{R}_{ij}$ by any positive constant will not change the value of $\mathbf{S}_{ij}$. Therefore, it is essentially the ratio between $r(t)$ and $\sqrt{\frac{P(x_{ip}|k)^t}{\max_p P(x_{ip}|k)^t} \times \frac{P(x_{jq}|k)^t}{\max_q P(x_{jq}|k)^t}}$ that determines how much the original network structure and the current ranking distribution, respectively, contribute to the adjusted network $\mathcal{G}^t_k$. Since we hope to progressively extract the sub-network belonging to each class k, and we want to gradually reduce the weights of links that do not belong to class k down to 0, we decrease $r(t)$ exponentially by setting $r(t) = \frac{1}{2^t}$.

Equation (4) is not the only way to gradually increase the weights of links between highly ranked objects in class k. For instance, the geometric mean of $\frac{P(x_{ip}|k)^t}{\max_p P(x_{ip}|k)^t}$ and $\frac{P(x_{jq}|k)^t}{\max_q P(x_{jq}|k)^t}$ can be replaced by the arithmetic mean, and $r(t)$ can be any positive function that decreases with t. We will show in Section 5 that even such simple adjustments as shown above can boost both the classification and ranking performance of RankClass.

4.4 Posterior Probability Calculation

Once the ranking distribution of each class has been computed by the iterative algorithm, we can calculate the posterior probability of each object of type $\mathcal{X}_i$ belonging to class k simply by Bayes' rule:

$$P(k|x_{ip}, \mathcal{X}_i) \propto P(x_{ip}|\mathcal{X}_i, k)P(k|\mathcal{X}_i)$$

where $P(x_{ip}|\mathcal{X}_i, k) = P(x_{ip}|\mathcal{X}_i, k)^*$, and $P(k|\mathcal{X}_i)$ represents the relative size of class k among type $\mathcal{X}_i$, which should also be estimated. We choose the $P(k|\mathcal{X}_i)$ that maximizes the likelihood of generating the set of objects of type $\mathcal{X}_i$:

$$\log L(x_{i1}, \ldots, x_{in_i}|\mathcal{X}_i) = \sum_{p=1}^{n_i} \log P(x_{ip}|\mathcal{X}_i)$$

$$= \sum_{p=1}^{n_i} \log\left(\sum_{k=1}^{K} P(x_{ip}|\mathcal{X}_i, k)P(k|\mathcal{X}_i)\right) \quad (5)$$

By employing the EM algorithm, $P(k|\mathcal{X}_i)$ can be iteratively estimated using the following two equations:

$$P(k|x_{ip}, \mathcal{X}_i)^t \propto P(x_{ip}|\mathcal{X}_i, k)P(k|\mathcal{X}_i)^t$$

$$P(k|\mathcal{X}_i)^t = \sum_{p=1}^{n_i} P(k|x_{ip}, \mathcal{X}_i)^t/n_i$$

where $P(k|\mathcal{X}_i)$ is initialized uniformly as $P(k|\mathcal{X}_i)^0 = 1/K$.

4.5 Computational Complexity Analysis

In this subsection, we analyze the computational complexity of the proposed RankClass algorithm. Let K denote the number of classes, $|V|$ denote the total number of objects, and $|E|$ denote the total number of links in the information network. It takes $O(K|V|)$ time to initialize the ranking distribution in step 0. At each iteration of step 1, we need to process each link twice to update the ranking distribution, once for each object at each end of the link. We also need $O(K|V|)$ time to learn from the initial ranking distribution. So the total time complexity for step 1 at each iteration is $O(K(|E| + |V|))$. In step 2, we need $O(K|E|)$ time to adjust the network structure at each iteration. After the ranking distribution is computed, we need $O(K|V|)$ time at each iteration of the EM algorithm to calculate the posterior probability. Finally, it takes $O(K|V|)$ time to generate the final class prediction in step 4. Hence the total time complexity of the RankClass algorithm is $O\big(N_1K(|E| + |V|) + N_2K|V|\big)$, where N_1 is the number of iterations in the computation of the ranking distribution, and N_2 is the number of iterations in the EM algorithm. We will experimentally demonstrate that this algorithm converges in a few iterations. And since the number of classes K is constant, the computational complexity is generally linear in the number of links and objects in the network.

5. EXPERIMENTS

In this section, we apply our proposed ranking-based classification scheme, RankClass, to a real heterogeneous information network extracted from the DBLP[2] database. We try to classify the bibliographic data into research communities, each of which consists of multi-typed objects closely related to the same area. All of the experiments were conducted on a PC with 3.00GHz CPU and 8GB memory. The following five classification methods on information networks are compared:

- Our proposed RankClass algorithm (RankClass).
- Graph-based regularization framework for transductive classification in heterogeneous information networks (GNetMine) [11].
- Learning with Local and Global Consistency (LLGC) [26].
- Weighted-vote Relational Neighbor Classifier (wvRN) [15, 16].
- Network-only Link-based Classification (nLB) [19, 16].

LLGC is a graph-based transductive classification algorithm for homogeneous networks, while GNetMine is its extension, which works on heterogeneous information networks. Weighted-vote relational neighbor classifier and link-based classification are two popular classification methods for networked data. Since a feature representation of nodes is not available for our problem, we use the network-only derivative of the link-based classifier (nLB) [16],

[2] http://www.informatik.uni-trier.de/~ley/db/

Table 3: Comparison of classification accuracy on authors (%)

$(a\%, p\%)$ of authors and papers labeled	nLB (A-A)	nLB (A-C-P-T)	wvRN (A-A)	wvRN (A-C-P-T)	LLGC (A-A)	LLGC (A-C-P-T)	GNetMine (A-C-P-T)	RankClass (A-C-P-T)
(0.1%, 0.1%)	25.4	26.0	40.8	34.1	41.4	61.3	82.9	**85.4**
(0.2%, 0.2%)	28.3	26.0	46.0	41.2	44.7	62.2	83.4	**88.0**
(0.3%, 0.3%)	28.4	27.4	48.6	42.5	48.8	65.7	86.7	**88.5**
(0.4%, 0.4%)	30.7	26.7	46.3	45.6	48.7	66.0	87.2	**88.4**
(0.5%, 0.5%)	29.8	27.3	49.0	51.4	50.6	68.9	87.5	**89.2**
average	28.5	26.7	46.3	43.0	46.8	64.8	85.5	**87.9**

Table 4: Comparison of classification accuracy on papers (%)

$(a\%, p\%)$ of authors and papers labeled	nLB (P-P)	nLB (A-C-P-T)	wvRN (P-P)	wvRN (A-C-P-T)	LLGC (P-P)	LLGC (A-C-P-T)	GNetMine (A-C-P-T)	RankClass (A-C-P-T)
(0.1%, 0.1%)	49.8	31.5	62.0	42.0	67.2	62.7	**79.2**	77.7
(0.2%, 0.2%)	73.1	40.3	71.7	49.7	72.8	65.5	**83.5**	83.0
(0.3%, 0.3%)	77.9	35.4	77.9	54.3	76.8	66.6	83.2	**83.6**
(0.4%, 0.4%)	79.1	38.6	78.1	54.4	77.9	70.5	83.7	**84.7**
(0.5%, 0.5%)	80.7	39.3	77.9	53.5	79.0	73.5	84.1	**84.8**
average	72.1	37.0	73.5	50.8	74.7	67.8	82.7	**82.8**

Table 5: Comparison of classification accuracy on conferences (%)

$(a\%, p\%)$ of authors and papers labeled	nLB (A-C-P-T)	wvRN (A-C-P-T)	LLGC (A-C-P-T)	GNetMine (A-C-P-T)	RankClass (A-C-P-T)
(0.1%, 0.1%)	25.5	43.5	79.0	81.0	**85.0**
(0.2%, 0.2%)	22.5	56.0	83.5	85.0	**85.5**
(0.3%, 0.3%)	25.0	59.0	87.0	87.0	**90.0**
(0.4%, 0.4%)	25.0	57.0	86.5	89.5	**92.0**
(0.5%, 0.5%)	25.0	68.0	90.0	94.0	**95.0**
average	24.6	56.7	85.2	87.3	**89.5**

which creates a feature vector for each node based on neighboring information. Note that LLGC, wvRN and nLB are classifiers which work with homogeneous networks, and cannot be directly applied to heterogeneous information networks. In order to compare all of the above algorithms, we can transform the heterogeneous DBLP network into a homogeneous network in two ways (see Section 5.2): (1) disregard the type differences between objects and treat all objects as the same type; or (2) extract a homogeneous sub-network on one single type of object, if that object type is partially labeled. We try both approaches in the accuracy study. The open-source implementation of NetKit-SRL[3] [16] is employed in our experiments.

5.1 Data Preparation

We extracted a connected sub-network of the DBLP data set on four research areas: database, data mining, information retrieval and artificial intelligence, which naturally form four classes. As previously discussed, this heterogeneous information network is composed of four types of objects: paper, conference, author and term. Among the four types of objects, we have three types of link relationships: paper-conference, paper-author, and paper-term. The data set we used contains 14376 papers, 20 conferences, 14475 authors and 8920 terms, with a total number of 170794 links[4].

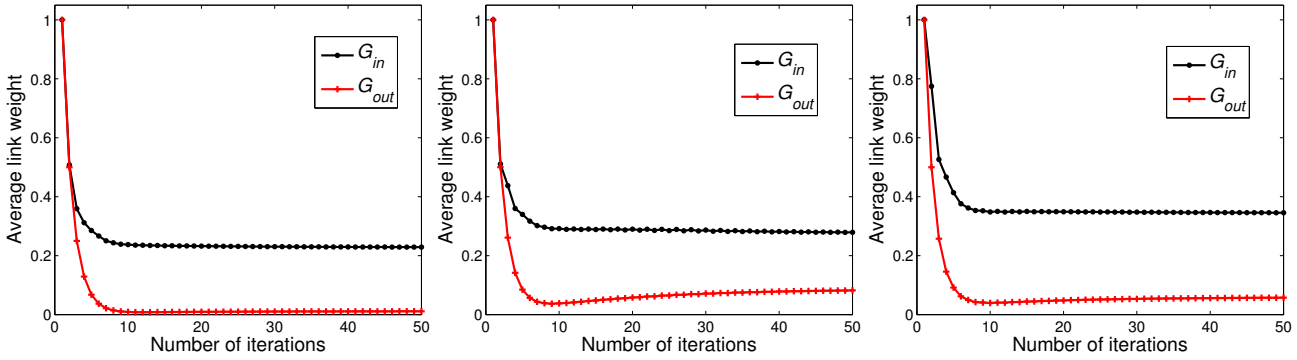
For accuracy evaluation, we use a labeled data set of 4057 authors, 100 papers and all 20 conferences. For more details about the labeled data set, please refer to [7, 21]. In the following sections, we randomly choose a subset of labeled objects and use their label information in the learning process. The classification accuracy is evaluated by comparing with manually labeled results on the rest of the labeled objects. Since terms are difficult to label even manually, as many terms may belong to multiple areas, we do not evaluate the accuracy on terms here.

[3] http://www.research.rutgers.edu/~sofmac/NetKit.html

[4] The data set is available at www.cs.illinois.edu/homes/mingji1/DBLP_four_area.zip for sharing and experiment repeatability.

5.2 Accuracy Study

In order to address the label scarcity problem in real life, we randomly choose $(a\%, p\%) = [(0.1\%, 0.1\%), (0.2\%, 0.2\%), \ldots, (0.5\%, 0.5\%)]$ of authors and papers, and use their label information in the classification task. For each $(a\%, p\%)$, we average the performance scores over 10 random selections of the labeled set. We set the parameters of LLGC and GNetMine to optimal values, which were determined experimentally. For our proposed RankClass method, as discussed above, the parameters λ_{ij} are used to select which types of links are important in the ranking process. We consider all types of objects and links to be important in the DBLP network, so we follow [11] and set $\alpha_i = 0.1$, $\lambda_{ij} = 0.2$, $\forall i, j \in \{1, \ldots, m\}$. This may not be the optimal choice, but it is good enough to demonstrate the effectiveness of our algorithm. Since labels are given for selected authors and papers, the results on conferences of wvRN, nLB and LLGC can only be obtained by mining the original heterogeneous information network (denoted by A-C-P-T) and disregarding the type differences between objects and links. While classifying authors and papers, we also tried constructing homogeneous author-author (A-A) and paper-paper (P-P) sub-networks in various ways, where the best results reported for authors are given by the co-author network, and the best results for papers are generated by linking two papers if they are published in the same conference. Note that there is no label information given for conferences, so we cannot build a conference-conference (C-C) sub-network for classification. We show the classification accuracy



(a) Links connected to authors (b) Links connected to conferences (c) Links connected to papers

Figure 1: Link weight change in 50 iterations

on authors, papers and conferences in Tables 3, 4 and 5, respectively. The last row of each table records the average classification accuracy while varying the percentage of labeled data.

RankClass outperforms all other algorithms when classifying authors, papers and conferences. Note that even though the number of authors is much higher than the number of conferences, RankClass achieves comparable accuracy for both of these types of objects. While classifying authors and papers, it is interesting to note that wvRN and nLB perform better on the *author-author* and *paper-paper* sub-networks than on the whole heterogeneous information network. We observe a similar result when we use LLGC to classify papers. These results serve to verify that homogeneous classifiers like wvRN, nLB and LLGC are more suitable for working with homogeneous data. However, transforming the heterogeneous information network into homogeneous sub-networks inevitably results in information loss. For example, in the *author-author* sub-network, the conferences where each author often publishes papers, and the terms that each author likes to use, are no longer known. Overall, GNetMine performs the second best by explicitly respecting the type differences in links and objects and thus encoding the typed information in the heterogeneous network in an organized way. Compared to GNetMine, RankClass achieves 16.6%, 0.58% and 17.3% relative error reduction in the average classification accuracy when classifying authors, papers and conferences, respectively. Although RankClass has a knowledge propagation framework similar to that of GNetMine, RankClass aims to compute the within-class ranking distribution to characterize each class, and further employs the ranking results to iteratively extract the sub-network corresponding to each specific class. Therefore, the knowledge propagation for each class is more accurate.

We randomly select objects to obtain label information in our experiments. Similar to [21], we observe that if we choose some representative (or highly ranked) objects (e.g., famous authors) to label, the classification performance will be generally slightly better than when using labels of low quality (e.g., authors closely related to multiple fields, or with few publications). However, the difference is not significant. In other words, the initial choice of labeled data does not drastically affect the quality of ranking and classification. This is because our graph-based ranking model in Equation (3) is a summation of two terms, where the first depends on the initial ranking, and the second depends on the network structure which ensures the smoothness of the learner. Even if the quality of the initial ranking distribution is not very high, RankClass can still generate a reasonable ranking distribution and label predictions. This is because RankClass exploits the relationships among objects in the network, iteratively propagating information throughout the network (see the first term of Equation (3)). Therefore, our algorithm is theoretically robust when working with random labels.

5.3 Convergence Study

Since our proposed RankClass algorithm iteratively adjusts the network structure to facilitate within-class ranking, we further explore the changes of the link weights within the network. According to ground truth for each class k, all of the links connected to object type $\mathcal{X}_i$ can be divided into two groups: one contains the links that connect to at least one object of class k (denoted as G_{in}), while the other group does not involve any objects of class k (denoted as G_{out}). Links in G_{in} compose the sub-network corresponding to class k, while links in G_{out} form the sub-network excluding class k. In Figure 1, we show the average weight of the links in G_{in} and G_{out} (averaged over the four classes) connected to authors, conferences and papers, along with the number of iterations, when $(0.5\%, 0.5\%)$ authors and papers are labeled. The link weight changes of G_{in} and G_{out} for each individual class are very similar to Figure 1, and are omitted due to space limitation.

From Figure 1, it can be observed that the average link weight of G_{in} and G_{out} are the same at first, and then decrease at various rates over iterations. During the first several iterations, the ranking scores have not propagated very far throughout the network and are close to 0 in value for many objects. Therefore, the weight of many links decreases almost exponentially as a result of multiplying by $r(t)$. Then the ranking scores of objects within each class k gradually increase, making the average link weight in G_{in} decrease more slowly than that in G_{out}. Within a few iterations, the average link weights in G_{in} and G_{out} converge to relatively stable values. There is also a clear gap between the average link weights of G_{in} and G_{out} (the former being much larger than the latter). Thus, the sub-network corresponding to each class k is well-separated from the rest of the network, and the within-class ranking can be accurately performed within the sub-network, rather than the global network. When the network structure stabilizes, the graph-based ranking scheme can be proven to converge, following a similar analysis to [26, 11].

5.4 Case Study

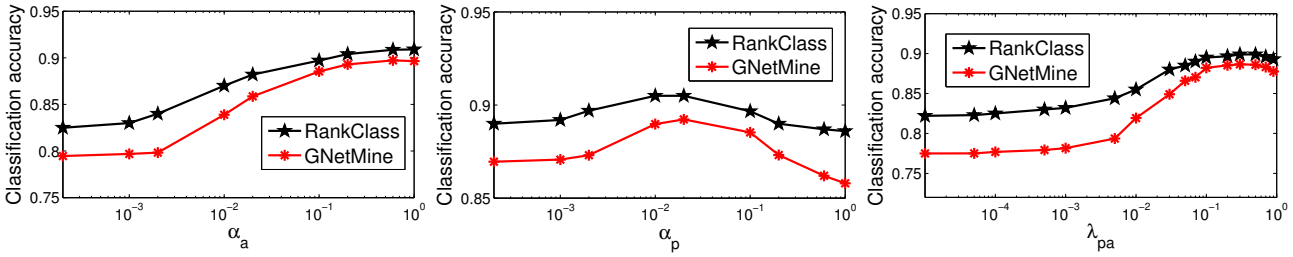
In this section, we present a simple case study by listing the top ranked data objects within each class. Recall that GNetMine performs the second best in the classification accuracy, and can generate a confidence score for each object related to each class [11]. Thus, we can also rank data objects according to the confidence scores related to each class as the within-class ranking. In Tables 6 and 7, we show the comparison of the ranking lists of confer-

Table 6: Top-5 conferences related to each research area generated by different algorithms

RankClass				GNetMine			
Database	Data Mining	AI	IR	Database	Data Mining	AI	IR
VLDB	KDD	IJCAI	SIGIR	VLDB	SDM	IJCAI	SIGIR
SIGMOD	SDM	AAAI	ECIR	ICDE	KDD	AAAI	ECIR
ICDE	ICDM	ICML	CIKM	SIGMOD	ICDM	ICML	CIKM
PODS	PKDD	CVPR	WWW	PODS	PAKDD	CVPR	IJCAI
EDBT	PAKDD	ECML	WSDM	CIKM	PKDD	ECML	CVPR

Table 7: Top-5 terms related to each research area generated by different algorithms

RankClass				GNetMine			
Database	Data Mining	AI	IR	Database	Data Mining	AI	IR
data	mining	learning	retrieval	interlocking	rare	failing	helps
database	data	knowledge	information	deindexing	extreme	interleaved	specificity
query	clustering	reasoning	search	seed	scan	cognition	sponsored
system	frequent	logic	web	bitemporal	mining	literals	relevance
xml	classification	model	text	debugging	associations	configuration	information



(a) Varying α_a (b) Varying α_p (c) Varying λ_{pa}

Figure 2: Model Selection when (0.5%, 0.5%) of authors and papers are labeled

ences and terms generated by RankClass and GNetMine, respectively, with $(0.5\%, 0.5\%)$ authors and papers labeled.

From comparing the ranking lists of the two types of objects, we can see that RankClass generates more meaningful ranking results than GNetMine. There is a high degree of consensus between the ranking list of conferences generated by RankClass and the top conferences in each research area. Similarly, the highly ranked terms generated by RankClass are in high agreement with the most representative keywords in each field. The reason why GNetMine fails to generate meaningful ranking lists is that the portions of labeled authors and papers are too limited to capture the distribution of the confidence score with regard to each class. In contrast, RankClass boosts the ranking performance by iteratively obtaining the clean sub-network corresponding to each class, which favors the within-class ranking.

5.5 Model Selection

In the graph-based ranking scheme in Equation (2), the α_i's and λ_{ij}'s are essential parameters which control the relative importance of different types of information. In the previous experiments, we empirically set α_i's as 0.1, and λ_{ij}'s as 0.2, $\forall i, j \in \{1, \ldots, m\}$. In this subsection, we study the impact of parameters on the performance of RankClass. Since only several authors and papers are labeled, the α_i associated with authors (denoted by α_a) and papers (denoted by α_p), as well as the λ_{ij} associated with the *author-paper* relationship (denoted by λ_{pa}) are empirically more important than other parameters. Therefore we fix all other parameters and let α_a, α_p and λ_{pa} vary. As GNetMine performs the second best in the classification task, and has been proven to be robust over a large range of parameters [11], we only compare RankClass with GNetMine in this experiment. Note that we also change the corresponding parameters α_a, α_p and λ_{pa} in GNetMine. We show the average classification accuracy on three types of objects (author, paper, conference) as a function of the parameters in Figure 2, with $(a\%, p\%) = (0.5\%, 0.5\%)$ authors and papers labeled.

We observe that over a large range of parameters, RankClass achieves better performance than GNetMine [11]. Since the graph-based ranking scheme in RankClass has a knowledge propagation framework similar to that of GNetMine, the changes in accuracy of the two algorithms over different parameters trend in a similar fashion. However, RankClass generates more accurate and robust results by employing the ranking results to iteratively extract the sub-network corresponding to each class. We therefore conclude that the performance of the RankClass algorithm is generally not very sensitive to the setting of its parameters.

5.6 Time Complexity Study

In this section, we vary the size of the database by randomly selecting connected sub-networks from the original network, and then test the running time of our algorithm. The size of the database is measured by the number of nodes in the network. As can be seen in Figure 3, the time complexity of our method is generally linear with respect to the size of the database, which is consistent with our analysis in Section 4.5.

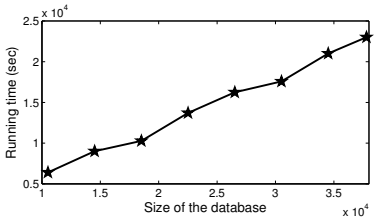


Figure 3: Running time w.r.t. database size

6. CONCLUSIONS

In this paper, we investigate a new problem of classifying data objects in a heterogeneous information network, while simultaneously ranking each object according to its importance within each class, in order to provide informative class summaries. A novel ranking-based classification algorithm called RankClass is proposed to iteratively solve this problem. During each iteration, we calculate the ranking distribution over the nodes of the network by authority propagation. The ranking results are then used to modify the network structure to allow the ranking model to improve the within-class ranking. Thus, we gradually extract the sub-network corresponding to each specific class from the global network. Finally, we calculate the posterior probability of each object belonging to each class to determine each object's optimal class membership.

In the future, we plan to more thoroughly analyze the theoretical foundation behind the ranking-based classification framework to further justify its ability to enhance both classification and ranking. It would also be interesting to study integrating ranking and classification when working with non-networked data.

7. ACKNOWLEDGEMENTS

The work was supported in part by U.S. National Science Foundation grants IIS-0905215, IIS-1017362, and the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053 (NS-CTA). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

8. REFERENCES

[1] W. Bian and D. Tao. Manifold regularization for sir with rate root-n convergence. In *NIPS 22*, pages 117–125. 2009.

[2] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998.

[3] B. Cao, N. N. Liu, and Q. Yang. Transfer learning for collective link prediction in multiple heterogeneous domains. In *ICML*, pages 159–166, 2010.

[4] D. R. Cutting, J. O. Pedersen, D. R. Karger, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *SIGIR*, pages 318–329, 1992.

[5] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Computer and System Sciences*, 55:119–139, 1997.

[6] B. Gao, T.-Y. Liu, Z. Ma, T. Wang, and H. Li. A general markov framework for page importance computation. In *CIKM*, pages 1835–1838, 2009.

[7] J. Gao, F. Liang, W. Fan, Y. Sun, and J. Han. Graph-based consensus maximization among multiple supervised and unsupervised models. In *NIPS 22*, pages 585–593, 2009.

[8] A. Guillory and J. Bilmes. Label selection on graphs. In *NIPS 22*, 2009.

[9] S. Hanneke and E. P. Xing. Network completion and survey sampling. In *AISTAT*, 2009.

[10] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction (2nd ed.)*. Springer-Verlag, 2009.

[11] M. Ji, Y. Sun, M. Danilevsky, J. Han, and J. Gao. Graph regularized transductive classification on heterogeneous information networks. In *ECML/PKDD (1)*, pages 570–586, 2010.

[12] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[13] B. Long, Z. M. Zhang, X. Wu, and P. S. Yu. Spectral clustering for multi-type relational data. In *ICML*, pages 585–592, 2006.

[14] Q. Lu and L. Getoor. Link-based classification. In *ICML*, 2003.

[15] S. A. Macskassy and F. Provost. A simple relational classifier. In *MRDM at KDD*, pages 64–76, 2003.

[16] S. A. Macskassy and F. Provost. Classification in networked data: A toolkit and a univariate case study. *J. Mach. Learn. Res.*, 8:935–983, 2007.

[17] J. Neville and D. Jensen. Relational dependency networks. *J. Mach. Learn. Res.*, 8:653–692, 2007.

[18] Z. Nie, Y. Zhang, J.-R. Wen, and W.-Y. Ma. Object-level ranking: bringing order to web objects. In *WWW*, pages 567–574, 2005.

[19] P. Sen and L. Getoor. Link-based classification. Technical Report CS-TR-4858, University of Maryland, February 2007.

[20] L. Sun, S. Ji, and J. Ye. Hypergraph spectral learning for multi-label classification. In *KDD*, pages 668–676, 2008.

[21] Y. Sun, Y. Yu, and J. Han. Ranking-based clustering of heterogeneous information networks with star network schema. In *KDD*, pages 797–806, 2009.

[22] T. Yang, R. Jin, Y. Chi, and S. Zhu. Combining link and content for community detection: a discriminative approach. In *KDD*, pages 927–936, 2009.

[23] O. Zamir and O. Etzioni. Grouper: A dynamic clustering interface to web search results. *Computer Networks*, 31(11-16):1361–1374, 1999.

[24] B. Zhang, H. Li, Y. Liu, L. Ji, W. Xi, W. Fan, Z. Chen, and W.-Y. Ma. Improving web search results using affinity graph. In *SIGIR*, pages 504–511, 2005.

[25] Y. Zhang and Z.-H. Zhou. Non-metric label propagation. In *IJCAI*, pages 1357–1362, 2009.

[26] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NIPS 16*, 2003.

[27] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *NIPS 16*, 2003.