

Classifying Networked Entities with Modularity Kernels

Dell Zhang
SCSIS

Birkbeck, University of London
London WC1E 7HX, UK
dell.z@ieee.org

Robert Mao
Microsoft Corp.

EPDC5/2352, South County Business Park
Leopardstown, Dublin, Ireland
robmao@microsoft.com

ABSTRACT

Statistical machine learning techniques for data classification usually assume that all entities are i.i.d. (independent and identically distributed). However, real-world entities often interconnect with each other through explicit or implicit relationships to form a complex network. Although some graph-based classification methods have emerged in recent years, they are not really suitable for complex networks as they do not take the degree distribution of network into consideration. In this paper, we propose a new technique, Modularity Kernel, that can effectively exploit the latent community structure of networked entities for their classification. A number of experiments on hypertext datasets show that our proposed approach leads to excellent classification performance in comparison with the state-of-the-art methods.

Categories and Subject Descriptors

H.2.8 [Database Management]: Database Applications—*data mining*; H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval; I.2.6 [Artificial Intelligence]: Learning; I.5.2 [Pattern Recognition]: Design Methodology—*classifier design and evaluation*

General Terms

Algorithms, Performance, Experimentation

Keywords

Graph Mining, Community Discovery, Semi-Supervised Learning, Kernel Methods, Hypertext Classification.

1. INTRODUCTION

We are in a connected age: real-world entities often interconnect with each other through explicit or implicit relationships to form a *complex network* [34], such as social networks, information networks, technological networks and

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CIKM'08, October 26–30, 2008, Napa Valley, California, USA.
Copyright 2008 ACM 978-1-59593-991-3/08/10 ...\$5.00.

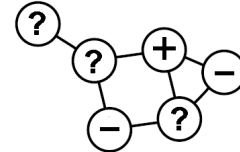


Figure 1: Classifying networked entities.

biological networks. One of the “10 challenging problems in data mining research”¹ [47] highlighted in ICDM-2005 is “data mining in a network setting”.

In this paper, we address the problem of classifying networked entities (or within-network classification). Given a set of both labelled and unlabelled entities that are interconnected with each other, our goal is to assign class labels to the unlabelled entities, as shown in Figure 1. For example, in the application of web spam filtering², the web pages are connected via hyperlinks in a complex network where some web pages are known to be ‘spam’ while some web pages are known to be ‘non-spam’, and the task is to classify the other web pages into the ‘spam’ or ‘non-spam’ class.

Statistical *machine learning* [5] techniques for data classification usually assume that all entities are i.i.d. (independent and identically distributed). Although such techniques could be applied straightforwardly to the problem of classifying networked entities, we anticipate that a better classification performance would be achieved by taking advantage of the additional link information because entities tend to connect to other members of their own class.

Most complex networks in the real-world are large but sparse, and they are found to have some common statistical properties such as the small-world phenomenon and the scale-free (power-law) degree distribution [34, 33, 9, 16], which distinguish them with simple (regular or random) graphs. Due to the sparsity of network, two entities in the same class may appear to be dissimilar on surface when they are not directly connected, but such a pair of entities are likely to be connected indirectly through some paths with a number of intermediate entities. Therefore it is usually not promising to use links directly as additional features. In other words, it is important to exploit not only local link information but also global structure information for effective classification of networked entities. We think the underlying *community structure* [35] of network contains valuable clues

¹<http://www.cs.uvm.edu/~icdm/10Problems/>

²<http://webspam.lip6.fr/>

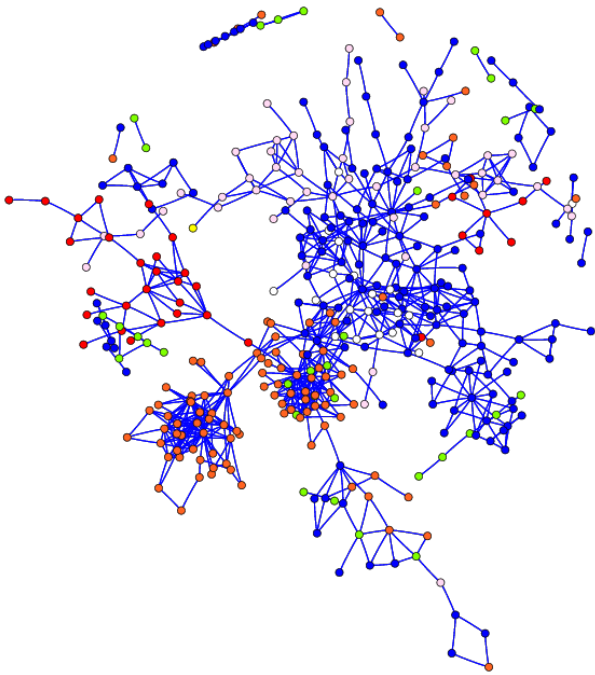


Figure 2: The community structure of the Cora-HA network.

about the right classification of its entities. For example, the community structure of the Cora-HA network (see Section 5) is visualised in Figure 2. It is clear that the nodes in the same class (with the same colour) tend to group together in a community.

Some graph-based classification methods have emerged in recent years (see Section 2). However, they are not really suitable for complex networks because they do not take the degree distribution of network into consideration. For example, the degree distribution of the above Cora-HA network roughly obeys the power law (as shown in Figure 3), which is very different with that of a simple graph.

In this paper, we propose a new technique, Modularity Kernel, that can effectively exploit the latent *community structure* [35] of networked entities for their classification. A number of experiments on hypertext datasets show that our proposed approach leads to excellent classification performance in comparison with the state-of-the-art methods.

The rest of this paper is organised as follows. In Section 2, we review the related work. In Section 3, we formally define the problem of classifying networked entities. In Section 4, we present our approach in details. In Section 5, we show the experimental results. In Section 6, we make conclusions.

2. RELATED WORK

Recently many approaches to the problem of classifying networked entities (such as web pages) have been proposed from different perspectives.

One possibility is to do network-aware *feature engineering* or *dimensionality reduction* first and then employ standard classification methods. Previous research studies have shown that using links directly as features does not work well in practice [12, 53]. Another simple method is to expand each entity’s feature vector by incorporating the fea-

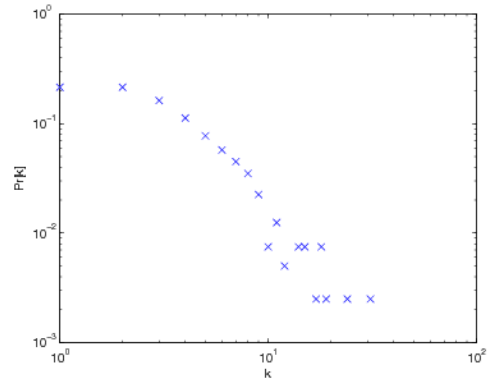


Figure 3: The degree distribution ($\Pr[k] \sim k$) of the Cora-HA network.

tures of its neighbours (directly-linked entities) [48], but this method seems to have difficulties with parameter tuning and often does not provide a robust solution [12, 1]. Inspired by the success of PageRank [10] and HITS [30] in web search and mining, people have tried to apply *link analysis* techniques to extract network structure features. Gyöngyi et al. propose a technique similar to Personalised PageRank for large-scale web page classification [24]. Cohn and Hofmann propose to address hypertext classification by constructing a latent space of both content and link information [18] where probabilistic LSI (PLSI) is used for content analysis [26] and probabilistic HITS (PHITS) is used for link analysis [17]. Joachims et al. propose to address hypertext classification in the kernel methods [38] framework by combining bag-of-words kernels and bag-of-links kernels [29]. Recently Zhu et al. propose to use the technique Matrix Factorization (MF) to find latent factors from both content matrix and link matrix for the task of hypertext classification, and they also extend it to Supervised Matrix Factorization (SupMF) by taking label information into account as well [53].

An alternative way is to use *relational learning* [27] methods such as Markov Random Fields [12, 11, 1], Inductive Logic Programming [43, 19], Probabilistic Relational Models [21], Relational Markov Networks [45], Dependency Networks [32] and Lazy Associative Classification [46]. Some comprehensive studies in this area can be found in [27, 31]. However, such relational learning methods do not necessarily converge, or only converge to a local optimal solution.

Yet another (probably more principled) way is to use *semi-supervised learning* methods [13, 54] based on *graph partitioning* [15, 40], such as MinCut (*st*-Cut) [6, 7], *t*-step Markov Random Walk [44], Gaussian Random Fields and Harmonic Functions [55], Spectral Graph Transducer [28], Tikhonov Regularisation [3], Manifold Regularisation [42, 41, 4], Cluster Kernel [14], Local and Global Consistency [50], Directed Graph Regularisation (DGR) [52, 51] and Laplacian Kernel [25, 2]. However, while these methods work well for simple graphs (like *k*-nearest-neighbours graphs), they are not really suitable for complex networks as they totally ignore the degree distribution of network.

3. PROBLEM

We define the problem of classifying networked entities formally as follows.

Given a complex network (graph) G that consists of n entities (nodes) and m links (edges), we can describe the network structure using its $n \times n$ adjacency matrix \mathbf{A} with elements $A_{ij} \geq 0$ representing the number or weight of edges between node \mathbf{x}_i and node \mathbf{x}_j . Since real-world complex networks are usually sparse, most elements in \mathbf{A} should be 0. In this paper we focus on undirected networks, so \mathbf{A} is symmetric. The degree of a node \mathbf{x}_i , i.e., the number of edges connected to a node \mathbf{x}_i , is given by $k_i = \sum_j A_{ij}$.

Without loss of generality, suppose that there are two classes of entities in the network: the entities in one class are labelled with +1 and the entities in the other class are labelled with -1. In the given set of networked entities $X = \{\mathbf{x}_i\}_{i=1}^n$, there are l entities $X_l := \{\mathbf{x}_i\}_{i=1}^l$ for which the labels $Y_l := \{y_i\}_{i=1}^l$ are provided, and u entities $X_u := \{\mathbf{x}_j\}_{j=l+1}^{l+u}$ for which the labels are absent. Obviously $X = X_l \cup X_u$ and $n = l + u$. Our goal is to learn a classification function f so that the class of any networked entity \mathbf{x} can be accurately predicted by the sign of $f(\mathbf{x})$. This is actually *semi-supervised learning*, i.e., learning from both labelled and unlabelled data [13, 54].

4. APPROACH

4.1 Learning Framework

Let us consider the problem of classifying networked entities in the framework of *kernel methods* [38, 39, 49]. A prominent advantage of kernel methods is that they typically lead to a *convex optimisation* problem so the global optimal solution can be computed efficiently.

For a Mercer kernel $K : X \times X \rightarrow \mathbb{R}$, there is an associated Reproducing Kernel Hilbert Space (RKHS) \mathcal{H}_K of functions $X \rightarrow \mathbb{R}$ with the corresponding norm $\|\cdot\|_K$. Given a set of labelled examples X_l as well as a set of unlabelled examples X_u , typical kernel methods for semi-supervised learning estimate the classification function to be

$$f^* = \arg \min_{f \in \mathcal{H}_K} \frac{1}{l} \sum_{i=1}^l V(x_i, y_i, f) + C \|f\|_K^2,$$

where the first term V is a loss function defined on X_l , the second term $\|f\|_K^2$ is a regulariser defined on $X_l \cup X_u$ and the parameter C controls its relative weight. The regulariser $\|f\|_K^2$ imposes smoothness conditions on f to avoid overfitting.

Different choices of the loss function V in the above optimisation problem lead to different learning algorithms. For example, substituting the logistic loss $\ln(1 + \exp(-y_i f(\mathbf{x}_i)))$ for V we get Regularised Logistic Regression (RLR); substituting the hinge loss $(1 - y_i f(\mathbf{x}_i))_+ = \max(0, 1 - y_i f(\mathbf{x}_i))$ for V we get Support Vector Machine (SVM).

A straightforward extension of the classic Representer Theorem [38] states that the solution to the above optimisation problem exists in \mathcal{H}_K and can be written as an expansion in terms of both labelled and unlabelled examples:

$$f^*(\mathbf{x}) = \sum_{i=1}^{l+u} \alpha_i K(\mathbf{x}_i, \mathbf{x}).$$

This can be proved by a variation of the simple orthogonality argument [38], as shown in [4]. Therefore the above learning problem can be reduced to optimising over the finite dimensional space of coefficients $\{\alpha_i\}_{i=1}^{l+u}$, which is the algorithmic basis of kernel methods.

4.2 Community Discovery

In real-world complex networks, entities (nodes) tend to group into communities, with a high density of links (edges) within communities and a low density of links (edges) between them [35]. The latent *community structure* of a complex network must contain valuable clues about the right classification of entities (nodes) in the network, because entities are likely to connect to other members of their own class. A “good” classification function f for networked entities should align well not only with the labelled data but also with the community structure — entities in the same community should have a high probability to be classified into the same class.

In the ideal situation, each class of entities in the complex network would group into a separate community. So if the value of f is restricted to be either +1 or -1, then $\mathbf{f} = (f(\mathbf{x}_1), \dots, f(\mathbf{x}_n))^T$ can be regarded as the binary indication vector for a division of the network into communities, and $\mathbf{f}^T \mathbf{f} = n$. Furthermore, let g_i denote the group to which vertex \mathbf{x}_i belongs. The function $\delta(g_i, g_j) = 1$ if $g_i = g_j$ and $\delta(g_i, g_j) = 0$ otherwise. It is easy to see that $\delta(g_i, g_j) = \frac{1}{2}(1 + f(\mathbf{x}_i)f(\mathbf{x}_j))$.

4.2.1 Minimising the Cut-Size

Most existing graph-based classification methods are rooted in *graph partitioning* [40] that divides the network into groups by minimising the *cut-size*, i.e., the number of edges running between different groups of nodes:

$$S = \frac{1}{2} \sum_{i,j} A_{ij} (1 - \delta(g_i, g_j)).$$

We can rewrite the cut-size for the division of network \mathbf{f} in matrix form as

$$S = \frac{1}{4} \mathbf{f}^T \mathbf{L} \mathbf{f},$$

where \mathbf{L} is the *Laplacian matrix* [15] defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

with $\mathbf{D} = \text{diag}(k_1, \dots, k_n)$.

It is well-known that all eigenvalues of \mathbf{L} are non-negative, and the graph G has z connected components if and only if \mathbf{L} has z zero eigenvalues with corresponding eigenvectors being piece-wise constant on the connected components [15]. Without loss of generality, assume that the eigenvalues of \mathbf{L} in increasing order are $0 = \lambda_1 = \dots = \lambda_z < \lambda_{z+1} \leq \dots \leq \lambda_n$ and the corresponding eigenvectors of \mathbf{L} are $\mathbf{u}_1, \dots, \mathbf{u}_z, \mathbf{u}_{z+1}, \dots, \mathbf{u}_n$.

If we allow the the elements of \mathbf{f} to take any real value but just keep the constraint $\mathbf{f}^T \mathbf{f} = n$, a *non-trivial* division of network that minimises the cut-size S is given by $\mathbf{f} = \mathbf{u}_{z+1}$, the eigenvector of \mathbf{L} corresponding to the smallest non-zero eigenvalue [15]. The sign of \mathbf{u}_{z+1} 's i -th element indicates the class to which \mathbf{x}_i belongs to and the value of \mathbf{u}_{z+1} 's i -th element indicates the strength of membership.

People have tried to use the cut-size S as the regulariser for learning and found that it is equivalent to taking the pseudo-inverse of Laplacian matrix, \mathbf{L}^+ , as the kernel matrix, which is called Laplacian Kernel (LapKer) [25, 2]. It has been shown that the Laplacian Kernel is interestingly connected to the resistance distance (the total resistance between two nodes) and the commute time (the average length of a random walk between two nodes) over the graph.

The *normalised Laplacian* [15] that has many attractive theoretical properties

$$\tilde{\mathbf{L}} = \mathbf{D}^{-1/2} \mathbf{L} \mathbf{D}^{-1/2}$$

can also be used in the above formulae. Since using $\tilde{\mathbf{L}}^+$ instead of \mathbf{L}^+ as Laplacian Kernel consistently provides better classification performance, the reported Laplacian Kernel experimental results (in Section 5) are all based on $\tilde{\mathbf{L}}^+$.

Despite its success on simple graphs (such as k -nearest-neighbours graphs), Laplacian based graph partitioning is poor in detecting natural communities in real-world complex networks, because the degree distribution of network has been totally ignored. The fundamental problem of using this technique for community discovery is that cut sizes are not really the right thing to optimise since they don't accurately reflect our intuitive concept of network communities. A good division of a network into communities is not merely one in which the number of edges running across communities is small. Rather, it is one in which the number of edges across communities is *smaller than expected*. It has been reported that Laplacian based graph partitioning often fails to find the right division of a complex network [37, 36]. Consequently the effectiveness of semi-supervised learning methods based on graph partitioning for classifying networked entities would be limited.

4.2.2 Maximising the Modularity

One proven-effective approach to community discovery is maximising the quality function known as *modularity* [37, 36] over the possible divisions of a network:

$$Q = \frac{1}{2m} \sum_{ij} [A_{ij} - P_{ij}] \delta(g_i, g_j),$$

where $P_{ij} = (k_i k_j) / (2m)$. In fact, P_{ij} is the *expected* number of edges between node \mathbf{x}_i and node \mathbf{x}_j in the 'null model' — a *random* graph with the same degree distribution as the given network. Optimising modularity reflects our intuition that the number of edges within communities should be higher than expected by chance. Only if the number of within-community edges $\frac{1}{2} \sum_{ij} A_{ij} \delta(g_i, g_j)$ is significantly higher than it would be expected purely by chance $\frac{1}{2} \sum_{ij} P_{ij} \delta(g_i, g_j)$ can we justifiably claim to have found significant community structure. Maximising modularity has been shown to produce excellent community discovery results in practice [20, 23].

We can rewrite the modularity for the division of network \mathbf{f} in matrix form as

$$Q = \frac{1}{4m} \mathbf{f}^T \mathbf{M} \mathbf{f},$$

where \mathbf{M} is the *modularity matrix*³ [36] defined as

$$\mathbf{M} = \mathbf{A} - \mathbf{P}.$$

Unlike the graph Laplacian, the modularity matrix \mathbf{M} is not guaranteed to be positive semi-definite, i.e., it may have negative eigenvalues. Without loss of generality, assume that the eigenvalues of \mathbf{M} in decreasing order are $\lambda_1 \geq \dots \geq \lambda_n$ and the corresponding eigenvectors of \mathbf{M} are $\mathbf{u}_1, \dots, \mathbf{u}_n$.

³We have extended the original definition of modularity matrix [36] to weighted networks here.

If we allow the elements of \mathbf{f} to take any real value but just keep the constraint $\mathbf{f}^T \mathbf{f} = n$, the optimal division of network that maximises the modularity Q is given by $\mathbf{f} = \mathbf{u}_1$, the eigenvector of \mathbf{M} corresponding to the largest eigenvalue [36]. The sign of \mathbf{u}_1 's i -th element indicates the class to which \mathbf{x}_i belongs to and the value of \mathbf{u}_1 's i -th element indicates the strength of membership.

4.3 Modularity Kernel

Motivated by modularity based community discovery, we propose to use the following matrix as the kernel matrix:

$$\widehat{\mathbf{M}} = \sum_{k=1}^p \lambda_k \mathbf{u}_k \mathbf{u}_k^T,$$

where $\lambda_1 \geq \dots \geq \lambda_p > 0$ are the p positive eigenvalues of \mathbf{M} and $\mathbf{u}_1, \dots, \mathbf{u}_p$ are the p corresponding eigenvectors. Since the original modularity matrix \mathbf{M} is not guaranteed to be positive definite, it could not be used straightforwardly as the kernel matrix otherwise the generated kernel function would be invalid and the resulted optimisation problem would no longer be convex. According to linear algebra [22], $\widehat{\mathbf{M}}$ is the positive definite matrix that best approximates the modularity matrix \mathbf{M} (in terms of $\|\widehat{\mathbf{M}} - \mathbf{M}\|_F$), therefore we use it instead⁴ and name this technique Modularity Kernel (ModKer).

The Modularity Kernel could be justified as follows. Let $\mathcal{H}(G)$ be the linear space of real-valued functions defined on the graph, i.e., an n -dimensional vector space whose elements are the real-valued vector $\mathbf{g} = (g_1, \dots, g_n)^T$. On $\mathcal{H}(G)$ we introduce the inner-product

$$\langle \mathbf{f}, \mathbf{g} \rangle = \mathbf{f}^T \widehat{\mathbf{M}}^{-1} \mathbf{g}.$$

This inner-product function is well-defined since

$$\widehat{\mathbf{M}}^{-1} = \sum_{k=1}^p \frac{1}{\lambda_k} \mathbf{u}_k \mathbf{u}_k^T$$

is symmetric and positive definite. Moreover, the function $\|g\| = \sqrt{\langle \mathbf{g}, \mathbf{g} \rangle}$, $\mathbf{g} \in \mathcal{H}(G)$ is a norm that measures the function smoothness or complexity. With a little derivation, it is easy to see that minimising the above defined norm of function f leads to $\mathbf{f} = \mathbf{u}_1$ which gives the optimal division of network into communities (according to the analysis in the above section). Therefore it is reasonable to use $\|f\|^2 = \langle \mathbf{f}, \mathbf{f} \rangle = \mathbf{f}^T \widehat{\mathbf{M}}^{-1} \mathbf{f}$ as the regulariser in the kernel methods learning framework. The reproducing kernel \mathbf{K} of $\mathcal{H}(G)$ should be an $n \times n$ matrix such that for every $\mathbf{g} \in \mathcal{H}(G)$ the reproducing kernel property $g_i = \langle \mathbf{K}_i, \mathbf{g} \rangle$ holds, where \mathbf{K}_i is the i -th column of \mathbf{K} . In fact, $\mathbf{K} = \widehat{\mathbf{M}}$, because if $\mathbf{g} \in \mathcal{H}(G)$ then $\widehat{\mathbf{M}}^{-1} \widehat{\mathbf{M}} \mathbf{g} = \mathbf{g}$, which implies that $g_i = \mathbf{e}_i^T \widehat{\mathbf{M}} \widehat{\mathbf{M}}^{-1} \mathbf{g} = \mathbf{K}_i^T \widehat{\mathbf{M}}^{-1} \mathbf{g} = \langle \mathbf{K}_i, \mathbf{g} \rangle$ where \mathbf{e}_i is the i -th coordinate vector, i.e., the reproducing kernel property holds. Hence $\mathbf{K} = \widehat{\mathbf{M}}$ is the reproducing kernel of $\mathcal{H}(G)$.

Modularity Kernel can effectively exploit the latent *community structure* of networked entities for their classification. It is particularly suitable for complex networks because it takes the degree distribution into account.

⁴We have found that in practice using only a small number (about 30–50) of largest positive eigenvalues and eigenvectors would achieve almost as good classification performance as using all p positive eigenvalues and eigenvectors.

4.4 Combining Content and Link

The proposed Modularity Kernel technique can be used straightforwardly for classifying networked entities. However, it should be beneficial to exploit both entity content information and entity link information [53]. We tried the following three linear combination methods with a parameter $0 \leq \mu \leq 1$.

- **Regulariser Combination**

$$\begin{aligned} \|f\|^2 &= (1 - \mu)\|f\|_{content}^2 + \mu\|f\|_{link}^2 \\ &= (1 - \mu)\|f\|_{content}^2 + \mu\mathbf{f}^T\widehat{\mathbf{M}}^{-1}\mathbf{f}. \end{aligned}$$

The content-based regulariser $\|f\|_{content}$ is the standard regulariser defined on the content feature space. In our hypertext classification experiments we use the l_2 norm of the original bag-of-words classification function.

- **Kernel Combination**

$$\begin{aligned} \mathbf{K} &= (1 - \mu)\mathbf{K}_{content} + \mu\mathbf{K}_{link} \\ &= (1 - \mu)\mathbf{K}_{content} + \mu\widehat{\mathbf{M}}. \end{aligned}$$

The content-based kernel $\mathbf{K}_{content}$ is the standard kernel defined on the content feature space. In our hypertext classification experiments we use the linear kernel, i.e., the inner-product of the original bag-of-words vectors, because it has been shown to consistently achieve best performance for various text classification tasks.

- **Graph Combination**

$$\mathbf{A} = (1 - \mu)\mathbf{A}_{content} + \mu\mathbf{A}_{link}.$$

We construct a hybrid graph which is actually the linear combination of two graphs — one the original network induced by the links among entities, the other derived from entity content information — and then use Modularity Kernel computed on the hybrid network. When deriving the content-based network, we simply connect each pair of entities with an edge that is weighted by their content (cosine) similarity. This combining method is actually in spirit of *co-training* [8, 28]

To distinguish the usage of Modularity Kernel based on only entity link information and that based on both entity content and entity link information, we denote the former **ModKer (link)** and the latter **ModKer (content+link)**.

5. EXPERIMENTS

5.1 Datasets

We conduct our experiments on two collections of real-world datasets: WebKB⁵ and Cora⁶.

The WebKB datasets consist of about 6,000 web pages from the computer science departments of four universities: *Cornell*, *Texas*, *Washington* and *Wisconsin*. The web pages are classified into categories such as ‘course’, ‘faculty’ and ‘student’. For each dataset, we use the *co-citation graph* derived from the original directed hyperlinks (citations) as

⁵<http://www.cs.cmu.edu/~webkb/>

⁶<http://www.cs.umass.edu/~mccallum/code-data.html>

the link-based entity network for our algorithms, i.e., two pages (nodes) are connected by an edge if there is a third page linking to or being linked to both of them, and multiple edges are allowed between one pair of nodes. This is because for web data communities formed by co-citations are usually more reliable than by hyperlinks, which has been reported in past research studies [49] and also confirmed by our experiments. The characteristics of the WebKB datasets are shown in Table 1.

The Cora datasets consist of the abstracts and references of about 34,000 computer science papers. In our experiments, we use only the papers in four research areas, *Data Structure (DS)*, *Hardware and Architecture (HA)*, *Machine Learning (ML)* and *Programming Language (PL)*, and we discard the papers without reference to other papers in the same area. The papers are classified according to their sub-fields in the research area. For each dataset, we use the *undirected graph* derived from the original directed references as the link-based entity network for our algorithms, i.e., two papers (nodes) are connected by an edge if one of them cites the other or vice versa, and multiple edges are allowed between one pair of nodes. The characteristics of the Cora datasets are shown in Table 2.

5.2 Settings

We perform our hypertext classification experiments on each of the above datasets using Regularised Logistic Regression (RLR) and Support Vector Machine (SVM) [38, 39] with the proposed Modularity Kernel⁷.

We measure classification performance by 5-fold cross-validation accuracy (mean \pm std %), i.e., the dataset is randomly split into five equal-size folds and the classification experiment is repeated for five times, each time with one fold for test and four other folds for training. For experiments with our proposed Modularity Kernel technique, we do not tune the RLR or SVM parameters but simply take their default values. For experiments using other methods, the parameters are tuned through cross-validation on the training-folds data, as in [53].

5.3 Results

5.3.1 Comparison of Learning Algorithms

We first compare the classification performance of RLR and SVM with ModKer (link). The experimental results on the WebKB datasets and the Cora datasets are shown in Table 3 and 4 respectively. It is clear that SVM with ModKer outperforms RLR with ModKer on all the datasets. Therefore SVM is used in all the following ModKer experiments.

5.3.2 Comparison of Combination Methods

We then compare the effectiveness of three combination methods (see Section 4.4) using SVM with ModKer (content+link). The experimental results on the WebKB datasets and the Cora datasets are shown in Table 5 and 6 respectively, where the combination parameter μ is simply set to its default value 0.5. It is clear that Graph Combination outperforms Regulariser Combination and Kernel Combination on all the datasets. Therefore Graph Combination is used in all the following ModKer experiments.

⁷Each kernel matrix is normalised [38, 39] in our experiments.

Table 1: Characteristics of the WebKB datasets.

dataset	Cornell	Texas	Washington	Wisconsin
the number of classes	7	7	7	6
the number of entities (nodes)	827	814	1166	1210
the number of terms	4134	4029	4165	4189
the number of edges	49560	59620	80564	91244
the minimum degree of a node	0	0	0	0
the maximum degree of a node	478	533	912	843
the median degree of a node	14	17	15	21
the average degree of a node	59.93	73.24	69.09	75.41

Table 2: Characteristics of the Cora datasets.

dataset	DS	HA	ML	PL
the number of classes	9	7	7	9
the number of entities (nodes)	751	400	1617	1575
the number of terms	6234	3989	8329	7949
the number of edges	2566	1586	8092	9836
the minimum degree of a node	1	1	1	1
the maximum degree of a node	32	31	55	76
the median degree of a node	2	3	4	4
the average degree of a node	3.42	3.97	5.00	6.25

Table 3: Comparison of learning algorithms on the WebKB datasets.

algorithm	Cornell	Texas	Washington	Wisconsin
Regularised Logistic Regression	89.60±2.40	93.98±1.64	91.85±1.68	87.60±1.55
Support Vector Machine	92.50±1.64	96.44±1.01	92.54±1.15	91.07±2.46

Table 4: Comparison of learning algorithms on the Cora datasets.

algorithm	DS	HA	ML	PL
Regularised Logistic Regression	72.31±2.65	77.25±4.45	78.35±2.73	69.33±2.17
Support Vector Machine	76.42±4.36	87.50±2.65	82.93±1.39	77.90±2.12

Table 5: Comparison of combination methods on the WebKB datasets.

combination	Cornell	Texas	Washington	Wisconsin
Regulariser Combination	87.19±2.66	84.64±2.68	90.14±1.14	88.35±2.21
Kernel Combination	94.80±2.85	96.31±1.64	94.08±1.78	93.47±0.18
Graph Combination	95.52±1.53	97.42±1.01	94.51±1.50	93.72±0.79

Table 6: Comparison of combination methods on the Cora datasets.

combination	DS	HA	ML	PL
Regulariser Combination	59.52±3.24	69.25±4.01	73.53±1.69	61.97±0.82
Kernel Combination	77.63±2.62	88.00±3.14	86.08±0.80	79.94±0.91
Graph Combination	80.83±2.48	89.25±3.19	86.58±1.39	80.32±2.52

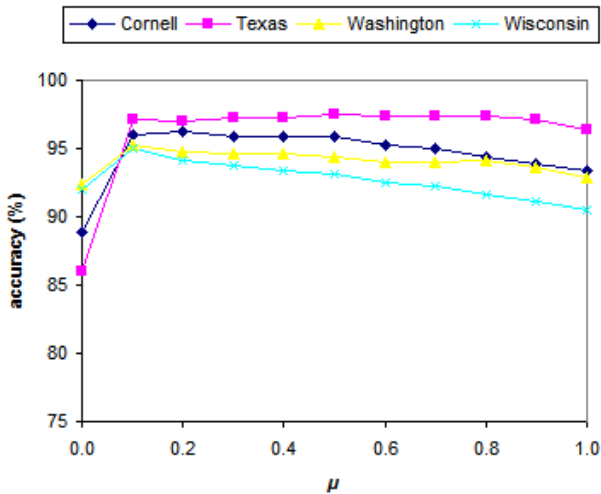


Figure 4: Effect of the combination parameter μ on the WebKB datasets.

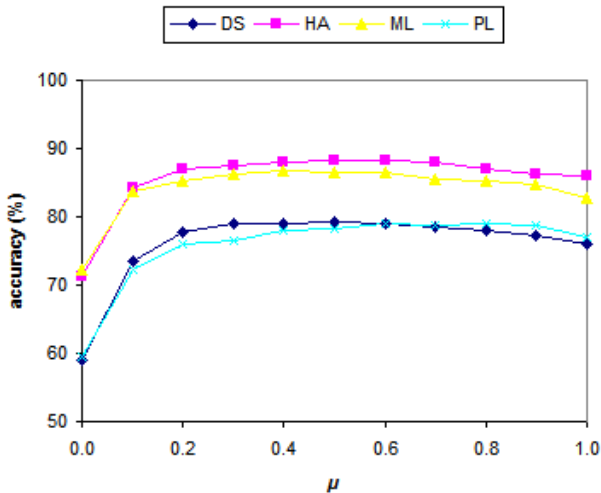


Figure 5: Effect of the combination parameter μ on the Cora datasets.

5.3.3 Effect of the Combination Parameter

Furthermore, we study the effect of the combination parameter μ using SVM with ModKer (content+link). The experimental results on the WebKB datasets and the Cora datasets are shown in Figure 4 and 5 respectively, where the combination parameter μ is varied from 0 to 1. The optimal μ on the WebKB datasets is about 0.1, which implies that the correct classification is mainly determined by the link information. The optimal μ on the Cora datasets is around 0.5, which implies that the content information and the link information have roughly equal influences on the correct classification. Nevertheless, we do not tune the combination parameter but simply set $\mu = 0.5$ when comparing ModKer with other approaches in the next section.

5.3.4 Comparison with Other Approaches

We finally compare our proposed approach with the state-of-the-art methods for classifying networked entities, some

using only entity link information while some using both entity content and entity link information. When using entity content information, each entity is regarded as a bag-of-words and pre-processed by TF \times IDF weighting. The methods being compared are briefly described as follows.

- **DGR (link)** refers to Directed Graph Regularisation [52, 51] based on only entity link information (see Section 2).
- **LapKer (link)** refers to SVM with Laplacian Kernel based on only entity link information (see Section 4).
- **ModKer (link)** refers to SVM using Modularity Kernel based on only entity link information (see Section 4).
- **PLSI+PHITS (content+link)** refers to probabilistic LSI plus probabilistic HITS [18] that exploits both entity content and entity link information (see Section 2).
- **MF (content+link)** refers to SVM using features extracted by Matrix Factorization [53] that exploits both entity content and entity link information (see Section 2). The SVM regularization parameter C is tuned on the training folds data via cross-validation.
- **SupMF (content+link+label)** refers to Supervised Matrix Factorization [53] that takes into account entity label information in addition to entity content and entity link information (see Section 2). The SVM regularization parameter C is tuned on the training folds data via cross-validation.
- **ModKer (content+link)** refers to SVM with Modularity Kernel using Graph Combination (with $\mu = 0.5$) to fuse entity content and entity link information (see Section 4).

The significance of performance difference is assessed using *McNemar's test*. When we use the term ‘significant’ in the following text, we mean a P -value ≤ 0.05 .

We show the experimental results of competing methods on the WebKB datasets in Table 7 and Figure 6. When using only entity link information, Modularity Kernel works significantly better than Laplacian Kernel on all four datasets, and outperforms Directed Graph Regularization on three of the four datasets. When using both entity content and entity link information, Modularity Kernel works significantly better than PLSI+PHITS, and slightly better than (Supervised) Matrix Factorization.

We show the experimental results of competing methods on the Cora datasets in Table 8 and Figure 7. When using only entity link information, Modularity Kernel works significantly better than Laplacian Kernel and Directed Graph Regularization on all four datasets. When using both entity content and entity link information, Modularity Kernel works significantly better than PLSI+PHITS, Matrix Factorization and Supervised Matrix Factorization.

In summary, Modularity Kernel is superior to Laplacian Kernel for real-world complex networks, and it can provide excellent classification performance in comparison with the state-of-the-art methods. It has also been shown that Modularity Kernel works well no matter whether the link-based network is derived from co-citations or direct-references and whether it is combined with the content-based network.

Table 7: Comparison with other approaches on the WebKB datasets.

method	Cornell	Texas	Washington	Wisconsin
DGR (link)	89.47±1.41	91.28±0.75	91.08±0.51	89.26±0.45
LapKer (link)	74.37±4.06	71.87±3.76	83.02±3.31	88.51±1.29
ModKer (link)	92.50±1.64	96.44±1.01	92.54±1.15	91.07±2.46
PLSI+PHITS (content+link)	80.74±0.88	76.15±1.29	85.12±0.37	83.75±0.87
MF (content+link)	93.50±0.80	96.20±0.50	93.60±0.50	92.60±0.60
SupMF (content+link+label)	93.80±0.70	97.07±1.11	93.70±0.60	93.00±0.30
ModKer (content+link)	95.52±1.53	97.42±1.01	94.51±1.50	93.72±0.79

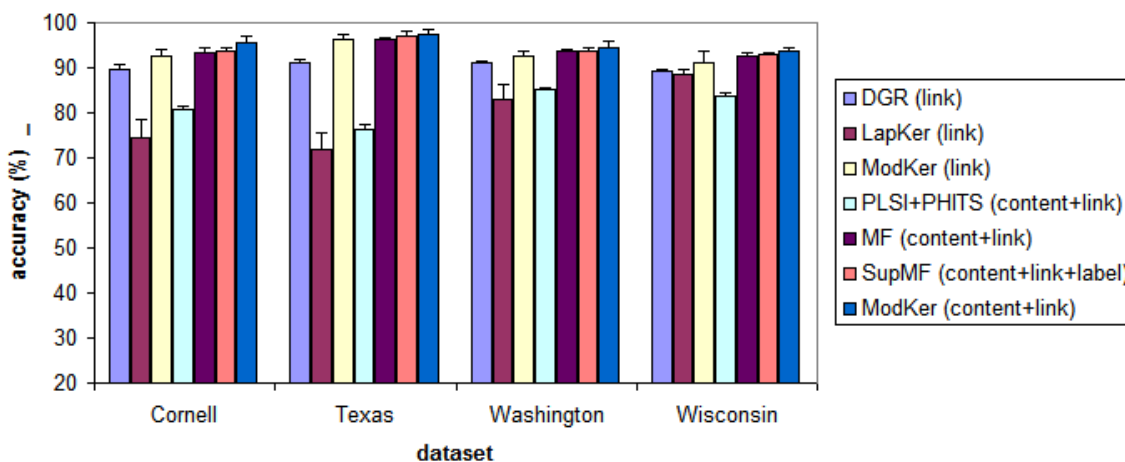


Figure 6: Comparison with other approaches on the WebKB datasets.

Table 8: Comparison with other approaches on the Cora datasets.

method	DS	HA	ML	PL
DGR (link)	46.07±0.82	65.50±2.30	59.37±0.96	56.06±0.84
LapKer (link)	57.65±4.60	62.75±6.09	77.24±2.10	75.81±2.04
ModKer (link)	76.42±4.36	87.50±2.65	82.93±1.39	77.90±2.12
PLSI+PHITS (content+link)	53.60±1.78	67.40±1.48	67.51±1.13	57.45±0.68
MF (content+link)	61.00±0.70	74.20±1.20	77.50±0.80	62.50±0.80
SupMF (content+link+label)	69.38±1.80	74.20±0.70	78.70±0.90	68.76±1.32
ModKer (content+link)	80.83±2.48	89.25±3.19	86.58±1.39	80.32±2.52

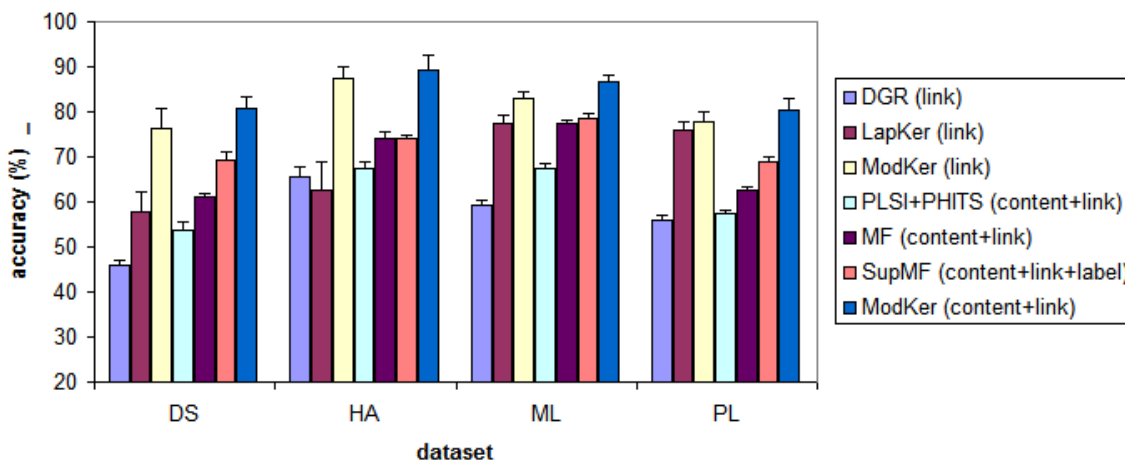


Figure 7: Comparison with other approaches on the Cora datasets.

6. CONCLUSIONS

We propose a new technique, Modularity Kernel, that can effectively exploit the latent community structure of networked entities for their classification. A number of experiments on hypertext datasets show that the proposed approach leads to excellent classification performance in comparison with the state-of-the-art methods.

So far we have focused on classification within undirected networks. It would be interesting to create a *directed* version of Modularity Kernel for classification within directed networks. We also plan to use Modularity Kernel for other applications such as clustering and ranking of networked entities, and improve the scalability of the algorithm.

Acknowledgements

We would like to thank Dr. Shenghuo Zhu for sharing his pre-processed datasets. Thanks also to the anonymous reviewers for their helpful comments.

7. REFERENCES

- [1] R. Angelova and G. Weikum. Graph-based text classification: Learn from your neighbors. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 485–492, Seattle, WA, 2006.
- [2] A. Argyriou, M. Herbster, and M. Pontil. Combining graph laplacians for semi-supervised learning. In *Advances in Neural Information Processing Systems (NIPS)*, volume 18, pages 67–74, Vancouver, Canada, 2005.
- [3] M. Belkin, I. Matveeva, and P. Niyogi. Regularization and semi-supervised learning on large graphs. In *Proceedings of the 17th Annual Conference on Learning Theory (COLT)*, pages 624–638, Banff, Canada, 2004.
- [4] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research (JMLR)*, 7:2399–2434, 2006.
- [5] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006.
- [6] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, pages 19–26, Williamstown, MA, 2001.
- [7] A. Blum, J. D. Lafferty, M. R. Rwebangira, and R. Reddy. Semi-supervised learning using randomized mincuts. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, Banff, Alberta, Canada, 2004.
- [8] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT)*, pages 92–100, Madison, WI, 1998.
- [9] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.-U. Hwang. Complex networks : Structure and dynamics. *Physics Reports*, 424(4–5):175–308, 2006.
- [10] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the 7th International World Wide Web Conference (WWW)*, pages 107–117, Brisbane, Australia, 1998.
- [11] S. Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan-Kaufman, 2002.
- [12] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 307–318, Seattle, WA, 1998.
- [13] O. Chapelle, B. Scholkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, 2005.
- [14] O. Chapelle, J. Weston, and B. Scholkopf. Cluster kernels for semi-supervised learning. In *Advances in Neural Information Processing Systems (NIPS)*, volume 15, pages 585–592, Vancouver, Canada, 2003.
- [15] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.
- [16] F. R. K. Chung and L. Lu. *Complex Graphs and Networks*. American Mathematical Society, 2006.
- [17] D. Cohn and H. Chang. Learning to probabilistically identify authoritative documents. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, pages 167–174, Standord, CA, 2000.
- [18] D. A. Cohn and T. Hofmann. The missing link - a probabilistic model of document content and hypertext connectivity. In *Advances in Neural Information Processing Systems (NIPS)*, pages 430–436, Denver, CO, USA, 2000.
- [19] M. Craven and S. Slattery. Relational learning with statistical predicate invention: Better models for hypertext. *Machine Learning*, 43(1/2):97–119, 2001.
- [20] L. Danon, J. Duch, A. Diaz-Guilera, and A. Arenas. Comparing community structure identification. *Journal of Statistical Mechanics*, page P09008, 2005.
- [21] L. Getoor, N. Friedman, D. Koller, and B. Taskar. Learning probabilistic models of link structure. *Journal of Machine Learning Research*, 3:679–707, 2002.
- [22] G. H. Golub and C. F. V. Loan. *Matrix Computations*. The Johns Hopkins University Press, 3rd edition, 1996.
- [23] M. Gustafsson, A. Lombardi, and M. Hornquist. Comparison and validation of community structures in complex networks. *Physica A*, 367:559–576, 2006.
- [24] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen. Web content categorization using link information. Technical report, Stanford University, 2007.
- [25] M. Herbster, M. Pontil, and L. Wainer. Online learning over graphs. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, pages 305–312, Bonn, Germany, 2005.
- [26] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 50–57, Berkeley, CA, 1999.
- [27] D. Jensen, J. Neville, and B. Gallagher. Why collective inference improves relational classification. In

- Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 593–598, Seattle, WA, 2004.
- [28] T. Joachims. Transductive learning via spectral graph partitioning. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, pages 290–297, Washington, DC, 2003.
- [29] T. Joachims, N. Cristianini, and J. Shawe-Taylor. Composite kernels for hypertext categorisation. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, pages 250–257, Williamstown, MA, 2001.
- [30] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
- [31] S. A. Macskassy and F. Provost. Classification in networked data: A toolkit and a univariate case study. *Journal of Machine Learning Research (JMLR)*, 8:935–983, 2007.
- [32] J. Neville and D. Jensen. Dependency networks for relational data. In *Proceedings of the 4th IEEE International Conference on Data Mining (ICDM)*, pages 170–177, Brighton, UK, 2004.
- [33] M. Newman, A.-L. Barabasi, and D. Watts. *The Structure and Dynamics of Networks*. Princeton University Press, 2006.
- [34] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45:167–256, 2003.
- [35] M. E. J. Newman. Detecting community structure in networks. *The European Physical Journal B*, 38:321–330, 2004.
- [36] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74:036104, 2006.
- [37] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences USA*, 103:8577–8582, 2006.
- [38] B. Scholkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.
- [39] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.
- [40] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22(8):888–905, 2000.
- [41] V. Sindhwani, M. Belkin, and P. Niyogi. The geometric basis of semi-supervised learning. In O. Chapelle, B. Scholkopf, and A. Zien, editors, *Semi-Supervised Learning*, chapter 12, pages 209–226. MIT Press, 2006.
- [42] V. Sindhwani, P. Niyogi, and M. Belkin. Beyond the point cloud: from transductive to semi-supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, pages 824–831, Bonn, Germany, 2005.
- [43] S. Slattery and M. Craven. Combining statistical and relational methods for learning in hypertext domains. In *Proceedings of the 8th International Workshop on Inductive Logic Programming (ILP)*, pages 38–52, Madison, WI, 1998.
- [44] M. Szummer and T. Jaakkola. Partially labeled classification with markov random walks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 945–952, Vancouver, Canada, 2001.
- [45] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Proceedings of the 18th Conference in Uncertainty in Artificial Intelligence (UAI)*, pages 485–492, Edmonton, Alberta, Canada, 2002.
- [46] A. Veloso, W. M. Jr., and M. J. Zaki. Lazy associative classification. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM)*, pages 645–654, Hong Kong, China, 2006.
- [47] Q. Yang and X. Wu. 10 challenging problems in data mining research. *International Journal of Information Technology and Decision Making*, 5(4):597–604, 2006.
- [48] Y. Yang, S. Slattery, and R. Ghani. A study of approaches to hypertext categorization. *Journal of Intelligent Information Systems (JIIS)*, 18(2–3):219–241, 2002.
- [49] T. Zhang, A. Popescul, and B. Dom. Linear prediction models with graph regularization for web-page categorization. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 821–826, Philadelphia, PA, 2006.
- [50] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Scholkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems (NIPS)*, Vancouver and Whistler, Canada, 2003.
- [51] D. Zhou, J. Huang, and B. Scholkopf. Learning from labeled and unlabeled data on a directed graph. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, pages 1036–1043, Bonn, Germany, 2005.
- [52] D. Zhou, B. Scholkopf, and T. Hofmann. Semi-supervised learning on directed graphs. In *Advances in Neural Information Processing Systems (NIPS)*, Vancouver, Canada, 2004.
- [53] S. Zhu, K. Yu, Y. Chi, and Y. Gong. Combining content and link for classification using matrix factorization. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 487–494, Amsterdam, The Netherlands, 2007.
- [54] X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005. http://www.cs.wisc.edu/~jerryzhu/pub/ssl_survey.pdf.
- [55] X. Zhu, Z. Ghahraman, and J. D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, pages 912–919, Washington, DC, 2003.