Labeling Actors in Social Networks Using a Heterogeneous Graph Kernel

Ngot Bui and Vasant Honavar

College of Information Sciences and Technology
The Pennsylvania State University
University Park, PA 16802, USA
{npb123,vhonavar}@ist.psu.edu

Abstract. We consider the problem of labeling actors in social networks where the labels correspond to membership in specific interest groups, or other attributes of the actors. Actors in a social network are linked to not only other actors but also items (e.g., video and photo) which in turn can be linked to other items or actors. Given a social network in which only some of the actors are labeled, our goal is to predict the labels of the remaining actors. We introduce a variant of the random walk graph kernel to deal with the heterogeneous nature of the network (i.e., presence of a large number of node and link types). We show that the resulting heterogeneous graph kernel (HGK) can be used to build accurate classifiers for labeling actors in social networks. Specifically, we describe results of experiments on two real-world data sets that show HGK classifiers often significantly outperform or are competitive with the state-of-the-art methods for labeling actors in social networks.

1 Introduction

Social networks (e.g. Facebook) and social media (e.g. Youtube) have provided large amounts of network data that link actors (individuals) with other actors, as well as diverse types of digital objects or items e.g., photos, videos, articles, etc. Such data are naturally represented using *heterogeneous* networks with multiple types of nodes and links. We consider the problem of labeling actors in social networks where the labels correspond to membership in specific interest groups, participation in specific activities, or other attributes of the actors. However, in many real-world social networks, labels are available for only a subset of the actors. Given a social network in which only some of the actors are labeled, our goal is to predict the labels of the remaining actors.

Accurate prediction of actor labels is important for many applications, e.g., recommending specific items (e.g., movies, musics) to actors. A variety of approaches to labeling nodes in social networks have been explored in the literature including methods that develop a relational learner to classify an actor by iteratively labeling an actor to the majority class of its neighbors [1, 2]; methods that effectively exploit correlations among the labels and attributes of objects [3–5]; semi-supervised learning or transductive learning methods [6, 7] such as random-walk based methods [8, 9] that assign a label to an actor based on the known label(s) of objects represented by node(s) reachable via random walk(s) originating at the node representing the actor. However, with

W.G. Kennedy, N. Agarwal, and S.J. Yang (Eds.): SBP 2014, LNCS 8393, pp. 27–34, 2014.

the exception of RankClass [10], Graffiti [8], EdgeCluster [11], and Assort [12, 13], most of the current approaches to labeling actors in social networks focus on *homogeneous* networks, i.e., networks that consist of a single type of nodes and/or links. RankClass and Graffiti offer probabilistic models for labeling actors in heterogeneous social networks. EdgeCluster mines the latent multi-relational information of a social network and convert it into useful features which can be used in constructing a classifier. Assort augments network data by combining explicit links with links mined from the nodes' local attributes to increase the amount of the information in the network and hence improve the performance of the network classifier [2]. Against this background, we introduce a heterogeneous graph kernel (HGK), a variant of the random walk graph kernel for labeling actors in a heterogeneous social network.

HGK is based on the following intuition: Two actors can be considered "similar" if they occur in the similar *contexts;* and "similar" actors are likely to have similar labels. We define the *context* of an object to include its direct and indirect neighbors and links between those neighbors. The similarity of two actors is defined in terms of the similarity of the corresponding contexts. We extend the random walk graph kernel [14–16] which has been previously used for labeling nodes in homogeneous networks to the setting of heterogeneous networks. The resulting HGK is able to exploit the information provided by the multiple types of links and objects in a social networks to accurately label actors in such networks. Results of experiments on two real-world data sets show that HGK classifiers often significantly outperform or are competitive with the state-of-the-art methods for labeling actors in social networks.

2 Preliminaries

A social network can be considered as a heterogeneous network of multiple types of objects and links. Formally, we define a social network as follows.

Definition 1. ***Heterogeneous Social Network.*** *A heterogeneous social network with multiple types of objects and links is represented by a graph $G = (V, E)$ in which $V = \{V_1 \cup V_2 \cup \ldots \cup V_m\}$ is a vertex set where V_p denotes the set of vertices of type p; and $E = \{\cup E_{pq} | 1 \leq p, q \leq m\}$ is an edge set where $E_{pq} = \{(x, y)\, | x \in V_p, y \in V_q\}$ is a set of edges between the two objects of types p and q, respectively.*

We define the transition probability between a node *x* and its neighbors as being inversely proportional to the number of *x*'s neighbors and the probability of remaining at node *x* with a certain stopping probability. Let T_{pq} be the transition probability matrix between nodes in V_p and V_q (note that $T_{pq} \neq T_{qp}$) and S_{pq} be the stopping probability matrix of nodes in V_q. Let $A \in \{V_1, V_2, \cdots, V_m\}$ be a set of actors in a network, we formally describe our problem as follows. Consider a social network in which each actor $x \in A$ belongs to one category in $C = \{c_1, c_2, \cdots, c_N\}$ and a subset of labeled actors A^L, our task is to complete the labeling for unlabeled actors in the subset $A^U = A - A^L$.

3 Heterogeneous Graph Kernel

3.1 Kernel Function

We consider two objects to be similar if they occur in similar contexts. We generalize the marginalized kernel [14] that has been used for computing similarity between two objects in a homogeneous networks to the heterogeneous network setting as follows.

Let x be a node in a heterogeneous network G, and let $h_x = x - x_1 - x_2 - \cdots - x_l$ be a random walk starting from x with the length of l. The probability of h_x is defined as: $P\left(h_x\right) = P_t\left(x_1|x\right) P_t\left(x_2|x_1\right) \cdots P_t\left(x_l|x_{l-1}\right) P_s\left(x_l\right)$ where P_s is stopping probability and P_t is transition probability.

We assume that the stopping probability for all nodes of all types is $P_s = \rho$ $(0 < \rho < 1)$. Let $\mathcal{N}_q\left(x_{i-1}\right)$ be the neighbors of type q of x_{i-1}, then we have:

$$\sum_{q=1}^{m} \sum_{x_i \in \mathcal{N}_q(x_{i-1})} P_t\left(x_i|x_{i-1}\right) + P_s\left(x_{i-1}\right) = 1 \tag{1}$$

Suppose that x_{i-1} is an object of type p. Let w_{pq} be the *transition probability weight* (TPW) from type p to type q where $\sum_q w_{pq} = 1$. Then, $\sum_{x_i \in \mathcal{N}_q(x_{i-1})} P_t\left(x_i|x_{i-1}\right) = (1-\rho)\, w_{pq}$. Transition probabilities from x_{i-1} to neighbors x_i of type q are assumed to be equal, i.e., $P_t\left(x_i|x_{i-1}\right) = \frac{(1-\rho) w_{pq}}{|\mathcal{N}_q(x_{i-1})|}$. We define the *linking preferences* of type p to be proportional to the TPWs of type p, i.e., $w_{p1} : w_{p2} : \ldots : w_{pm}$. In the absence of prior knowledge, we assume that $\forall q : 1 \leq q \leq m$, w_{pq} are equal.

We define the *kernel* induced *similarity* between two objects x and y of type p as follows.

$$K^p\left(x, y\right) = \sum_{h_x} \sum_{h_y} R_h\left(h_x, h_y\right) P\left(h_x\right) P\left(h_y\right) \tag{2}$$

where $K^p\left(x, y\right)$ is a kernel function; $R_h\left(h_x, h_y\right)$, the similarity between two paths h_x and h_y, is equal to 0 if they are of different lengths; otherwise, $R_h = \prod_{i=1}^{l} R_0\left(x_i, y_i\right)$. If x_i and y_i are of the same type (say p), then $R_0\left(x_i, y_i\right) = R_0^p\left(x_i, y_i\right)$ where $R_0^p\left(x_i, y_i\right)$ is defined using *Jaccard similarity* coefficient on the sets of directed neighbors of x_i and y_i as follows.

$$R_0^p\left(x_i, y_i\right) = \frac{\sum_{q=1}^{m} |\mathcal{N}_q(x_i) \cap \mathcal{N}_q(y_i)|}{\sum_{q=1}^{m} |\mathcal{N}_q(x_i) \cup \mathcal{N}_q(y_i)|} \tag{3}$$

where $\mathcal{N}_q(x_i)$ and $\mathcal{N}_q(y_i)$ are neighbors of type q of objects x_i and y_i, respectively. Otherwise, $R_0\left(x_i, y_i\right) = 0$.

3.2 Efficient Computation

Computing the kernel value between two nodes by generating random walks starting from the two nodes is computationally expensive. We adapt a technique introduced in [14, 17] for efficient computation of the random walk graph kernel in the case of homogeneous networks to the heterogeneous network setting as follows.

$$K^p\left(x, y\right) = \sum_{l=1}^{\infty} R_l^p\left(x, y\right) \tag{4}$$

where $R_l^p(x,y)$ is recursively defined using matrix form as follows: $R_l^p = \sum_{q=1}^{m} T_{pq} \left(R_0^q \circ R_{l-1}^q\right) T_{pq}^t$ and $R_1^p = \sum_{q=1}^{m} \left(T_{pq} \circ S_{pq}\right) R_0^q \left(T_{pq} \circ S_{pq}\right)^t$, T_{pq}^t is transpose of T_{pq} and "$\circ$" is Hadamard product (see the Appendix A and B for the formation and convergence proof of (4), respectively).

Let $d = \max\left(|V_1|, \ldots, |V_m|\right)$, then the time for computing kernel matrix corresponding to a random walk of length of 1 (i.e., R_1) is $O\left(|V_p| d^2\right)$. The time for computing kernel matrix R_l $(l > 1)$ is $O\left(d^3\right)$. As a result, time complexity for computing kernel matrix for type p is $O\left(l d^3\right)$.

3.3 Learning to Label Actors in Social Network

We first compute the kernel matrix that captures the pair-wise similarity between actors and normalize it to obtain: $\hat{K}^p(x,y) = {K^p(x,y)}/{\sqrt{K^p(x,x)K^p(y,y)}}$. We train a support vector machine (SVM[1]) using HGK for labeling actors in social networks[2].

4 Experimental Settings and Results

We describe results of experiments that compare the performance of HGK with that of several baseline classifiers. We also study the sensitivity of the performance of HGK to length of random walk and to linking preferences of type actor.

4.1 Social Media Data

We crawled two real-world heterogeneous social networks. The first data set is from Last.fm music network. We manually identified 11 disjoint groups (categories of users who share similar interests in music e.g., `http://www.last.fm/group/Metal` in the case of users who enjoy Heavy Metal) that contain approximately equal number of users in the network; we then crawled users and items and the links that denote the relations among the objects in the network. In particular, the subset of the Last.fm data that we use consists of 1612019 links that connect 25471 nodes. The 25471 nodes belong to one of 4 types: 10197 users (actors), 8188 tracks, 1651 artists and 5435 tags; The 1612019 links belong to one of 5 types: 38743 user-user, 765961 user-track, 8672 track-artist, 702696 track-tag, and 95947 artist-tag links.

Our second data set is from Flickr. We manually identified 10 disjoint groups (communities of users who share the same taste in pictures e.g., `http://www.flickr.com/groups/iowa/` in the case of users who share an interest in pictures that relate to the state of Iowa) of approximately equal numbers of users. The data set constructed by crawling the Flickr network contains 361787 links that connect 22347 nodes. The nodes are of one of three types: 6163 users, 14481 photos, and 1703 tags; and the links are of one of three types: 88052 user-user, 144627 user-photo, and 129108 photo-tag. In both data sets, we use the group memberships of users as class labels to train and test all models.

[1] `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

[2] The method can be applied more generally, e.g., for labeling *any* type of nodes in social networks.

4.2 Methods

We compare the performance of SVM trained using HGK with several state-of-the-art methods for labeling actors in social networks:

1. Weighted-Vote Relational Neighbor Classifier with network data augmentation (wvRN-Assort) [2, 12, 13]: a method that first augments networked data by combining explicit links with links mined from the nodes' local attributes and then uses the augmented network as input to a Weighted-Vote Relational Neighbor Classifier.
2. Network-Only Link-Based Classification with network data augmentation (nLB-Assort) [2, 12, 13]: a method which is similar to wvRN-Assort but uses the augmented network as input to a Network-Only Link-Based [4] classifier that constructs a relational feature vector for each node by aggregating the labels of its neighbors which is used to train a logistic regression model.
3. EdgeCluster [11]: a method which extracts the social dimensions of each actor, i.e., the affiliations of the actor in a number of latent social groups and uses the resulting features to generate a discriminative model to classify actors.
4. EdgeCluster-Cont [11]: a method that combines both social dimensions and features extracted from user profiles to build predictive models. For both data sets, we report results obtained using a subset of user profile features (e.g., artists, tags) that yield the best performance.
5. Augmented-Graph Kernel (AGK): a method that uses a homogeneous graph kernel [14, 17]. We augment the network data by adding an edge between two actors if they share links to a specified number (n) of items (When $n = \infty$, the method defaults to the use of homogeneous graph kernel on the unaugmented network data). For both data sets, we report results for a choice of n that yields the best performance.

4.3 Experimental Design and Results

In the first set of experiments, we compare the performance of different methods as a function of the percentage of actors in the network with known labels. For each choice of the percentage of labeled actors, we randomly select the corresponding fraction of labeled data for each node label for training and the rest for testing. We repeat this process 10 times and report the average accuracy. The length of the random walk was set equal to 1 and the linking preferences were set to be equal (i.e., TPWs of a type were set to be equal).

Figure 1 shows the results of the first set of experiments. In particular, in Last.fm data set, HGK significantly ($p < 0.05$) outperforms all other methods when at least 4% of the actors are labeled. nLB-Assort does not work well when the fraction of actors with known labels is less than 10%; This can be explained by the fact that it relies on the statistics of labels aggregated from the neighbors of an actor to label an actor. Not surprisingly, EdgeCluster-Cont which uses more information than EdgeCluster outperforms EdgeCluster. In the Flickr data set, AGK and HGK outperform other methods when the fraction of actors with known labels is less than 10%. Furthermore, AGK significantly outperforms HGK when the fraction of actors with known labels is less than 7% and both HGK and AGK significantly outperform other methods when the fraction

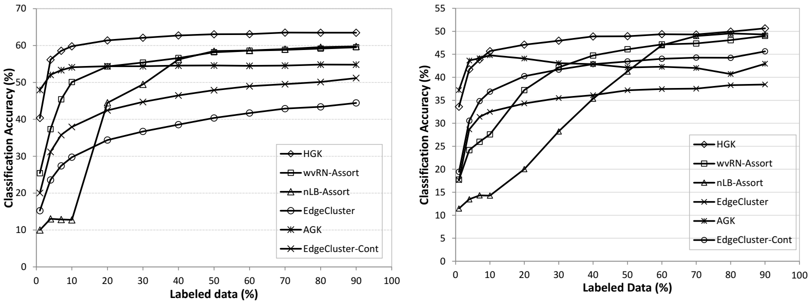

Fig. 1. Accuracies of six methods on Last.fm (left) and on Flickr (right)

of actors with known labels is between 7% to 10%. HGK significantly outperforms all other methods with labeled data when the fraction of actors with known labels ranges between 10% and 60%. Both HGK and nLB-Assort outperform other methods when the fraction of actors with known labels is between 70% and 80%. On both data sets, HGK often significantly outperforms, or is at least competitive with all other methods. This can be explained by the fact that HGK is able to exploit information provided by multiple node and link types to uncover multi-relational latent information to reliably discriminate between different actor labels.

The second set of experiments explores the sensitivity of kernel methods (HGK and AGK) as a function of the length of the of random walk. The length l of the random walk is varied from 0 to 10 with the linking preferences to be equal (across all the links from an actor). We report results averaged over 10-fold cross validation runs.

Table 1. Accuracies (%) of kernel methods with different lengths of walk. Bold numbers represent best results based on paired t-test ($p < 0.05$) on 10-fold cross validation.

	l	0	1	2	3	4	5	6	7	8	9	10
Last.fm	HGK	**61.4**	**63.8**	**62.9**	**61.0**	**59.9**	**58.8**	**58.3**	56.7	56.6	55.4	55.5
	AGK	39.0	43.8	51.2	53.1	54.5	55.1	55.0	55.1	54.9	54.2	54.0
Flickr	HGK	**49.8**	**49.7**	**46.1**	**46.0**	**46.5**	**44.8**	**44.6**	**43.2**	42.8	41.8	41.3
	AGK	33.9	38.1	42.1	42.2	41.2	41.6	40.9	41.1	41.1	41.1	41.2

Table 1 shows that kernel methods work well at some shorter walks (e.g., $l = 1, 2$). As the walk becomes longer, the performances of HGK and AGK decrease or remain the same. This indicates that the further the neighbor is from the node to be classified, the less informative it is for prediction. HGK significantly outperforms AGK with $l \leq 6$ (Results for $l = 0$ correspond to simply using the similarity values given by R_0^p).

The last set of experiments examines the performance of the learned model by fixing the length of random walk (i.e., $l = 1$) and varying the linking preferences of actors. Specifically, in Last.fm, let $w_1 = w_{useruser}$ and $w_2 = w_{usertrack}$ be the TPWs from type user. We examine the performance of the model by changing ratio $w_1 : w_2$ (1:5, 1:4, 1:3, 1:2, 1:1, 2:1, ..., 9:1). We do the same for Flickr with $w_1 = w_{useruser}$ and $w_2 = w_{userphoto}$. We report classification accuracy averaged over a 10-fold cross-validation runs.

Figure 2 shows the results of the last experiment set which investigates the influence of *linking preferences* on the performance of the HGK. In the case of Last.fm, the model performs better when the linking preference between a user and a track is higher than that between users whereas in the case of Flickr, the situation is reversed. Our results suggest that in the case of Last.fm, a surfer (exploring music) is likely to move from a user to a track more often than from a user to another user, with the opposite being true in the case of a surfer exploring pictures in Flickr.

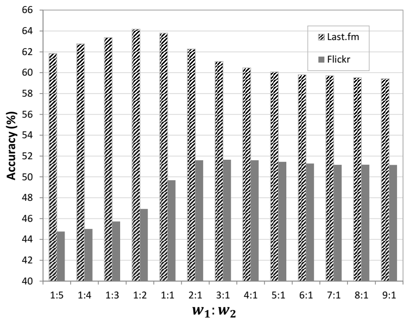

Fig. 2. Accuracies when turning $w_1 : w_2$

5 Summary

We introduced a generalization of random walk graph kernel from the setting of homogeneous networks, i.e., networks consisting of only one type of nodes and one type of links, to the setting of heterogeneous networks, i.e., networks consisting of multiple types of nodes and links. We used the resulting kernel, HGK, to train an SVM classifier for labeling actors in heterogeneous social networks. The results of our experiments show that HGK classifiers often significantly outperform or are competitive with the state-of-the-art methods for labeling actors in social networks. Some promising directions for further research include: (i) combining multiple kernels [18] that capture different notions of similarity between nodes in heterogeneous networks; and (ii) using *linking preferences* directly estimated from the data to improve the accuracy of predicted labels.

Appendix A. Based on (2), $K^p(x,y)$ can be regrouped [14] as follows: $K^p(x,y) = \sum_{l=1}^{\infty}\Big(\sum_{x_1,y_1}\Big(P_t(x_1|x)P_t(y_1|y)R_e(x_1,y_1)\Big(\sum_{x_2,y_2}\Big(P_t(x_2|x_1)P_t(y_2|y_1)\times R_e(x_2,y_2)\Big(\cdots\Big(\sum_{x_l,y_l}\Big(P_t(x_l|x_{l-1})P_t(y_l|y_{l-1})R_e(x_l,y_l)P_s(x_l)P_s(y_l)\Big)\Big)\cdots\Big)\Big)\Big)\Big)\Big)$. Now let $R_1^p(x,y) = \sum_{x_1,y_1}\Big(P_t(x_1|x)P_t(y_1|y)R_0(x_1,y_1)P_s(x_1)P_s(y_1)\Big)$. After some derivation steps, we have $R_l^p(x,y) = \sum_{x_1,y_1}\Big(P_t(x_1|x)\,P_t(y_1|y)\,R_0(x_1,y_1)\,R_{l-1}(x_1,y_1)\Big)$. So, $K^p(x,y) = \sum_{l=1}^{\infty} R_l^p(x,y)$.

We have $R_1^p = \sum_{x_1,y_1}\Big(P_t(x_1|x)P_t(y_1|y)R_0(x_1,y_1)P_s(x_1)P_s(y_1)\Big)$. Since $R_0(x_1,y_1) = 0$ for all pair (x_1,y_1) when x_1 and y_1 are not the same type, so $R_1^p = \sum_{q=1}^{m}\Big(\sum_{x_1,y_1\in V_q}\Big(P_t(x_1|x)P_s(x_1)R_0^q(x_1,y_1)P_t(y_1|y)P_s(y_1)\Big)\Big) = \sum_{p=1}^{m}(T_{pq}\circ S_{pq})R_0^q(T_{pq}\circ S_{pq})^t$. Using the same derivation method, we have $R_l^p = \sum_{q=1}^{n} T_{pq}\left(R_0^q \circ R_{l-1}^q\right)T_{pq}^t$.

Appendix B. We prove that $K^p\left(x,y\right)$ is converged when $l \rightarrow \infty$. Applying the ratio test, (4) converges when $\lim_{l\rightarrow\infty} \frac{R^p_{l+1}(x,y)}{R^p_l(x,y)} < 1 \quad (5)$. We have $R^p_l(x,y) - R^p_{l+1}(x,y) = \sum_{x_1,y_1}\Big(P_t(x_1|x)P_t(y_1|y)R_0(x_1,y_1)\Big(\sum_{x_2,y_2}\Big(P_t(x_2|x_1)P_t(y_2|y_1)R_0(x_2,y_2)\times \Big(\cdots P_s(x_l)P_s\left(y_l\right)\Big(1-\sum_{x_{l+1},y_{l+1}}P_t(x_{l+1}|x_l)P_t(y_{l+1}|y_l)R_0(x_{l+1},y_{l+1})\Big)\cdots\Big)\Big)\Big)\Big) \quad (6)$. It is sufficient for (6) to hold if $\sum_{x_{l+1},y_{l+1}} P_t\left(x_{l+1}|x_l\right) P_t\left(y_{l+1}|y_l\right) R_0\left(x_{l+1},y_{l+1}\right) < 1\,(7)$ From (1), $\sum_{x_{l+1}} P_t\left(x_{l+1}|x_l\right) = \sum_{p=1}^{m}\sum_{x_{l+1}\in\mathcal{N}_p(x_l)} P_t\left(x_{l+1}|x_l\right) = 1-\rho$. So, $\sum_{x_{l+1},y_{l+1}} P\left(x_{l+1}|x_l\right)P_t(y_{l+1}|y_l) = \sum_{x_{l+1}}P_t(x_{l+1}|x_l)\sum_{y_{l+1}}P_t(y_{l+1}|y_l) \leq (1-\rho)^2$. Since $R_0\left(.,.\right) \leq 1$ and $\rho \in (0,1)$, $\sum_{x_{l+1},y_{l+1}}P_t(x_{l+1}|x_l)P_t(y_{l+1}|y_l)R_0(x_{l+1},y_{l+1}) < 1$. □

References

1. Macskassy, S.A., Provost, F.: A simple relational classifier. In: MRDM Workshop at KDD 2003, pp. 64–76 (2003)
2. Macskassy, S.A., Provost, F.: Classification in networked data: A toolkit and a univariate case study. J. Mach. Learn. Res. 8, 935–983 (2007)
3. Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., Eliassi-Rad, T.: Collective classification in network data. AI Magazine, 93–106 (2008)
4. Lu, Q., Getoor, L.: Link-based classification. In: ICML, pp. 496–503 (2003)
5. Eldardiry, H., Neville, J.: Across-model collective ensemble classification. In: AAAI (2011)
6. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: NIPS, pp. 321–328 (2004)
7. Wang, J., Jebara, T.: Graph transduction via alternating minimization. In: ICML (2008)
8. Angelova, R., Kasneci, G., Weikum, G.: Graffiti: graph-based classification in heterogeneous networks. World Wide Web Journal (2011)
9. Lin, F., Cohen, W.W.: Semi-supervised classification of network data using very few labels. In: Proceedings of the 2010 International Conference on Advances in Social Networks Analysis and Mining, pp. 192–199 (2010)
10. Ji, M., Han, J., Danilevsky, M.: Ranking-based classification of heterogeneous information networks. In: KDD, pp. 1298–1306 (2011)
11. Tang, L., Liu, H.: Scalable learning of collective behavior based on sparse social dimensions. In: CIKM, pp. 1107–1116 (2009)
12. Macskassy, S.A.: Improving learning in networked data by combining explicit and mined links. In: AAAI, pp. 590–595 (2007)
13. Macskassy, S.A.: Relational classifiers in a non-relational world: Using homophily to create relations. ICMLA (1), 406–411 (2011)
14. Kashima, H., Tsuda, K., Inokuchi, A.: Marginalized kernels between labeled graphs. In: ICML, pp. 321–328 (2003)
15. Vishwanathan, S.V.N., Schraudolph, N.N., Kondor, R., Borgwardt, K.M.: Graph kernels. Journal of Machine Learning Research 11, 1201–1242 (2010)
16. Kang, U., Tong, H., Sun, J.: Fast random walk graph kernel. In: SDM, pp. 828–838 (2012)
17. Li, X., Chen, H., Li, J., Zhang, Z.: Gene function prediction with gene interaction networks: a context graph kernel approach. Trans. Info. Tech. Biomed. 14(1), 119–128 (2010)
18. Cortes, C., Mohri, M., Rostamizadeh, A.: Multi-class classification with maximum margin multiple kernel. In: ICML 2013, vol. 28, pp. 46–54 (May 2013)