Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 09-010

Within-network classification using local structure similarity

Chrsistian Desrosiers and George Karypis

March 30, 2009

Within-network classification using local structure similarity

Christian Desrosiers
Dep. of Computer Science & Engineering
University of Minnesota, Twin Cities
desros@cs.umn.edu

George Karypis
Dep. of Computer Science & Engineering
University of Minnesota, Twin Cities
karypis@cs.umn.edu

Abstract

Within-network classification, where the goal is to classify the nodes of a partly labeled network, is a semi-supervised learning problem that has applications in several important domains like image processing, the classification of documents, and the detection of malicious activities. While most methods for this problem infer the missing labels collectively based on the hypothesis that linked or nearby nodes are likely to have the same labels, there are many types of networks for which this assumption fails, e.g., molecular graphs, trading networks, etc. In this paper, we present a collective classification method, based on relaxation labeling, that classifies entities of a network using their local structure. This method uses a marginalized similarity kernel that compares the local structure of two nodes with parallel random walks in the network. Through experimentation on different datasets, we show our method to be more accurate than several state-of-the-art approaches for this problem.

1 Introduction

Networked data is commonly used to model the relations between the entities of a system, such as hyperlinks connecting web pages, citations relating research papers, and calls between telephone accounts. In such models, entities are represented by nodes whose label gives their type, and edges are relations between these entities. As is it often the case, important information on the nature of certain entities and links may be missing from the network. The task of recovering the missing types of entities and links (i.e. node and edge labels) based on the available information, known as within-network classification, is a semi-supervised learning problem key to several applications like image processing [3, 11], classifying document and web pages [6, 8, 21, 26, 32], classifying protein interaction and gene expression data [29], part-of-speech tagging [19], detecting malicious or fraudulent activities [7, 23, 26], and recommending items to consummers [9, 13].

Unlike traditional machine learning approaches, methods operating on networked data must deal with additional challenges that result from the relational nature of the data. One of the main challenges comes from the fact that networked data is generally not independent and identically distributed [15]. Thus, the classification of a node may have an influence on the class membership of another related node, and vice-versa. To overcome this problem, it has been widely recognized that the class membership of the nodes should be inferred simultaneously instead of individually, a technique known as collective classification [12, 16, 24, 30].

In general, collective classification methods for this problem are based on the homophily hypothesis, in which linked nodes have a tendency to belong to the same class. While strong evidence suggests this assumption to be true for several types of networks, such as social networks [1, 2], there are many types of networks for which the homophily hypothesis fails. For instance, in molecules, nearby atoms are no more likely to have the same type than distant ones. In such networks, the type of a node is instead dictated by underlying rules which may be learned by considering the relations of this node with other ones. Furthermore, while a few methods use information on linked nodes to infer the class membership, these methods only consider nodes that are directly linked, or groups of nodes that are tightly couples (e.g. cliques).

Contributions

This paper makes two contributions to the problem of within-network classification. First, it introduces a novel collective classification framework that extends the relaxation approach proposed in [27], which will be described in Section 2. While our method also uses similarity between nodes to define the class membership probabilities, it is more general in the sense that it allows the use of complex similarity kernels that are not based on a vectorial representation of the neighborhood. Secondly, although methods based random walks have recently been proposed for the within-network classification problem [6, 35], such methods strongly rely on the homophily assumption that nearby nodes are likely to have the same class. Following the success of structural kernels on the problem of graph classification [4, 17, 20], we present a novel relational classifier based on marginalized graph kernels [17], that evaluates the local structure similarity of two nodes in a network with parallel random walks. As we will show in the experimental section of this paper, considering the local structure of a node in the network can yield better classification results than simply considering the label distribution of neighbor nodes.

The rest of this paper is organized as follows. In Section 2, we present a brief review of related methods. We then describe our method in Section 3, and evaluate it experimentally on several datasets in Section 4. Finally, we conclude with a short summary of our approach and contributions.

2 Related work

Collective inference methods for the within-network classification problem can generally be divided in two groups: exact and approximate inference methods.

Exact inference methods

Exact inference methods attempt to learn the joint probability distribution of class membership (i.e. labels). Among the best known methods of this groups are those using Markov Random Fields (MRF) [19, 31]. In MRFs, the joint distribution of label probabilities is defined as the product of potential functions that operate on the cliques of the network, and the conditional probability distribution of a set of nodes can be obtained by summing over all possible assignment of labels to nodes that are not in this sets. Because this approach is generally intractable for networks having more than a few nodes, alternative approaches, such as as Gibbs sampling [11] or the junction-tree algorithm [33], are generally used. These approaches also have some practical problems. Thus, Gibbs sampling can take a long time to converge, especially for large networks [3]. Furthermore, the computational complexity of the junction-tree algorithm is exponential in the treewidth of the junction tree formed by the graph [33] which can be important for certain types of networks. Various extensions to MRFs, that also take into account observed attribute data, have also been developed. Among these are Conditional Random Fields [19], Relational Markov Networks [31] and Markov Logic Networks [10]. Probabilistic directed relational models extending the Bayesian framework, such as Relational Bayesian Networks [32], have also been proposed. However, these methods suffer from the same problems as MRFs.

Approximation methods

Due to the computational complexity of exact inference, approximation methods are normally used for the within-network classification problem. A collective classification approach that performs approximate inference on MRFs, by passing messages across links in network, is the Loopy Belief Propagation algorithm [34]. A related approach is Relaxation Labeling (RL) [8, 14, 34], where a vector containing the label probabilities of each node of unknown label is maintained. These vectors are initialized with apriori probabilities, either given or obtained from the data, and, at each subsequent iteration, are recomputed using a given relational classifier, until convergence or a maximum number of iterations is reached. Nodes of unknown label are then given the label of greatest probability. Unlike RL methods, Iterative Classification (IC) approaches [21, 25, 26] assign, at every iteration, a label to each node of unknown label, using a given relational classifier. To facilitate convergence, the amount of classified nodes is gradually increased during the process: while only the most probable classifications are made in the first iterations, every node is classified at the end of the process.

Relational classifiers

As pointed out in [24], the performance of RL and IC methods greatly depends on the relational classifier used. A classifier strongly based on the homophily assumption, the Weighted-Vote Relational Neighbor (WVRN) [22], computes the probability of a node to be in a given class as a weighted sum of the probabilities of neighbor nodes of having the same label. This simple classifier was found to work well with an RL method in the classification of documents and web pages [24]. Another classifier, which is related to the approach presented in this paper, is the Class-Distribution Relational Neighbor (CDRN) [27]. This classifier assigns, to each node v of known label, a vector whose k-th element contain the sum, over each neighbor u of v, of the probability of u to have label k. A reference vector is then obtained for each label k as the average of the vectors belonging to nodes with known label k, and the probability of a node to have label k is defined as the similarity (L_1, L_2, cosine, etc.) between its vector and the reference vector of label k. Two other relational classifiers are the Network-Only Bayes (NOB) classifier [8, 30] and the Network-Only Link-Based (NOLB) [21, 30] classifier. The former, which was originally used with an RL method to classify documents employs a naive Bayes approach to compute the label probabilities of a node, assumed to be independent, conditioned on the labels of its neighbors. Finally, the NOLB classifier learns a multiclass logistic regression model using the label distribution (raw or normalized counts, or aggregation of these values) in the neighborhood of nodes with known labels. In [21], this classifier was used within an IC method to classify documents.

3 Our classification approach

3.1 Relaxation labeling framework

Although the methods presented in this paper can be extended to the multivariate case of the within-network classification problem, we will focus on the unviariate case. We will model networked-data as a partially labeled graph $G = (V, E, W, L_V, L_E, l)$ where V is a set of nodes, E a set of edges between the nodes of V, $W \subset V$ is the set of nodes for which the true labels are known, L_V and L_E are respectively the sets of node and edge labels, and l is a function that maps each node and edge to a label of the corresponding set. To simplify the notation, we will write $l_u = l(u)$ the label of a node u and $l_{u,v} = l(u, v)$ the label of an edge (u, v). Denoting U the set of unlabeled nodes of G, i.e. $U = V \setminus W$, we need to assign to each $u \in U$ a label in L_V based on the labels of nodes in W.

As other RL methods, our approach works by iteratively updating the label probabilities of each unlabeled nodes of G until convergence. For any node $v \in V$ and any label $k \in L_V$, we denote $\pi_{v,k}$ the probability of v to have label k. If the true label of a node w is known, i.e. $w \in W$, then this value is binary: $\pi_{w,k} = \delta\left(l_w = k\right)$, where δ is the Kronecker delta such that $\delta(x = y) = 1$ if $x = y$ and 0 otherwise. Furthermore, let $K : |V|^2 \to \mathbb{R}$ be a function that evaluates the similarity between two nodes and, for notation purposes, let $\sigma_{u,v} = K(u, v)$. The probability of an unlabeled node $u \in U$ of having label $k \in L_V$ is computed from the other nodes as

$$\pi_{u,k} \;=\; \frac{\sum\limits_{v \in V} \pi_{v,k}^{\alpha}\, \sigma_{u,v}^{\beta}}{\sum\limits_{v \in V} \pi_{v,k}^{\alpha}}, \tag{1}$$

where $\alpha, \beta \geq 0$ are user-supplied parameters. In the default case where $\alpha = \beta = 1$, $\pi_{u,k}$ is a convex combination of the similarity between u and labels $v \in V$, weighted by the probability of v to have label k. For a fixed β, parameter α controls how label uncertainty influences the computation of $\pi_{u,k}$. Thus, if we set $\alpha \to \infty$, Equation 1 becomes

$$\begin{aligned}\pi_{u,k} &= \frac{\sum\limits_{v \in V} \delta\left(l_v = k\right)\ \sigma_{u,v}}{\sum\limits_{v \in V} \delta\left(l_v = k\right)} \\ &= \frac{1}{|W_k|} \sum_{v \in W_k} \sigma_{u,v},\end{aligned}$$

where $W_k \subseteq W$ is the set of nodes that have label k. As mentioned before, the RL method of [27] can be described in this framework as follows. Let $\mathbf{x}_u$ be the class vector of a node u, whose elements are the number of neighbors of u that have a certain label. Moreover, let $\overline{\mathbf{x}}_k$ be the reference vector of class k defined as

$$\overline{\mathbf{x}}_k = \frac{1}{|W_k|} \sum_{v \in W_k} \mathbf{x}_v.$$

Using the dot product of the class vectors as similarity function, i.e. $\sigma_{u,v} = \mathbf{x}_u^{\mathrm{T}} \mathbf{x}_v$, we then get

$$\begin{aligned}\pi_{u,k} &= \frac{1}{|W_k|} \sum_{v \in W_k} \mathbf{x}_u^{\mathrm{T}} \mathbf{x}_v \\ &= \mathbf{x}_u^{\mathrm{T}} \left(\frac{1}{|W_k|} \sum_{v \in W_k} \mathbf{x}_v \right) \\ &= \mathbf{x}_u^{\mathrm{T}} \overline{\mathbf{x}}_k,\end{aligned}$$

which is the CDRN classifier of [27]. The second parameter, β, controls how much the similarity of a node impacts the classification. If we use a large enough β, only the most similar nodes will influence the classification, as in nearest-neighbor classification.

Figure 1 summarizes our RL method. At step $t = 0$, we initialize the label probability of unlabeled nodes using apriori probabilities, either known or approximated from the labeled nodes. Then, at each step t, we update the label probabilities of each unlabeled node $u \in U$ and label $k \in L_V$, $\pi_{u,k}$, based on Equation (1), using the values of iteration $t-1$. This process is repeated until the label probabilities converge, i.e. the average change is inferior to a given threshold $\epsilon > 0$, or we reach a given number of iteration $T_{\max}$. Finally, we assign to each unlabeled node $u \in U$ the label k of highest probability $\pi_{u,k}$.

Note that the method presented above can also be used to classify the unlabeled edges of a graph G. The idea is to transform G by replacing each edge $(u, v) \in E$ by a new node uv and two new edges, (u, uv) and (uv, v). The graph obtained in this way will have $|V| + |E|$ nodes with labels from a set of $|L_V| + |L_E|$ nodes labels, and $2|E|$ edges with the same label. In this new graph, the nodes of the original graph are only connected to nodes corresponding to an edge, and vice-versa, such that using only the labels of direct neighbors, as it is done in the relational classifiers described in Section 2, would not provide much information.

3.2 Random walk structure similarity kernel

Relational classifiers such as those described above are based on the assumption that the information necessary to classify a node u is entirely captured by the distribution of labels in the direct neighborhood of u (i.e. first-order Markov

Algorithm 1: Relaxation labeling

```
%Initialization;
foreach $v \in V,\ k \in L_V$ do
    if $v \in W$ then $\pi^{(0)}_{v,k} \leftarrow \delta\,(l_v = k)$;
    else $\pi^{(0)}_{v,k} \leftarrow$ the apriori probability of label $k$;
$t \leftarrow 0$ ;
%Main loop;
repeat
    Compute the node similarities $\sigma^{(t)}_{u,v}$ using the $\pi^{(t)}_{u,k}$ ;
    $t \leftarrow t+1$ ;
    foreach $u \in U,\ k \in L_V$ do
        $\hat{\pi}^{(t)}_{u,k} \leftarrow \frac{\sum_{v \in V} \left(\pi^{(t-1)}_{v,k}\right)^{\alpha} \left(\sigma^{(t-1)}_{u,v}\right)^{\beta}}{\sum_{v \in V} \left(\pi^{(t-1)}_{v,k}\right)^{\alpha}}$;
    foreach $u \in U,\ k \in L$ do
        $\pi^{(t)}_{u,k} \leftarrow \frac{\hat{\pi}^{(t)}_{u,k}}{\sum_{k' \in L} \hat{\pi}^{(t)}_{u,k'}}$;
until converges or $t = T_{\max}$ ;
%Solution;
foreach $u \in U$ do
    $k \leftarrow \arg\max_{k' \in L_V} \pi^{(t)}_{u,k'}$ ;
    Assign label $k$ to $u$;
```

Figure 1: Relaxation labeling algorithm to label a set of unlabeled nodes.

assumption). As we will see in Section 4, this simple assumption does not always produce the best results. To improve this model, one could include the labels of nodes that are not directly connected to u in the distribution. However, information on the structure of this extended neighborhood, i.e. how the nodes are connected, is once more ignored. Also, this approach does not provide a clear way to give more importance to nodes closer to u in the classification.

To overcome these problems, we present a technique based on the marginalized graph kernel of [17], that extracts information on the local structure of nodes using random walks. However, there are two important differences between that setting and the one of this paper: 1) the kernel evaluates the similarity between two nodes of a same graph, instead of between two different graphs, and 2) the labels of some nodes are only known as a probability. While the kernel of [17] computes the similarity between two graphs as the probability of generating the same sequence of labels in parallel random walks traversing each of these graphs, our kernel evaluates the similarity between two nodes u and u' as the probability of generating the same sequence with random walks starting at u and u'. Moreover, to cope with label uncertainty, we make the label generation stochastic such that label k is generated at node v with probability $\pi_{v,k}$.

Using a constant walk termination probability γ, a node transition probability uniformly distributed over the edges leaving a node, as shown in Appendix A, the probability $R^{(N)}_{u,u'}$ of generating the same sequences of at most N labels starting from nodes u and u' can be expressed recursively as

$$R^{(N)}_{u,u'} = \frac{(1-\gamma)^2}{d_u d_{u'}} \sum_{v \in N_u} \sum_{v' \in N_{u'}} \sum_{k} \delta\left(l_{u,v} = l_{u',v'}\right) \pi_{v,k} \pi_{v',k} \left(\gamma^2 + R^{(N-1)}_{v,v'}\right), \qquad (2)$$

where N_u is a set containing the neighbors of a node u and $d_u = |N_u|$ is the degree of u. Other than being computed between pairs of nodes instead of graphs, this expression differs from the one of [17] by the fact that the label probabilities are also marginalized. To compute the kernel, we use the iterative approach summarized in Figure 2. In this bottom-up approach, we use Equation 2 to compute the probabilities $R^{(N)}$ based on $R^{(N-1)}$. We repeat this process for increasing values of N, until the similarity values converge, i.e. the average change is smaller than a given ϵ, or N reaches a given limit $N_{\max}$.

Algorithm 2: Structure similarity kernel

```
%Initialization;
foreach $u, u' \in V$ do  $R^{(0)}_{u,u'} \leftarrow 0$;
$N \leftarrow 0$ ;

%Main loop;
repeat
    $N \leftarrow N + 1$ ;
    foreach $u, u' \in V$ do
        Compute $R^{(N)}_{u,u'}$ using Equation 2 ;
until converges or $N = N_{\max}$ ;

%Solution;
return $R^{(N)}$ ;
```

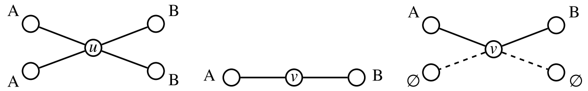
Figure 2: Bottom-up computation of the kernel.

Exploiting node degrees

A problem with the kernel definition of Equation 2 is that it does not consider the difference between the degrees of two nodes u and v, while evaluating their similarity. To illustrate this, suppose we limit the walk length in Equation 2 to $N_{\max} = 1$, i.e. we consider only the direct neighbors. Moreover, suppose that the label of every node is known, i.e. $\pi_{u,k} = \delta\left(l_u = k\right)$. Under these constraints, the similarity kernel becomes

$$K(u,v) = \frac{(1-\gamma)^2\gamma^2}{d_u d_v} \sum_{k \in L_V} n_{u,k} n_{v,k},$$

where $n_{u,k} \leq d_u$ denotes the number of neighbors of u that have label k. Thus, this simplified kernel simply compares ratios of neighbors having each label k, similar to what is done in the CDRN classifier of [27]. Using this formulation, the similarity between the nodes u and v of Figure 3 (a)-(b) is equal to the self-similarity of these nodes: $K(u,u) = K(v,v) = K(u,v) = \frac{1}{2}(1-\gamma)^2\gamma^2$.



(a) (b) (c)

Figure 3: (a)-(b) The neighborhood of two nodes u, v and (c) the transformed neighborhood of v.

In order to consider the difference in the degrees, we modify the kernel formulation as

$$R^{(N)}_{u,u'} = \frac{(1-\gamma)^2}{\max\{d_u, d_{u'}\}^2} \sum_{v\in N_u} \sum_{v'\in N'_u} \sum_{k\in L_V} \delta\left(l_{u,v} = l_{u',v'}\right) \pi_{v,k}\pi_{v',k} \left(\gamma^2 + R^{(N-1)}_{v,v'}\right) \quad (3)$$

This modification to the kernel can be interpreted in the parallel random walks framework as follows. If the degree of the node visited by a walk is less than the degree of the node visited by the other walk, temporary edges are added from this node to a dummy node of label $\varnothing \notin L_V$, such that both nodes have the same degree. With the same probability as the true neighbors, the random walk can jump to this dummy node, after which the probability of generating the same sequence becomes null. Figure 3(c) illustrates this idea for nodes u, v of (a) and (b). Using this new formulation, the similarity values for nodes u and v, again limiting the walk length to $N_{\max} = 1$, are $K(u,u) = K(v,v) = \frac{1}{2}(1-\gamma)^2\gamma^2 \geq \frac{1}{4}(1-\gamma)^2\gamma^2 = K(u,v)$.

3.3 Convergence and complexity

While the convergence of the similarity kernels defined above is proved in Appendix B, the collective classification method presented in this paper, as most RL methods, is not guaranteed to converge since the node structure similarities $\sigma_{u,v}$ vary from one iteration to the next. However, because we limit the number of allowed iterations to $T_{\max}$, the algorithm of Figure 1 will eventually return. Denote $h_K(G)$ the time complexity of the similarity kernel on a graph G, the time complexity of this algorithm is $O\left(T_{\max}(h_K(G) + |V|^2|L_V|)\right)$. Let $d_{\max}$ be the maximum degree of a node in G, the time complexity of the kernel presented in this paper, if we limit the maximum length of walk to $N_{\max}$, is $h_K(G) = O\left(N_{\max}|V|^2 d^2_{\max}|L_V|\right)$.

Although the classification process using this kernel can be expensive in the worst-case, i.e. $O\left(T_{\max}N_{\max}|V|^4|L_V|\right)$, its complexity is usually much lower in practice due to four reasons: 1) in most within-network classification problems, the number of different node labels is small, 2) the nodes of many real-life graphs have a low bounded degree (e.g., the degree is always less than 8 in molecular graphs), 3) most of the relevant information on the local structure of a node is contained within a a short distance of this node, and 4) the RL algorithm either converges in a few iterations (if there is convergence), regardless on the number of nodes in G. Following these observations, it is often possible to consider $|L_V|$, $d_{\max}$, $N_{\max}$ and $T_{\max}$ as independent of the number of nodes $|V|$ in the network, such the computational complexity of our classification approach grows quadratically with $|V|$.

4 Experimental evaluation

In this section, we test our framework on the problem of classifying the unlabeled nodes of a partly labeled graph.

4.1 Experiment data

The data used in our experiments come from three datasets provided by the IAM Graph DB Repository for Graph Based Pattern Recognition and Machine Learning [28]. The first dataset, originally proposed in [18] for the problem of predicting the mutagenicity, an adverse property that hampers the potential of a chemical compound to become a marketable drug, contains 4,337 chemical compounds tagged as mutagenic or not.

The second dataset contains a set of 2,000 chemical compounds constructed from the AIDS Antiviral Screen Database of Active Compounds [1], which are identified as having activity against HIV or not. This dataset was used as a benchmark in the problem of predicting the activity of a chemical compound [5]. The molecules of these two datasets are modeled as undirected graphs where the nodes represent atoms, node labels are the chemical symbols of these atoms, and edge are covalent bonds between atoms. Edge labels give the valence of these bonds.

The third dataset, originally used in [4], contains 600 graphs modeling proteins of the Protein Data Bank[2], equally divided in each of the six Enzyme Commission (EC) top level enzyme classes. These proteins are converted into undirected graphs using their secondary structure, such that nodes are secondary structure elements (SSE) labeled as helix, sheet, or turn. Every node is connected with an edge to its three nearest neighbors in space, and edges are labeled with their structural type. Note that a node can have more than three neighbors since the relation “nearest-neighbor” is not symmetric.

Table 1: Properties of the datasets.

Property	Mutagen.	AIDS	Protein
Nb. graphs	4,337	2,000	600
Avg. nodes	30.3	15.7	32.6
Avg. edges	30.8	16.2	62.1
Node labels	14	38	3
Edge labels	3	3	5
Freq. class	44.3%	59.3%	49.4%

Table 1 gives some properties of these datasets: the number of graphs, average number of nodes and edges, number of node and edge labels, and percentage of nodes having the most frequent class label. We notice that the graphs of these datasets are fairly sparse, i.e. average degree ranging from 2 to 4, and that the distribution of labels is uneven. Thus, in each of these datasets, there is a class (label) which contains close to half of the nodes. Since, by classifying every node to the most frequent class, it is possible to achieve accuracy equal to the

[1] `http://dtp.nci.nih.gov/docs/aids/aids_data.html`.

[2] `http://www.rcsb.org/pdb/home/home.do`.

frequency of this class, these values can be used as a baseline of the classification accuracy. For each of three datasets, we generated six test graphs by dividing the first 600 first graphs into six equally sized sets, and joining the graphs of each of these sets into one large graph. We then use the first test graph to tune the classifications methods, and the remaining five to evaluate the performance of these methods.

4.2 Experimental setting

As suggested in [24], we compare our approach with the classification methods implemented in NetKit-SRL[3]. This toolkit provides a general framework for within-network classification that allows the user to choose any combination of collective inference approach, i.e. RL, IC or Gibbs sampling, and relational classifier, i.e. WVRN, CDRN, NOB or NOLB (using either raw or normalized counts of neighbors with a given label). For additional information, the reader may refer to Section 2 or to [24]. Although we have tested every possible combination of collective classification approach and relational classifier, we have kept, for each classifier, the approach which worked best. Including the two methods proposed in this paper, i.e. our RL framework with the similarity kernels of Equations 2 and 3, a total of 7 methods, described in Table 2, are tested.

Table 2: Tested classification methods.

Method	Description
RL-WVRN:	RL with WVRN
RL-CDRN:	RL with CDRN (cosine sim. on norm. counts)
IC-NOB:	IC with NOB
IC-NOLB-count:	IC with NOLB (raw counts)
IC-NOLB-norm:	IC with NOLB (normalized counts)
RL-RW:	Our RL with the kernel of Equation 2.
RL-RW-deg:	Our RL with the kernel of Equation 3.

Again following [24], we perform 10 runs on each of the 5 test graphs, where we randomly select a subset of nodes from which we remove the labels. Once the classification methods are done, we compute the precision and recall using a weighted-average of the precision and recall obtained for each of the classes:

$$\begin{aligned} \text{precision} &= \sum_{k \in L_V} n_k \frac{\text{tp}_k}{\text{tp}_k + \text{fp}_k} \Big/ \sum_{k \in L_V} n_k, \\ \text{recall} &= \sum_{k \in L_V} n_k \frac{\text{tp}_k}{\text{tp}_k + \text{fn}_k} \Big/ \sum_{k \in L_V} n_k \\ &= \sum_{k \in L_V} \text{tp}_k \Big/ \sum_{k \in L_V} n_k, \end{aligned}$$

where $n_k = \text{tp}_k + \text{fn}_k$ is the number of classified nodes with true label k, and tp_k, fp_k, fn_k are the number of true positives, false positives and false negatives obtained by the classification method for this class. Thus, the recall corresponds to the global accuracy obtained over all classified nodes. Finally, the precision and recall values obtained in this way are averaged over the 5×10 classification runs.

[3] `http://netkit-srl.sourceforge.net/`.

4.3 Results

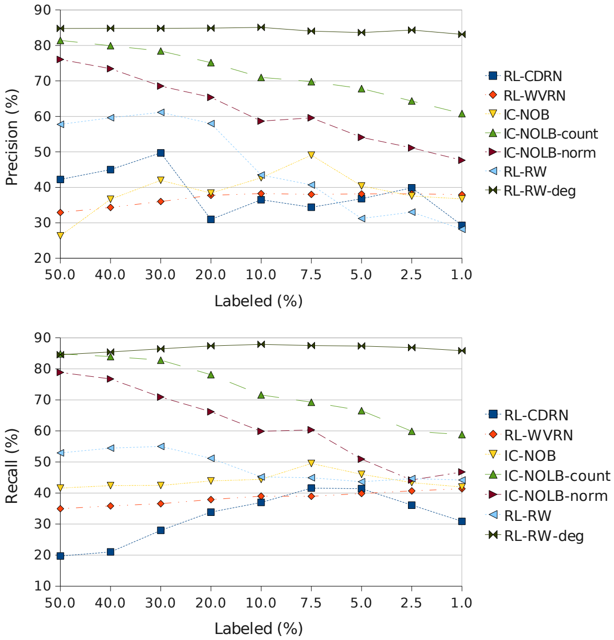


Figure 4: Precision and recall obtained for the Mutagenicity data

Figures 4, 5 and 6 give the precision and recall obtained by the seven tested methods on the Mutagenicity, AIDS and Protein data, for decreasing percentages of labeled nodes. From these results, we can see that our structure similarity kernel approach that considers node degrees, i.e. RL-RW-deg, outperforms the other classification methods, especially when a small portion of nodes are labeled. Thus, on the Mutagenicity data, this method is correctly classifies 15% more nodes than the second most accurate method, when 10% of the nodes are labeled, and 25% more when the percentage of labeled node is only 1%. Within the classification methods of Netkit-SRL, the IC method based on the multiclass regression using the raw counts, i.e. IC-NOLB-count, provides results comparable with RL-RW-deg when the labels of a sufficient number of nodes are known. However, as the number of labeled nodes reduces, this method fails to learn a proper regression model and its accuracy quickly drops. As expected, the methods based on the homophily assumption, such as RL-WDRN, perform poorly on this type of data.

Comparing our two similarity kernels, we can observe a variation in the results obtained for the three types of data. While RL-RW-deg is significantly better than RL-RW on the Mutagenicity data, the performance of these two methods is comparable on the AIDS data. This is due to the fact valence of an atom, i.e. degree of a node, is a good indicator of the type of this atom, but this information is noisy in the AIDS data since bonds to hydrogen atoms have been omitted. For the Protein data, the degree of a node provides a weak signal

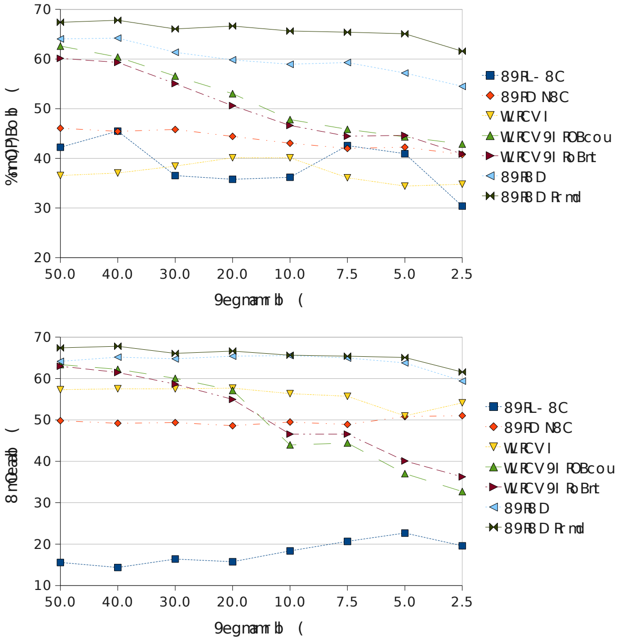


Figure 5: Precision and recall obtained for the AIDS data

since the data was created in such way that each node has approximately the same number of neighbors. This could explain the fact that RL-RW slightly outperforms RL-RW-deg on this data.

4.4 Influence of parameters and runtimes

The within-network classification framework presented in this paper is controlled by 3 parameters: the RL parameters α and β controlling the impact of label uncertainty and similarity on the classification, and the kernel parameter γ which controls the random walk lengths.

Table 3 gives the average accuracy of RL-RW-deg on the Mutagenicity data (using a percentage of labeled nodes of 50%) for different values of α and β. Interestingly, we notice that the accuracy can be improved, for this data, by increasing the importance of nodes with uncertain labels w.r.t. nodes of known label. Indeed, for every tested value of β, the greatest accuracy is obtained using a value of α lower than 1. This can be explained by the fact that using $\alpha < 1$ helps avoid local optima by providing a smoother convergence. This could also explain the poor results of the RL-CDRN method, which corresponds to using $\alpha \to \infty$ in our framework (assuming the random walk length is limited 1). We also observe that the influence of β on the accuracy varies depending on α. When $\alpha < 1$, the accuracy can be improved by increasing the importance of similar nodes in the classification, i.e. using $\beta > 1$, while the opposite occurs when $\alpha \geq 1$. This seems to indicate that basing the classification on the most similar nodes of known label actually deteriorates the classification.

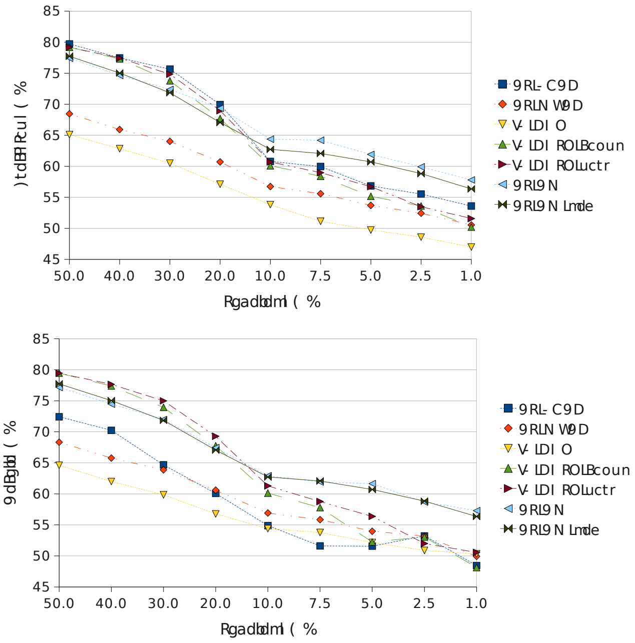


Figure 6: Precision and recall obtained for the Protein data

Table 3: Impact of parameters α and β on classification accuracy.

RL	RL β						
α	0.5	1.0	1.5	2.0	2.5	3.0	3.5
0.25	87.42	87.48	87.75	88.22	88.42	88.68	89.28
0.50	87.42	87.55	88.02	88.42	88.42	88.82	89.41
0.75	86.95	87.95	88.15	87.88	88.02	88.68	88.88
1.00	86.42	87.82	87.75	87.08	85.82	83.69	81.89
1.25	86.22	86.88	84.82	82.36	77.03	72.44	67.44
1.50	84.75	83.02	76.23	69.04	59.19	44.67	43.81
1.75	75.50	72.90	69.77	56.39	32.49	30.49	35.75
2.00	68.71	58.59	56.06	43.08	28.63	27.56	47.20

The impact of kernel parameter γ on the classification of the AIDS data (using a percentage of labeled nodes of 50%) is shown in Table 4. To illustrate how this parameter influences the length of the random walks, we varied the maximum walk length $N_{\max}$ of the kernel. When the walk length is not artificially limited, i.e. $N_{\max} \geq 10$, we notice that the accuracy is reduced when the termination probability γ increases. We also see that the greatest gain in accuracy occurs for $N_{\max} = 2$, suggesting that most of the node structure information, for this data, is contained within a distance of two edges.

Table 4: Impact of parameter γ on classification accuracy.

Kernel	Kernel γ				
$N_{\max}$	0.1	0.3	0.5	0.7	0.9
1	62.54	62.54	62.54	62.54	62.54
2	66.92	66.92	67.49	65.80	62.76
3	65.69	67.15	66.25	64.45	62.76
4	65.46	67.82	66.02	64.00	62.76
5	67.71	67.82	66.02	64.00	62.76
6	66.92	67.60	66.02	64.00	62.76
7	66.36	67.48	66.02	64.00	62.76
8	67.26	67.37	66.02	64.00	62.76
9	67.48	67.37	66.02	64.00	62.76
10	67.71	67.37	66.02	64.00	62.76

The last analysis focuses on the times required to run our methods on a machine equipped with two 2.60GHz i686 processors and 1Gb of RAM. Figure 7 gives the mean runtimes of RL-RW-deg on the Mutagenicity data (using a percentage of labeled nodes of 50%), for different values of kernel parameter γ. As a reference, we also give the runtime of RL-CDRN, the slowest Netkit-SRL classification method for this data. As predicted, the runtime roughly increases quadratically with the number of nodes in the network. While our method is noticeably slower than methods based only on direct neighbors, such as RL-CDRN, it performed the classification within 2 minutes for networks with 3,000 nodes, suggesting it could be used for networks with up to 10,000 nodes.

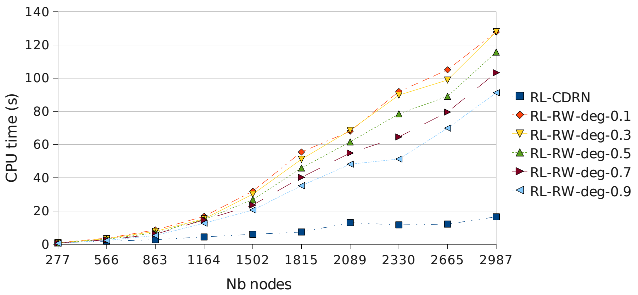


Figure 7: Runtime of our approach on the Mutagenicity data.

5 Conclusion

We presented a novel approach for the problem of within-network classification. Unlike other methods for this problem, which rely on the assumption that nearby nodes are more likely to be in the same class, this approach uses the structural similarity between nodes to infer their class. Moreover, while such methods only use the distribution of labels in the neighborhood of a node, our approach employs a similarity kernel based on random walks that extracts more information on the local structure of this node. The collective classification framework, proposed in this paper, extends a relaxation labeling method described in [27] by allowing to control the influence of label uncertainty and similarity in the classification. Furthermore, this framework uses a novel similarity kernel, based on the marginalized graph kernels proposed in [17], that computes the structure similarity between two nodes as the probability of generating the same sequence of labels in parallel random walks starting at these nodes. We also proposed a modified version of this kernel which considers the difference between the degrees of the nodes visited in the random walks. To evaluate our approach, we tested it on real-life data from the fields of toxicity and activity/function prediction in chemical compounds. The results of these experiments have shown our method to outperform several state-of-the-art methods for this problem.

References

[1] L. Adamic and E. Adar. Friends and neighbors on the web. *Social Networks*, 25(3):211–230, 2003.

[2] A. Barabasi, H. Jeong, Z. Neda, E. Ravasz, A. Schubert, and T. Vicsek. Evolution of the social network of scientific collaborations. *Physica A*, 311(3-4):590–614, 2002.

[3] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society*, 48(3):259–302, 1986.

[4] K. Borgwardt, C. Ong, S. Schönauer, S. Vishwanathan, A. Smola, and H.-P. Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21(1):47–56, 2005.

[5] H. Bunke and K. Riesen. A family of novel graph kernels for structural pattern recognition. In L. Rueda, D. Mery, and J. Kittler, editors, *Proc. 12th Iberoamerican Congress on Pattern Recognition*, LNCS 4756, pages 20–31, 2007.

[6] J. Callut, K. Francoisse, M. Saerens, and P. Dupont. Semi-supervised classification from discriminative random walks. In *Lecture Notes in Artificial Intelligence No. 5211, ECML PKDD 08*, pages 162–177. Springer, 2008.

[7] C. Castillo, D. Donato, A. Gionis, V. Murdock, and F. Silvestri. Know your neighbors: web spam detection using the web topology. In *SIGIR '07: Proc. of the 30th annual Int. ACM SIGIR Conf. on Research and development in information retrieval*, pages 423–430, New York, NY, USA, 2007. ACM.

[8] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *SIGMOD '98: Proc. of the 1998 ACM SIGMOD Int. Conf. on Management of data*, pages 307–318, New York, NY, USA, 1998. ACM.

[9] P. Domingos and M. Richardson. Mining the network value of customers. In *KDD '01: Proc. of the seventh ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 57–66, New York, NY, USA, 2001. ACM.

[10] P. Domingos and M. Richardson. Markov logic: A unifying framework for statistical relational learning. In *Proc. of the ICML-2004 Workshop on Statistical Relational Learning and its Connections to Other Fields*, pages 49–54, 2004.

[11] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *Neurocomputing: foundations of research*, pages 611–634, 1988.

[12] L. Getoor and C. P. Diehl. Link mining: a survey. *SIGKDD Explorations: Special Edition on Link Mining*, 7(2):3–12, 2005.

[13] Z. Huang, H. Chen, and D. Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Transactions on Information Systems*, 22(1):116–142, 2004.

[14] R. A. Hummel and S. W. Zucker. On the foundations of relaxation labeling processes. *Readings in computer vision: issues, problems, principles, and paradigms*, pages 585–605, 1987.

[15] D. Jensen and J. Neville. Linkage and autocorrelation cause feature selection bias in relational learning. In *ICML '02: Proc. of the Nineteenth Int. Conf. on Machine Learning*, pages 259–266, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[16] D. Jensen, J. Neville, and B. Gallagher. Why collective inference improves relational classification. In *KDD '04: Proc. of the tenth ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*, pages 593–598, New York, NY, USA, 2004. ACM.

[17] H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 321–328. AAAI Press, 2003.

[18] J. Kazius, R. McGuire, and R. Bursi. Derivation and validation of toxicophores for mutagenicity prediction. *Journal of Medicinal Chemistry*, 48(1):312–320, 2005.

[19] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML '01: Proc. of the Eighteenth Int. Conf. on Machine Learning*, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[20] X. Li, Z. Zhang, H. Chen, and J. Li. Graph kernel-based learning for gene function prediction from gene interaction network. In *BIBM '07: Proc. of the 2007 IEEE Int. Conf. on Bioinformatics and Biomedicine*, pages 368–373, Washington, DC, USA, 2007. IEEE Computer Society.

[21] Q. Lu and L. Getoor. Link-based classification. In T. Fawcett, N. Mishra, T. Fawcett, and N. Mishra, editors, *Proc. 12th Int'l Conf. Machine Learning (ICML)*, pages 496–503. AAAI Press, 2003.

[22] S. A. Macskassy and F. Provost. A simple relational classifier. In *Proc. of the 2nd Workshop on Multi-Relational Data Mining (MRDM 03)*, pages 64–76, 2003.

[23] S. A. Macskassy and F. Provost. Suspicion scoring of networked entities based on guilt-by-association, collective inference, and focused data access. In *In Proc. of the First Int. Conf. on Intelligence Analysis (IA)*, 2005.

[24] S. A. Macskassy and F. Provost. Classification in networked data: A toolkit and a univariate case study. *Journal of Machine Learning Research*, 8:935–983, 2007.

[25] L. McDowell, K. M. Gupta, and D. W. Aha. Cautious inference in collective classification. In *Proc. of the 22nd AAAI Conf. on Artificial Intelligence*, pages 596–601, 2007.

[26] J. Neville and D. Jensen. Leveraging relational autocorrelation with latent group models. In *MRDM '05: Proc. of the 4th Int. workshop on Multi-relational mining*, pages 49–55, New York, NY, USA, 2005. ACM.

[27] C. Perlich and F. Provost. Distribution-based aggregation for relational learning with identifier attributes. *Machine Learning*, 62(1-2):65–105, 2006.

[28] K. Riesen and H. Bunke. IAM graph database repository for graph based pattern recognition and machine learning. accepted for publication in SSPR 2008, 2008.

[29] E. Segal, H. Wang, and D. Koller. Discovering molecular pathways from protein interaction and gene expression data. *Bioinformatics*, 19(S1 (Proc. ISMB)):264–272, 2003.

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

[31] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *UAI '02, Proc. of the 18th Conf. in Uncertainty in Artificial Intelligence*, pages 485–492. Morgan Kaufmann, 2002.

[32] B. Taskar, E. Segal, and D. Koller. Probabilistic classification and clustering in relational data. In *In Proc. of the Seventeenth Int. Joint Conf. on Artificial Intelligence*, pages 870–878, 2001.

[33] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. Technical report, Dept. of Statistics, September 2003.

[34] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *Information Theory, IEEE Transactions on*, 51(7):2282–2312, 2005.

[35] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *In Proc. of the Twentieth Int. Conf. on Machine Learning (ICML)*, pages 912–919, 2003.

A Kernel derivation

Denote $p_t(v|u)$ the probability that the walk jumps from a node u to an adjacent node v and $p_e(v)$ the probability that the walk stops at node v, satisfying the constraint that

$$p_e(u) + \sum_{v \in V} p_t(v|u) \;=\; 1. \tag{4}$$

Following these definitions, the probability of visiting a sequence of nodes $\mathbf{v} = (\mathbf{v}_0, \ldots, \mathbf{v}_n)$ in a random walk starting at node $\mathbf{v}_0$ is

$$p(\mathbf{v}) = \left(\prod_{i=1}^{n} p_t(\mathbf{v}_i|\mathbf{v}_{i-1}) \right) p_e(\mathbf{v}_i).$$

Let $p_l(k|v)$ and $p_l(k|u,v)$ denote, respectively, the probability of generating label $k \in L_V$ at node v and the probability of generating label $k' \in L_E$ while traversing edge (u,v). Given node sequence $\mathbf{v}$, the conditional probabilities of generating the sequences of node labels $\mathbf{s}$ and edge labels $\mathbf{q}$ are

$$\begin{aligned} p(\mathbf{s}|\mathbf{v}) &= \prod_{i=1}^{n} p_l(\mathbf{s}_i|\mathbf{v}_i) \\ p(\mathbf{q}|\mathbf{v}) &= \prod_{i=1}^{n} p_l(\mathbf{q}_i|\mathbf{v}_{i-1}, \mathbf{v}_i) \end{aligned}$$

Let $\mathcal{W}_u^{(n)}$ be the set of possible sequences of $n+1$ nodes visited in a random walk starting at node u. The marginalized probability of a sequence $\mathbf{s}$, given a start node $u = \mathbf{v}_0$, is obtained by summing over all sequences of $\mathcal{W}_u^{(n)}$:

$$\begin{aligned} p(\mathbf{s},\mathbf{q}|u) &= \sum_{n=1}^{\infty} \sum_{\mathbf{v}\in\mathcal{W}_u^{(n)}} p(\mathbf{s}|\mathbf{v})p(\mathbf{q}|\mathbf{v})p(\mathbf{v}) \\ &= \sum_{n=1}^{\infty} \sum_{\mathbf{v}} \left(\prod_{i=1}^{n} p_t(\mathbf{v}_i|\mathbf{v}_{i-1}) p_l(\mathbf{s}_i|\mathbf{v}_i) p_l(\mathbf{q}_i|\mathbf{v}_{i-1},\mathbf{v}_i) \right) p_e(\mathbf{v}_i) \end{aligned}$$

Denote $\mathcal{S}^{(n)}$ and $\mathcal{Q}^{(n)}$ the set containing, respectively, all sequences of n node labels and edge labels, the probability of generating the same sequence in two parallel random walks starting at nodes u and u', written $K(u,u')$ is given by

$$\begin{aligned} K(u,u') &= \sum_{n=1}^{\infty} \sum_{\mathbf{s}\in\mathcal{S}^{(n)}} \sum_{\mathbf{q}\in\mathcal{Q}^{(n)}} p(\mathbf{s},\mathbf{q}|u)p(\mathbf{s},\mathbf{q}|u') \\ &= \sum_{n=1}^{\infty} \sum_{\mathbf{s},\mathbf{q}} \sum_{\mathbf{v}\in\mathcal{W}_u^{(n)}} \sum_{\mathbf{v}'\in\mathcal{W}_{u'}^{(n)}} \left(\prod_{i=1}^{n} p_t(\mathbf{v}_i|\mathbf{v}_{i-1}) p_t(\mathbf{v}'_i|\mathbf{v}'_{i-1}) p_l(\mathbf{s}_i|\mathbf{v}_i) p_l(\mathbf{s}_i|\mathbf{v}'_i) \right. \\ &\qquad \left. \ldots p_l(\mathbf{q}_i|\mathbf{v}_{i-1},\mathbf{v}_i) p_l(\mathbf{q}_i|\mathbf{v}'_{i-1},\mathbf{v}'_i) \right) p_e(\mathbf{v}_n) p_e(\mathbf{v}'_n) \\ &= \sum_{n=1}^{\infty} \sum_{\mathbf{s},\mathbf{q}} \sum_{\mathbf{v},\mathbf{v}'} \left(\prod_{i=1}^{n} a(\mathbf{v}_{i-1},\mathbf{v}'_{i-1},\mathbf{v}_i,\mathbf{v}'_i,\mathbf{s}_i,\mathbf{q}_i) \right) p_e(\mathbf{v}_n) p_e(\mathbf{v}'_n), \end{aligned}$$

where

$$\begin{aligned} a(\mathbf{v}_{i-1},\mathbf{v}'_{i-1},\mathbf{v}_i,\mathbf{v}'_i,\mathbf{s}_i,\mathbf{q}_i) &= p_t(\mathbf{v}_i|\mathbf{v}_{i-1}) p_t(\mathbf{v}'_i|\mathbf{v}'_{i-1}) \\ &\qquad \ldots p_l(\mathbf{s}_i|\mathbf{v}_i) p_l(\mathbf{s}_i|\mathbf{v}'_i) p_l(\mathbf{q}_i|\mathbf{v}_{i-1},\mathbf{v}_i) p_l(\mathbf{q}_i|\mathbf{v}'_{i-1},\mathbf{v}'_i). \end{aligned}$$

The computation of $K(u,u')$ can be greatly simplified using the following recurrence: the probability of generating the same sequence of n labels two parallel random walks starting at nodes u and u', written $r^{(n)}_{u,u'}$, can be obtained from the probability of visiting nodes v and v', respectively from u and u', and the probability of generating the same sequences of $n-1$ node and edge labels, starting at nodes v and v'. This recurrence can be written as

$$r^{(n)}_{u,u'} = \begin{cases} \displaystyle\sum_{v,v'\in V} \sum_{k\in L_V} \sum_{k'\in L_E} a(u,u',v,v',k,k')\; r^{(n-1)}_{v,v'} & , n \geq 1 \\ p_e(u)\; p_e(u') & , n = 0 \end{cases}$$

The probability of generating the same sequences of at most N labels starting from nodes u and u', written $R^{(N)}_{u,u'}$, is then

$$\begin{aligned} R^{(N)}_{u,u'} &= \sum_{n=1}^{N} r^{(n)}_{u,u'} \\ &= \sum_{n=1}^{N} \sum_{v,v' \in V} \sum_{k \in L_V} \sum_{k' \in L_E} a(u,u',v,v',k,k')\; r^{(n-1)}_{v,v'} \\ &= \sum_{v,v'} \sum_{k,k'} a(u,u',v,v',k,k') \sum_{n=1}^{N} r^{(n-1)}_{v,v'} \\ &= \sum_{v,v'} \sum_{k,k'} a(u,u',v,v',k,k')\; \left(p_e(v) p_e(v') + R^{(N-1)}_{v,v'} \right), \end{aligned}$$

where $R^{(0)}_{u,u'} = 0$ for all u, u'.

Denote N_u the neighbors of node u and let $d_u = |N_u|$ be the degree of u. Setting the termination probabilities of u to a constant $p_e(u) = \gamma$, and letting the transition probabilities be uniform over the neighbors of u, following the constraint of Equation 4, we have $p_t(v|u) = (1-\gamma)/d_u$ if $v \in N_u$ and 0 otherwise. Furthermore, using $p_l(k|v) = \pi_{v,k}$ and $p_l(k'|u,v) = \delta\left(l_{u,v} = k'\right)$ as node and edge label probabilities, the formulation of the kernel becomes

$$\begin{aligned} R^{(N)}_{u,u'} &= \frac{(1-\gamma)^2}{d_u d_{u'}} \sum_{v \in N_u} \sum_{v' \in N_{u'}} \sum_{k \in L_V} \sum_{k' \in L_E} \delta\left(l_{u,v} = k'\right) \delta\left(l_{u',v'} = k'\right) \pi_{v,k} \pi_{v',k} \left(\gamma^2 + R^{(N-1)}_{v,v'} \right) \\ &= \frac{(1-\gamma)^2}{d_u d_{u'}} \sum_{v,v'} \sum_{k} \delta\left(l_{u,v} = l_{u',v'}\right) \pi_{v,k} \pi_{v',k} \left(\gamma^2 + R^{(N-1)}_{v,v'} \right) \end{aligned}$$

B Proof of convergence

Proposition 1. *The similarity kernel defined by Equation 2 converges for any* $0 < \gamma \leq 1$.

Proof. Following the ratio test, $R^{(N)}_{u,u'}$ converges for every $u, u' \in V$ if

$$\lim_{n \to \infty} |r^{(n+1)}_{u,u'}| / |r^{(n)}_{u,u'}| < 1.$$

We prove this by induction. Since d_v and $\pi_{v,k}$ are non-negative, for all $v \in V$ and $k \in L_V$, then every $r^{(n)}_{u,u'}$, computed by summing and multiplying these terms, is also non-negative. Suppose that, for all $u, u' \in V$ and all $1 \leq m \leq n$, $r^{(m)}_{u,u'} < r^{(m-1)}_{u,u'}$ holds. We show that these properties still hold for $n+1$. Thus,

$$\begin{aligned} r^{(n+1)}_{u,u'} - r^{(n)}_{u,u'} &= \frac{(1-\gamma)^2}{d_u d_{u'}} \sum_{v,v'} \sum_{k} \delta\left(l_{u,v} = l_{u',v'}\right) \pi_{v,k} \pi_{v',k} \left(r^{(n)}_{v,v'} - r^{(n-1)}_{v,v'} \right) \\ &\leq 0, \end{aligned}$$

since the differences at the right of the sum are negative. For the same reason, $r^{(n+1)}_{u,u'} - r^{(n)}_{u,u'} = 0$ implies that $r^{(n+1)}_{u,u'} = r^{(n)}_{u,u'} = 0$. In this case, all terms of the series following n are null and the series converges. We therefore assume that

$r^{(n+1)}_{u,u'} < r^{(n)}_{u,u'}$ and show that the property holds for $n = 1$. We have

$$\begin{aligned} r^{(1)}_{u,u'} - r^{(0)}_{u,u'} &= \frac{(1-\gamma)^2\gamma^2}{d_u d_{u'}} \sum_{v,v'} \sum_{k} \delta\left(l_{u,v} = l_{u',v'}\right) \pi_{v,k}\pi_{v',k} \; - \; \gamma^2 \\ &\leq \gamma^2 \left[\frac{(1-\gamma)^2}{d_u d_{u'}} \sum_{v,v'} \sum_{k} \pi_{v,k}\pi_{v',k} \quad - 1 \right]. \end{aligned}$$

Once again, we ignore the converging case where $r^{(1)}_{u,u'} = r^{(0)}_{u,u'} = 0$, and prove $r^{(1)}_{u,u'} < r^{(0)}_{u,u'}$ by showing that the sum on the right-hand side of the previous equation is strictly inferior to 1:

$$\begin{aligned} \frac{(1-\gamma)^2}{d_u d_{u'}} \sum_{v,v'} \sum_{k} \pi_{v,k}\pi_{v',k} &< \frac{(1-\gamma)^2}{d_u d_{u'}} \sum_{v,v'} \sum_{k,k'} \pi_{v,k}\pi_{v',k'} \\ &= \frac{(1-\gamma)^2}{d_u d_{u'}} \left(\sum_{v} \sum_{k} \pi_{v,k} \right) \left(\sum_{v'} \sum_{k'} \pi_{v',k'} \right) \\ &= \frac{(1-\gamma)^2}{d_u d_{u'}} d_u d_{u'} \quad < \; 1, \end{aligned}$$

where we have used the facts that $\gamma > 0$ and that $\pi_{v,k}$ sum to 1 over k. □

Proposition 2. *The similarity kernel defined by Equation 3 converges for any* $0 < \gamma \leq 1$.

Proof. This can be shown using the same approach as with Proposition 1 □