Transduction on Directed Graphs via Absorbing Random Walks

Jaydeep De*, Xiaowei Zhang*, Feng Lin†, Li Cheng*‡, *Senior Member, IEEE*
*Bioinformatics Institute, A*STAR, Singapore
†School of Computer Science and Computer Engineering, Nanyang Technological University, Singapore
‡School of Software Engineering, Chongqing University, China

Abstract—In this paper we consider the problem of graph-based transductive classification, and we are particularly interested in the directed graph scenario which is a natural form for many real world applications. Different from existing research efforts that either only deal with undirected graphs or circumvent directionality by means of symmetrization, we propose a novel random walk approach on directed graphs using *absorbing* Markov chains, which can be regarded as maximizing the accumulated expected number of visits from the unlabeled transient states. Our algorithm is simple, easy to implement, and works with large-scale graphs on binary, multiclass, and multi-label prediction problems. Moreover, it is capable of preserving the graph structure even when the input graph is sparse and changes over time, as well as retaining weak signals presented in the directed edges. We present its intimate connections to a number of existing methods, including graph kernels, graph Laplacian based methods, and spanning forest of graphs. Its computational complexity and the generalization error are also studied. Empirically, our algorithm is evaluated on a wide range of applications, where it has shown to perform competitively comparing to a suite of state-of-the-art methods. In particular, our algorithm is shown to work exceptionally well with large sparse directed graphs with e.g. millions of nodes and tens of millions of edges, where it significantly outperforms other state-of-the-art methods. In the dynamic graph setting involving insertion or deletion of nodes and edge-weight changes over time, it also allows efficient online updates that produce the same results as of the batch update counterparts.

Index Terms—Random Walks on Directed Graphs, Transductive Learning, Absorbing Markov Chain, Transduction Generalization Error

1 Introduction

We focus on the following graph transduction problem: Given a directed graph with certain nodes labeled, make predictions on the unlabeled nodes by propagating the class labels following the underlining directed graph structure. Different from undirected graphs that delineate symmetric weight between adjacent nodes, in directed graphs, edge or link directions are well preserved in its weight matrix. This information is particularly useful in many real life applications that can be naturally characterized using directed graphs, including automated delineation of filamentary structures in images and videos [1], [2], [3], [4], [5], image classification [6] and clustering [7], network and link analysis in hyperlinks of webpages as well as citations of papers [8], [9], [10], [11], among others. Moreover, the fast pace of big data production and storage, together with the scarcity of annotated labels, also create the need for algorithms capable of scaling up to make predictions on large-scale directed graphs with few known labels.

Since being introduced by Vapnik in the nineties, many research efforts have already been devoted to graph-based transduction or transductive learning, as can be found in the comprehensive reviews in [12], [13]. Most of the existing literature work with undirected graphs. For example, the harmonic functions on Gaussian fields [14], the local and global consistency of [15], the quadratic criterion [16], the commute time kernel [11], the partially absorbing random walks [17], and the greedy max-cut [18]. Besides, there are a few methods specifically dedicated to directed graphs, including in particular [19], [20], [6]. One major difficulty in learning with directed graphs lies in the asymmetric nature of the weight matrix introduced by these directed edges or links. This is often regarded as cumbersome when aligning with the key concepts developed for undirected graphs that are symmetric by nature, such as graph Laplacians. It thus leads to the widely adopted symmetrization trick in e.g. the construction of graph Laplacians [19], or co-link similarity matrices [10], or covariance kernel [20]. Unfortunately, important information conveyed by edge directions is still lost. There have been a few methods for large-scale transduction such as [21], [22], which are however not ready to be used for directed graphs.

We propose a random walk on absorbing Markov chains approach to the problem of transductive learning on directed graphs, where the edge directions – the key aspect of directed graphs, are well preserved. Our algorithm is intuitive to understand, easy to implement, and by working with absorbing Markov chains [23], the sparse nature of the graph structure is also retained, which is important in the context of predictions on large-scale directed graphs.

The most related work is that of [19], which generalizes their earlier framework of regularized risk minimization on undirected graphs [15] to directed graphs by discriminatively normalizing in-links and out-links, as well as adopting the directed graph Laplacian of [24]. The method of [19] involves utilizing a symmetrization trick to construct symmetric Laplacian matrix for digraphs [24]. In contrast, we directly work with asymmetric matrices, which is the key in preserving edge directions. In addition, the construction in [19] relies on an irreducible Markov chain,

The first two authors contributed equally in this work. Corresponding author: Li Cheng (email: chengli@bii.a-star.edu.sg).

which however only apply to *strongly connected* directed graphs, that is, there is a directed path from any node to any other node of the graph. Since in practice the directed graphs are usually not necessarily strongly connected, a teleporting random walk trick (e.g [25]) is adopted by inserting bi-directional edges between all node pairs with an equal weight. The resulting method thus works only with non-sparse matrices, instead our algorithm is able to preserve sparse graph structures and edge directional information of the input. Besides, our algorithm is able to efficiently work with large-scale graphs that might be too big to be considered by [19]. More recently, the family of partially absorbing random walks (PARWs) is proposed in [17], which can be formulated as a special case of the absorbing Markov chains considered in our approach. Notice that absorbing random walks have also been considered by [26], which recursively constructs absorbing nodes. However, [26] focuses on ranking with diversity, which is an entirely different problem in natural language processing.

The main contributions of this work are four folds. (1) A novel random walk on *absorbing* Markov chain approach is proposed to the problem of transductive classification on directed graphs. (2) An efficient algorithm is provided that exploits the inherent sparse graph structure, while it also maintains and directly utilizes edge directional information. In addition, an optimal one-step increment/decrement update (aka online update) is introduced, which is handy in scenarios where local changes are made to graphs over time, or prediction are made on out-of-sample instances. The proposed algorithm also bears interesting connections to existing works including undirected graph Laplacian, diffusion graph kernels, spanning forest of graphs, graph-based proximity measure, among others. (3) We present several interesting properties of the fundamental matrix of absorbing Markov chain, a central element in our approach. (4) We conduct theoretical analysis on its generalization error, as well as extensive and systematic empirical simulations on various applications to examine the characteristics of the proposed algorithm and its performance with respect to existing state-of-the-art methods. [1]

2 Our Approach: Random Walks on Directed Graphs

Let $G = (\mathcal{V}, \mathcal{E}, W)$ denote a directed graph or digraph, where $\mathcal{V} = \{v_1, v_2, \dots v_n\}$ is the set of nodes, $\mathcal{E}$ is the set of directed edges each connecting two adjacent nodes, and $W = [w_{ij}] \in \mathbb{R}^{n\times n}$ is the asymmetric non-negative matrix with $w_{ij} \geq 0$ denoting the weight associated with the directed edge from v_i to v_j. The in-degree of each node v_j is computed as $d_j^- = \sum_{i=1}^n w_{ij}$, and define in-degree matrix $D = \mathrm{diag}(d_1^-, \cdots, d_n^-)$. A column stochastic transition probability matrix, $P = [p_{ij}]$, can be constructed as $p_{ij} = \frac{w_{ij}}{d_j^-}$, or equivalently as $P = WD^{-1}$. An important remark is that random walks on an undirected graph can be regarded as a special case in our context, since in its weight matrix W, the symmetric pair w_{ij} and w_{ji} correspond to bi-directional edges with the same weights – which can be characterized by a reversible Markov chain. In fact, as illustrated in Fig. 1(a), *loops*, *self-loops*, and *bi-directional* edges (i.e. two edges between adjacent nodes), as well as mixed graphs (of directed and undirected edges) are all within the scope of our digraph definition.

1. Implementations of our approach and related comparison methods can be obtained from https://web.bii.a-star.edu.sg/archive/machine_learning/Projects/filaStructObjs/Tracing/transDigraph/index.htm.

In this paper, we focus on a transductive inference scenario where labels from the set of few labeled nodes $\mathcal{V}_l$ are to be propagated to the rest of nodes, namely the set of unlabeled nodes $\mathcal{V}_u$, with $\mathcal{V} = \mathcal{V}_l \cup \mathcal{V}_u$. The labels here could be binary or multiclass. To simplify the notation, we assume $\mathcal{V}_l$ contains the first l nodes, $\mathcal{V}_l = \{v_1, \dots, v_l\}$. To accommodate label information, we define a label matrix Y of size $n \times K$ (assuming there are K class labels available), with each entry Y_{ik} containing 1 provided the node i belongs to $\mathcal{V}_l$ and is labeled with class k, and 0 otherwise. Besides, we define the ground-truth label vector $\mathbf{y}$, a vector of length n including two disjoint parts $\mathbf{y}_l$ and $\mathbf{y}_u$: $\mathbf{y}_l$ is the input label vector of length l over the set of labeled nodes, with each entry y_i for the input class assignment of node $v_i \in \mathcal{V}_l$; $\mathbf{y}_u$ is the hold-out ground-truth label for the unlabeled nodes, i.e. a vector of length $n-l$. Similarly, define the initial label vector $\hat{\mathbf{y}}$ containing also two parts, $\hat{\mathbf{y}}_l := \mathbf{y}_l$ and $\hat{\mathbf{y}}_u = \mathbf{0}$, where $\mathbf{0}$ is an all zero vector of length $n-l$. Define the prediction vector $\mathbf{y}^*$ with also two parts $\mathbf{y}_l^* := \mathbf{y}_l$, as well as $\mathbf{y}_u^*$ of length $n-l$, containing the prediction results, where each y_i^* denotes the predicted class assignment for a node $v_i \in \mathcal{V}_u$.

2.1 Our Absorbing Markov Chain and its Fundamental Matrix

There however exists an issue: the graph often contains source nodes (i.e. nodes with $d_j^- = 0$) [2], for which the corresponding columns in P are zero vectors – then P is not stochastic anymore. Inspired by the PageRank method used in Google search engine [25], we consider an augmented graph to ameliorate this situation: One extra node v_{n+1} is introduced that is further connected to each of the source nodes $\mathcal{V}_l$ with a equal weight of 1 and connected to the rest of nodes with a equal weight of $(1-\alpha)$ where $\alpha \in (0, 1)$. A self-connecting edge $e = (v_{n+1}, v_{n+1})$ with a transition probability 1 is further imposed on this new node v_{n+1}, to save itself from being another source node. This leads to a digraph as displayed in Fig. 1(b), which also corresponds to an augmented column-stochastic transition probability matrix

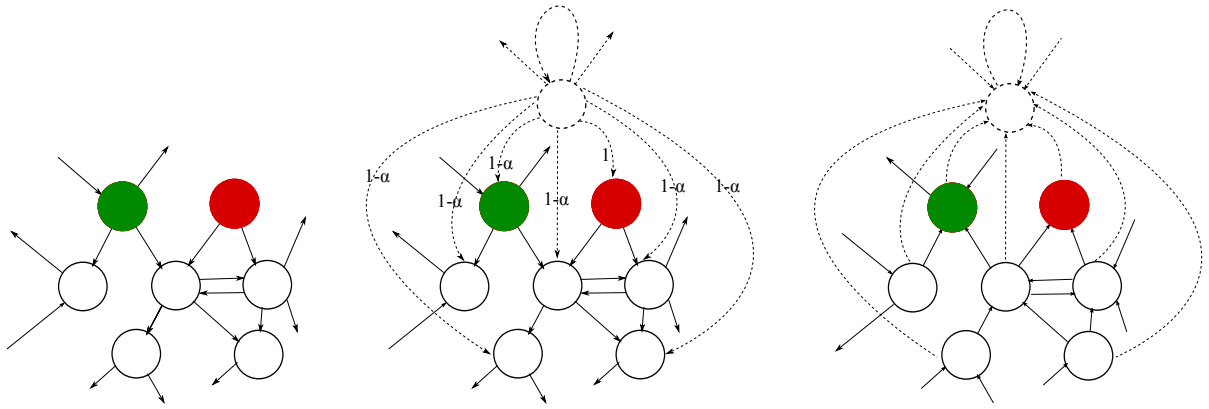
$$\tilde{P} = \left(\begin{array}{c|c} \alpha P & 0 \\ \hline \mathbf{c} & 1 \end{array}\right) \in \mathbb{R}^{(n+1)\times(n+1)},$$

where $\mathbf{c} \in \mathbb{R}^{1\times n}$ is a vector with the elements corresponding to source nodes being 1 and the remaining elements being $1-\alpha$. The reason of introducing α is as follows: Each transient node has to secure a positive value in $\mathbf{c}$ to ensure the final absorption into v_{n+1} for a valid absorbing Markov chain. This naturally introduces α to down-weight P to αP.

Let us pause for a moment and recall that in our context, labels of the source nodes (if there are any) are usually known and are to be propagated to the remaining nodes including the leaf nodes. Edges of the input graph however flows from source to leaf nodes, as exemplified in Fig. 1(b) for $\tilde{P}$. It is more informative to *reverse* all the edge directions, to compute instead the affinity of each unlabeled node toward the labeled nodes (where many are source nodes). As presented in Fig. 1(c), algebraically this corresponds to the matrix *transpose*, $\tilde{Q} := \tilde{P}^T$. Surprisingly, now this row-stochastic transition probability matrix

$$\tilde{Q} \quad = \quad \left(\begin{array}{c|c} \alpha P^T & \mathbf{c}^T \\ \hline 0 & 1 \end{array}\right) = [\tilde{q}_{ij}] \qquad (1)$$

2. Usually source nodes are within $\mathcal{V}_l$, i.e. they are labeled nodes.



(a) P (b) $\tilde{P}$ (c) $\tilde{Q} = \tilde{P}^T$

Fig. 1: (a) An illustrative example of a weighted digraph-based transduction setting: Two different class labels are to be propagated from the labeled nodes (the green and the red nodes, each for one class) to other nodes following the graph structure. Here only a subset of the graph nodes and edges are displayed. Quantities such as W and P can be computed accordingly. (b) As some nodes might have zero in-degree (i.e. source nodes), a new node is further added with directed edges to every nodes including itself, which gives $\tilde{P}$. According to the $\mathbf{c}$ vector, the edges toward those previous source nodes are weighted by 1, and other edges are weighted by $1 - \alpha$. (c) Its transpose, $\tilde{Q}$, corresponds to the same graph but with edge directions being reversed. This facilitates the evaluation of affinity scores flowing from unlabeled nodes (e.g. leaf nodes) to the labeled nodes (e.g. source nodes).

defines an *absorbing Markov chain* on the augmented digraph, $\tilde{G}$. Considering random walks in the Markov chain theory [23], each node v_i of the digraph is equivalent to a Markov chain state s_i, and collectively, the set of nodes naturally identifies a set of states $S = \{s_1, s_2, \ldots, s_{n+1}\}$ in the Markov chain induced by the graph $\tilde{G}$ of $n+1$ nodes. In what follows node v and state s are sometimes used interchangeably. In addition, s_{n+1} is the only absorbing state, while $S_t = \{s_1, s_2, \ldots, s_n\}$ denotes the set of transient or non-absorbing states connecting to s_{n+1} by at least one path. Here each entry $\tilde{q}_{ij}$ denotes the transition probability from state s_i to state s_j with $s_i, s_j \in S_t$. It is also known that within a finite number of steps, a particle in state s_i moving randomly by $\tilde{Q}$ will be absorbed into s_{n+1} with probability 1.

Let us inspect further the upper left block of the matrix $\tilde{Q}$ in (1), denote $Q = \alpha P^T = [q_{ij}]$, and I an identity matrix, both of size $n \times n$. From Markov chain theory [23] we know every element of $Q^t = \underbrace{Q \ldots Q}_{t}$ denotes the probability of a particle starting from s_i to visit s_j in t steps. The expected number of visits from s_i to s_j ($s_i \to s_j$) in t steps is $e_t(s_i \to s_j) = \sum_{k=0}^{t} q_{ij}^{(k)}$, or in its matrix form

$$E_t = I + Q + Q^2 + \ldots + Q^t. \tag{2}$$

Proposition 1. The fundamental matrix of the absorbing Markov chain $\tilde{Q}$ is

$$E = (I - \alpha P^T)^{-1} = [e_{ij}]. \tag{3}$$

The detailed proof is described in Appendix A. In what follows, we present a transductive learning algorithm based on the fundamental matrix of the above absorbing Markov chain $\tilde{Q}$.

2.1.1 Our Algorithm Maximizes the Accumulated Expected Number of Visits

An important fact [23] about the fundamental matrix E of our Markov chain $\tilde{Q}$ is that its (i, j)-th entry e_{ij} provides the *expected number of times* a particle from a transient state s_i visits the transient state s_j. This provides a notion of affinity from state i to j. The intuition is, if a state j is close to the initial state i in terms of graph structure, it will be visited by the particle more often than if it is far away from initial state (We visit our close relatives more often than our distant ones). Now define the affinity matrix as

$$A = EY = (I - \alpha P^T)^{-1} Y. \tag{4}$$

It is a matrix of size $n \times K$, with each entry a_{ik} being associated with an affinity score of state i belonging to class k. In other words, it is the accumulated expected number of visits from state v_i to those states in $\mathcal{V}_l$ that are labeled with class k. Here $\alpha \in (0, 1)$ acts as a parameter which controls how long the random walker stays among the transient states before it gets absorbed. If α is closer to 1 then the random walker stays for a longer period of time before getting absorbed, and vice versa. Empirically, it is observed that our approach is insensitive to varying α values to anything between .01 and .99. We set $\alpha = 0.1$ throughout the experiments. To infer $\mathbf{y}_u^*$ of the unlabeled states $\mathcal{V}_u$, our algorithm predicts each entry's class assignment by identifying a label with the largest affinity score, namely

$$\mathrm{y}_i^* = \arg\max_k a_{ik}, \qquad \forall v_i \in \mathcal{V}_u. \tag{5}$$

2.1.2 Multi-label Classification

With slight modification our approach is also able to work with multi-label classification. That is, starting with a few nodes of the input graph being labeled, to predict multiple target labels for each of the remaining nodes. Instead of Y, we consider a matrix $\tilde{Y}$ of size $n \times \tilde{K}$ for the input label matrix. Here, each column of $\tilde{Y}$ is for one of the $\tilde{K}$ labels, and each entry contains $\tilde{Y}_{ik} = 1$ if the i-th instance is positive for label k, -1 if it is a negative instance, or 0 if it is unlabeled. Instead of $\mathbf{y}^*$, define $\tilde{Y}^*$ the prediction matrix of size $n \times \tilde{K}$. To infer the row vectors $\tilde{Y}_{u:}^*$ of the unlabeled states $\mathcal{V}_u$ from the incurred affinity matrix $A = [a_{ik}]$, we replace (5) with the following one: $\forall v_i \in \mathcal{V}_u, k \in \{1, \cdots, K\}$, $\tilde{y}_{ik}^* = 1$ if $a_{ik} > 0$, and $\tilde{y}_{ik}^* = -1$ otherwise. In other words, a particular entry is assigned positive, if its accumulative expected number of visits to positively labeled instances is more than that to those negative ones, and vice versa. The same procedure can be carried on over all labels.

Algorithm 1 Transduction by Random Walks on Directed Graphs

Input: A digraph $G = (\mathcal{V}, \mathcal{E}, W)$, label information Y, $\mathbf{y}_l$, and $\alpha \in (0, 1)$.
Output: $\mathbf{y}_u^*$
Compute the in-degree matrix D.
Compute the transition probability matrix $P = WD^{-1}$.
Compute the affinity matrix A by (4).
Produce prediction $\mathbf{y}_u^*$. The i-th entry is computed by (5), for an unlabeled node $v_i \in \mathcal{V}_u$.

2.1.3 One-step Increment/Decrement Update (aka Online Update)

In a dynamic graph setting, over the time its graph weights or even structure might subject to changes, being either inserting or deleting edges or nodes of the graph, or merely adjustments of the edge weights. Note the node insertion case corresponds to the out-of-sample instance scenario. These operations can all be accomplished by one-step increment/decrement edge update. In what follow we present a simple $O(n)$ procedure to perform such update in our context. Consider G (or G') being a digraph (its updated digraph) with transient submatrix $Q = \alpha P^T$ (Q'), and fundamental matrix E (E'), respectively. Our aim is to efficiently update E in the following three cases: (1) Delete an edge or decrease an edge weight, $\Delta q_{ij} < 0$; (2) Add an edge or increase an edge weight, $\Delta q_{ij} > 0$, and $\Delta q_{ij} e_{ji} \neq 1$; (3) Add a new node with its edges. Furthermore, the matrix Q has the property that the summation of each row equals either α or 0. In this case, a change of weight entry q_{ij} will lead to changes in the entire i-th row of Q. Fortunately, as described in Proposition 2 below, the cases described above can all be addressed, once we establish a mean to update E' by Proposition 2(i) focusing on the change of only a single entry q_{ij} between Q and Q':

Proposition 2. (i) Suppose for an arbitrary entry q_{ij}, the amount of change, Δq_{ij}, satisfies $|\Delta q_{ij}| \leq q_{ij}$ if $\Delta q_{ij} < 0$, and $\Delta q_{ij} e_{ji} \neq 1$ otherwise. The incurred amount of change in E is

$$\Delta E := E' - E = \frac{\Delta q_{ij}}{1 - \Delta q_{ij}\, e_{ji}} E_{:i}\, E_{j:} \tag{6}$$

where $E_{:i}$ and $E_{j:}$ denote the i-th column and j-th row of E, respectively.
(ii) Consider the changes in the entire i-th row of Q, and assume the amount of change in each entry satisfies the condition described above. To update matrix E, we can either apply (6) n times with each time dealing with one entry change, or equivalently apply the following result:

$$E' = E + \frac{E_{:i}(\Delta Q_{i:}\, E)}{1 - \Delta Q_{i:}\, E_{:i}}, \tag{7}$$

where $\Delta Q_{i:}$ denotes the amount of change in the i-th row of Q.
(iii) Suppose a new node is added to the graph such that the matrix Q becomes $Q' = \begin{bmatrix} Q & u \\ v^T & q \end{bmatrix}$, then the new fundamental matrix E' is given by

$$E' = \begin{bmatrix} E + \gamma (Eu)(v^T E) & \gamma (Eu) \\ \gamma (v^T E) & \gamma \end{bmatrix}, \tag{8}$$

where $\gamma = \frac{1}{(1-q) - v^T E u}$.

The proof is detailed in Appendix A. Notice the imposed condition of $\Delta q_{ij} e_{ji} \neq 1$ in (i) for adding an edge is to guarantee that E is well-defined. Empirically online updates are shown to produce the same results as the batch update counterpart (i.e. our normal algorithm), with negligible entry-wise difference (on the order of 10^{-10}) but with an order of magnitude speedup.

2.1.4 Properties of E

We present here several interesting properties regarding the fundamental matrix E, a central element in our approach.

- **Nonnegativity.** For a digraph, elements of its fundamental matrix satisfies $e_{ij} \geq 0$, $1 \leq i, j \leq n$.
- **Edge reversal property.** By simply reversing all the edge directions of a digraph with a fundamental matrix E, the corresponding new fundamental matrix is equal to E^T.
- **Connectivity and Transitivity.** (i) For any edge $(i, j) \in \{1, \cdots, n\}$ in a digraph, $e_{ij} > 0$ iff there is at least one path form v_i to v_j; (ii) For any $i, j, k \in \{1, \cdots, n\}$, if $e_{ij} > 0$ and $e_{jk} > 0$, then $e_{ik} > 0$. As boundary condition we assume there is one path of length 0 from any node to itself.

Moreover, if $\max_{i,j} q_{ij} < \frac{1}{n}$, the following properties are also true:

- **Diagonal dominance.** For $i, j \in \{1, \cdots, n\}$ of a digraph with $i \neq j$, $e_{ii} > \max\{e_{ij}, e_{ji}\}$.
- **Triangular inequality.** For $i, j, k \in \{1, \cdots, n\}$ of a digraph with $j \neq i$ and $k \neq i$, $e_{ii} \geq \max\{e_{ij} + e_{ik} - e_{jk}, e_{ji} + e_{ki} - e_{kj}\}$.
- **Transit inequality.** For distinct indices $i, j, k \in \{1, \cdots, n\}$ of a digraph, if there exists a path from v_i to v_k and each path from v_i to v_t includes v_k, then $e_{ik} > e_{it}$.
- **Monotonicity.** Suppose the entry q_{kt} concerning the edge from v_k to v_t is increased by $\Delta q_{kt} > 0$. Then
  (i) $\Delta e_{kt} > 0$, and for any $i, j \in \{1, \cdots, n\}$ such that $i \neq k$ or $j \neq t$ we have $\Delta e_{kt} > \Delta e_{ij}$;
  (ii) for any $i \in \{1, \cdots, n\}$, if there is a path from v_i to v_k, then $\Delta e_{it} > \Delta e_{ik}$.

Detailed proofs are presented in the supplementary file due to space limit, which are adapted from the proofs in [27] tackling proximity measures in a more general setting. Although the above properties are well-established as proximity measures between vertices of graphs, to our best knowledge, many of them are not shown before for the fundamental matrix of a absorbing Markov chain.

2.2 Connections to Existing Methods

Graph Kernels: Algebraically our algorithm is similar to several graph kernels, including the Von Neumann Kernel $K_{VND} = \sum_{k=0}^{\infty} \alpha^k W^k = (I - \alpha W)^{-1}$ [28], and the regularized commute time kernel $K_{RCT} = (D - \alpha W)^{-1}$ [15], [11], [29]. These kernel functions are however constructed specifically from undirected graphs (i.e. within the cone of symmetric positive definite matrices) and based on considerably different motivations and derivations.

PageRank and Digraph Laplacian: Our approach is also related to PageRank [25], which resolves the issue of source nodes by teleporting random walks that introduce bi-directional edges to all node pairs with equal weights, i.e. $\bar{P} = (1-\eta)P + \frac{\eta}{n}\mathbf{e}\mathbf{e}^T$ with $\mathbf{e}$ a $n \times 1$ vector of all ones, and η a tiny positive real. A very similar idea is also used by the closely related work [19] based on digraph Laplacian [24]. They are very different from our approach. First, both operate on graphs with *irreducible* Markov chains rather than the *absorbing* Markov chains considered in our context. By definition irreducibility requires each node can be reached from any other node, i.e. a strongly connected graph – algebraically this often gives rise to a dense matrix, as shown in the teleporting operation. Second, as side-effects of introducing the teleporting operation, the input graph structure is not well preserved, and weak edge signals also tend to be washed away. In contrast our approach is able to retain the input graph structure as well as weak signals.

Graph Laplacian in Undirected Graphs: Our algorithm also works with *undirected* graphs as a special case (i.e. equivalent to bi-directional edges with equal weights). An interesting observation is that here our algorithm can be shown as a scaled variant of the graph Laplacian based method in [15], which has been specifically developed for undirected graphs. This is discussed in details in Appendix A.

Partially Absorbing Random Walks (PARW) [17]: It can be shown that the absorbing Markov chains considered in our context is quite general: The random walks of [17] correspond to a very special kind of such absorbing Markov chains where the submatrix of W concerning transient nodes forms a symmetric non-negative matrix. In other words, the transient nodes are interconnected with undirected edges, while the edges from transient to absorbing nodes are still directed. Details are relegated to Appendix A.

Spanning Forest of Digraphs [30]: The celebrated Matrix-Tree theorem has been extended to general digraphs [31], where the quantity $\mathcal{Q} := \left(I + \tau L\right)^{-1}$ with $L := D - W$ is shown to be the normalized counts of spanning out-forests. It turns out $\mathcal{Q}$ is a scaled version of $(I - \alpha P^T)^{-1}$, the central piece of our approach. Details are relegated to the supplementary file due to space limit.

2.3 Analysis of Algorithm 1

Computational complexity: The complexity of Algorithm 1 is dominated by the cost of computing the affinity matrix A in (4), which can be accomplished by solving the following linear system

$$(I - \alpha P^T)A = Y.$$

For a general dense matrix P, the computational time is $O(n^3 + n^2 K)$. This is e.g. about the same complexity of [19], one of our main competing methods. Fortunately, P is usually a sparse matrix in our context, which can be exploited to reduce the computational time. There are many efficient solvers for large sparse linear systems, including both direct [32], [33] and iterative methods [34], [35]. In our implementation, we adopt the direct solver UMFPACK [32] which exists as a built-in routine (for LU, backslash, and forward slash functions) in MATLAB. The specific complexity depends on the size (n), the number of non-zero entries and the sparsity pattern of P, which remains a challenging task to provide a tighter complexity measure dedicated to our context. Nevertheless, our approach is practically much more efficient comparing to state-of-the-art methods including [19], as is empirically verified in experiments.

Error Bound Based on Transductive Rademacher Complexity: A data-dependent generalization error bound is provided for the proposed algorithm, where we focus on the binary-class case for the sake of simplicity. The bound provided by our analysis is built on top of the work of [36] on transductive Rademacher complexity.

We start by reformulating our algorithm (4) as an equivalent representation

$$\mathbf{h} = E\hat{\mathbf{y}} = (I - \alpha P^T)^{-1}\hat{\mathbf{y}}, \tag{9}$$

where $\hat{\mathbf{y}}$ is the initial label vector with partial labels $\hat{\mathrm{y}}_i \in \{\pm 1\}$ for $v_i \in \mathcal{V}_l$, and $\hat{\mathrm{y}}_i = 0$ otherwise. The obtained $\mathbf{h}$ is the "soft" label vector with h_i being the "soft" label for node v_i, which will be assigned with class label $\mathrm{sign}(\mathrm{h}_i)$ when making predictions [3]. We denote by $\mathcal{H}_{out}$ the set of feasible soft label vectors generated by our algorithm (9). Since there are l labeled nodes, it follows that

$$\mathcal{H}_{out} \subseteq \mathcal{H} := \left\{ \mathbf{h} \;\middle|\; \mathbf{h} = (I - \alpha P^T)^{-1}\hat{\mathbf{y}}, \|\hat{\mathbf{y}}\|_2 \leq \sqrt{l} \right\}, \tag{10}$$

which naturally admits a *vanilla unlabeled-labelled representation* proposed in [36]. We proceed with the definition of transductive Rademacher complexity.

Definition 3. [36] Let $\mathcal{F} \subseteq \mathbb{R}^n$ and $p \in [0, 1/2]$. The transductive Rademacher complexity of $\mathcal{F}$ with parameter p is defined as

$$R_{l,n}(\mathcal{F}, p) := \left(\frac{1}{l} + \frac{1}{n-l}\right) \mathbb{E}_{\boldsymbol{\sigma}} \left[\sup_{\mathbf{f} \in \mathcal{F}} \boldsymbol{\sigma}^T \mathbf{f} \right], \tag{11}$$

where $\boldsymbol{\sigma} = (\sigma_1, \cdots, \sigma_n)^T$ is a vector of i.i.d. random variables such that

$$\sigma_i := \begin{cases} 1, & \text{with probability } p, \\ -1, & \text{with probability } p, \\ 0, & \text{with probability } 1 - 2p. \end{cases} \tag{12}$$

Different from inductive Rademacher complexity [37], the transductive complexity does not depend on any underlying distribution. Besides, for any label vector $\mathbf{h}$, define the *test error* as $\mathcal{L}_{l,n}(\mathbf{h}) := \frac{1}{n-l}\sum_{i=l+1}^{n} \ell(\mathrm{h}_i, \mathrm{y}_i)$ with respect to its $0/1$ loss function ℓ satisfying $\ell(\mathrm{h}_i, \mathrm{y}_i) = 1$ if $\mathrm{h}_i \neq \mathrm{y}_i$ and $\ell(\mathrm{h}_i, \mathrm{y}_i) = 0$ otherwise, and define the *empirical error* of $\mathbf{h}$ as $\hat{\mathcal{L}}_{l,n}(\mathbf{h}) := \frac{1}{l}\sum_{i=1}^{l} \ell(\mathrm{h}_i, \mathrm{y}_i)$. Based on the aforementioned transductive Rademacher complexity, in what follows we present our risk bound and relegate the proof to the supplementary file due to space limit.

Theorem 4. Let $\mathcal{H}_{out}$ be the set of feasible soft label vectors generated by applying (9) to all possible sample set $\{(v_i, \mathrm{y}_i)\}_{i=1}^n$.

3. We should remark that predictions made in this way are exactly the same as the predictions made by Algorithm 1 in the binary case. Let $\mathcal{I}_1$ and $\mathcal{I}_2$ denote the index sets of labeled data from classes 1 and 2, respectively, it follows that $Y_{i1} = 1$ if $i \in \mathcal{I}_1$, $Y_{i2} = 1$ if $i \in \mathcal{I}_2$ and $Y_{ij} = 0$ otherwise; $\hat{\mathbf{y}}_i = 1$ if $i \in \mathcal{I}_1$, $\hat{\mathbf{y}}_i = -1$ if $i \in \mathcal{I}_2$ and $\hat{\mathbf{y}}_i = 0$ otherwise. Then, from equations (4) and (9) we have

$$A = \left[\sum_{i \in \mathcal{I}_1} E_{:i} \;\; \sum_{i \in \mathcal{I}_2} E_{:i} \right] \text{ and } \mathbf{h} = \sum_{i \in \mathcal{I}_1} E_{:i} - \sum_{i \in \mathcal{I}_2} E_{:i},$$

which implies that for $1 \leq i \leq n$

$$\mathrm{sign}(\mathrm{h}_i) = \arg \max_{k \in \{1,2\}} a_{ik}.$$

Let $c_0 := \sqrt{32\ln(4e)/3}$, $q := 1/l + 1/(n-l)$ and $s := \frac{n}{(n-1/2)(1-1/(2\max(l,n-l)))}$. For any $\delta \in (0,1)$, with probability $1-\delta$ over random draws of sample $\{(v_i, \mathrm{y}_i)\}_{i=1}^n$, for all $\mathbf{h} \in \mathcal{H}_{out}$,

$$\mathcal{L}_{l,n}(\mathbf{h}) \leq \hat{\mathcal{L}}_{l,n}(\mathbf{h}) + \sqrt{\frac{2l}{n(n-l)}\|(I-\alpha P^T)^{-1}\|_F^2} + c_0 q\sqrt{\min(l,n-l)} + \sqrt{\frac{sq}{2}\ln\frac{1}{\delta}}, \quad (13)$$

It is easy to see that when $l \to \infty$ and $(n-l) \to \infty$, $s \to 1$. Then the convergence rate is determined by the slack terms $c_0 q\sqrt{\min(l,n-l)} + \sqrt{\frac{sq}{2}\ln\frac{1}{\delta}}$, which is in the order of $O\Big(\frac{1}{\sqrt{\min(l,n-l)}}\Big)$. So far we provide an transductive Rademacher bound for the binary scenario. In addition, a transductive bound based on PAC-Bayes is also provided in what follows for general multiclass setting where binary classification can be regarded as a special case. It is known that the first bound is tighter but more focused on binary classification, while the PAC-Bayes bound is more general.

PAC-Bayesian Transduction Bound: In this part we present a PAC-Bayesian bound for Algorithm 1. The error bound presented in the next theorem is an adaptation of the PAC-Bayesian bound for general transductive learning developed in [38]. It mainly shows that the test error can be upper bounded by the empirical error plus some complexity term. Due to space constraint, the proof is in the supplementary file.

Theorem 5. Let $\hat{\mathcal{L}}_{l,n}(\mathbf{h})$ and $\mathcal{L}_{l,n}(\mathbf{h})$ be the empirical error and test error, respectively, defined the same as in the previous section with respect to the $0/1$ loss function ℓ. Then, for any deterministic classifier $\mathbf{h}$ determined by our Algorithm 1 and any $\delta \in (0,1)$, with probability at lest $1-\delta$ over random draws of $\mathcal{V}_l$ from $\mathcal{V}$, the following bound holds

$$\mathcal{L}_{l,n}(\mathbf{h}) \leq \hat{\mathcal{L}}_{l,n}(\mathbf{h}) + \sqrt{\frac{1}{2(1-\frac{l}{n})}\left(\ln\frac{n}{l} + \frac{1}{l}\ln\frac{C(l,n)}{\delta} + \ln(Ke)\right)}, \quad (14)$$

where $C(l,n) = (\sqrt{2}\ln l + 8)\sqrt{l(1-\frac{l}{n})}$ and e is the base of the natural logarithm.

Notice that when $\beta_l \leq l/n \leq \beta_u$ for any constant β_l, β_u satisfying $0 < \beta_l \leq \beta_u < 1$ (e.g., $l/n = \beta_0$ the number of labelled sample is a constant proportion of the number of full sample), the complexity term (i.e., the second term on the right-hand side of (14)) converges to $\sqrt{\frac{\ln(\frac{Ke}{\beta_l})}{2(1-\beta_u)}}$ as $n \to \infty$, which means that in such a case the test error $\mathcal{L}_{l,n}(\mathbf{h})$ will not excess the empirical error $\hat{\mathcal{L}}_{l,n}(\mathbf{h})$ by a constant value.

3 EXPERIMENTS

Our approach is empirically evaluated in various applications, including citation problems, UCI datasets, social network problems [39], as well as a relatively unconventional problem: the retinal blood vessel tracing problem. For the citation problem three datasets are employed: CoRA [9], CiteseerX [9], and US Patent [40]. We also conduct experiments on three UCI datasets: COIL20 [41], TDT2 [4], and 20Newsgroups [42]. For social network we consider the Google+ and the Twitter datasets of [39], where the goal is to identify the social circles of individual users. Finally, the tracing problem involves three datasets: a synthetic dataset, as well as DRIVE [3] and STARE [1]. Our approach is also compared with 12 state-of-the-art methods that directly work with directed graphs:

- Network-only Bayes Classifier (NBC) [43].
- Network-only Link Based classifier (NLB) [44].
- Class Distribution Relational Neighbor classifier (CDRN) [9].
- Weighted Vote Relational Neighbor classifier (WVRN) [9].
- Digraph variant of the Commute Time Kernel classifier (CTKd) [11].
- Regularized Commute Time Kernel classifier (RCTKd) [11].
- Symmetrized Graph Laplacian (SGL) [19].
- Zero-mode Free Laplacian (ZFL) [10].
- Sum Over Path covariance kernel (SOP) [20].
- Biased Discriminative random Walks (bDWalk) [45], [46].
- Bounded normalized random walk with restart (bNRWR) [45].
- Approximate normalized, regularized, Laplacian (aNRL) [45].

Out of these methods, four (NBC, NLB, CDRN, and WVRN) are implemented by NetKit [9] in Java, SOP [20] is obtained from the authors, while other methods (CTKd, RCTKd, SGL, ZFL, bDWALK, bNRWR, aNRL, and Ours) are implemented by ourselves in MATLAB. Note that the original Commute Time Kernel classifier (or CTKu) only works with undirected graphs. To work with digraphs, we instead replace its original undirected graph Laplacian with the symmetrized digraph Laplacian of [24]. As a result, this variant is referred to as CTKd in this paper. To ensure fair evaluations, the internal parameters of the comparison methods are either set to as is from the authors' original source code, or as suggested by their respective authors: For example, according to [19], the regularization parameter is set to 0.1 and the jumping factor used in teleporting random walk is set to 0.01 for SGL. In particular, for the four methods implemented by NetKit, the uniform local-classifier is used as the "local" model, and for collective inference relaxation labeling has been used for NBC, CDRN, and WVRN, while iterative classification is used for NLB. This setting is reported to deliver the best performance in [9]. For each of bNRWR and aNRL, there are two tuning parameters $\alpha \in (0,1)$ denoting the probability that the random walker continues the walk and $\tau \in \mathbb{N}$ denoting the maximum walk length. Following [45], we use 5-fold cross-validation to search the optimal parameters on the grid $\{0.1, 0.2, \cdots, 0.9\} \times \{2^1, 2^2, \cdots, 2^5\}$ on all datasets except US patent, where we set $\alpha = 0.1$ and $\tau = 4$. [5]

In term of evaluation metric, the micro-averaged accuracy (AC) [47] is adopted in most of the experiments as the accuracy

4. NIST Topic Detection and Tracking corpus at http://www.nist.gov/speech/tests/tdt/tdt98/index.html.

5. In [45], the authors only mentioned that they used cross-validation to select optimal parameters without providing candidate values for each parameters. The candidate sets used in our experiments are based on the fact that $\alpha \in (0,1)$ and observation that both bNRWR and aNRL converges within 10 iterations. On US patent dataset, since it takes prohibitively long time for bNRWR and aNRL to conduct cross-validation, we use the parameters where both methods perform well on other datasets.

TABLE 1. Averaged CPU time (seconds) for all competing methods. "–" denotes the cases where the method either fails to compute a solution due to out of memory or takes too long time to compute.

	NBC	NLB	CDRN	WVRN	CTKd	RCTKd	SGL	ZFL	SOP	bDWALK	bNRWR	aNRL	Ours
CoRA	1.01	1.25	2.90	0.91	2.81	1.31	2.84	1.98	2.84e+4	2.84e+4	2.48	2.30	**1.11e-2**
CiteseerX	1.04	1.25	3.41	0.96	4.07	1.73	4.13	2.81	6.93e+4	6.93e+4	2.61	2.48	**9.79e-3**
US Patent	1.14e+3	–	1.10e+3	2.65e+2	–	–	–	–	–	–	3.57e+4	3.22e+4	**40.53**
COIL20	3.48	1.95	6.65	1.13	0.36	0.15	0.35	0.30	1.76e+3	1.76e+3	3.28	2.61	**1.52e-2**
TDT2	1.60e+2	–	2.85e+2	3.95	50.71	12.99	52.10	42.44	–	–	87.40	64.01	**0.35**
20Newsgroups	15.72	79.95	10.65	2.96	1.99e+2	42.07	2.08e+2	1.92e+2	–	–	1.05e+2	92.13	**2.38**

measure, which is the sum of all true positive counts divided by the total number of instances. For the social network applications where there is a need to evaluate the partial correctness of predicted labels in the multi-label setting, a modified version of F_1 score [48] is used:

$$F_1 = \frac{1}{nk} \sum_{i,k} \frac{2|\tilde{y}^*_{ik} \bigcap z_{ik}|}{|\tilde{y}^*_{ik}| + |z_{ik}|} \tag{15}$$

where $\tilde{y}^*_{ik}$ and z_{ik} are predicted and true labels for the k-th label of the i-th instance. For the vessel tracing problem, we employ the DIADEM score (DS) [49] instead, which is a dedicated measure widely used by the biological tracing community. An example is provided in the supplementary file to illustrate how the DIADEM score is computed.

3.1 Citation Problem

Paper citations naturally form a digraph and the aim here is to predict a prescribed topic for each of the unlabeled papers at hand, provided a few are labeled a priori. We first conduct evaluations on CoRA [9] and CiteseerX [50]. The CoRA dataset contains a citation digraph of 2,708 nodes and 5,429 directed links (edges) on computer science research papers spanning 7 topics. CiteseerX is another citation dataset of 3,312 papers and 4,732 citations (directed edges) from 6 categories. We also examine our approach on a *large-scale* dataset, US Patent [40], which consists of 13 million directed edges connecting 2.7 million nodes that can be categorized into 418 distinct topics. For all three datasets, the adjacency matrices are adopted as their corresponding weight matrices.

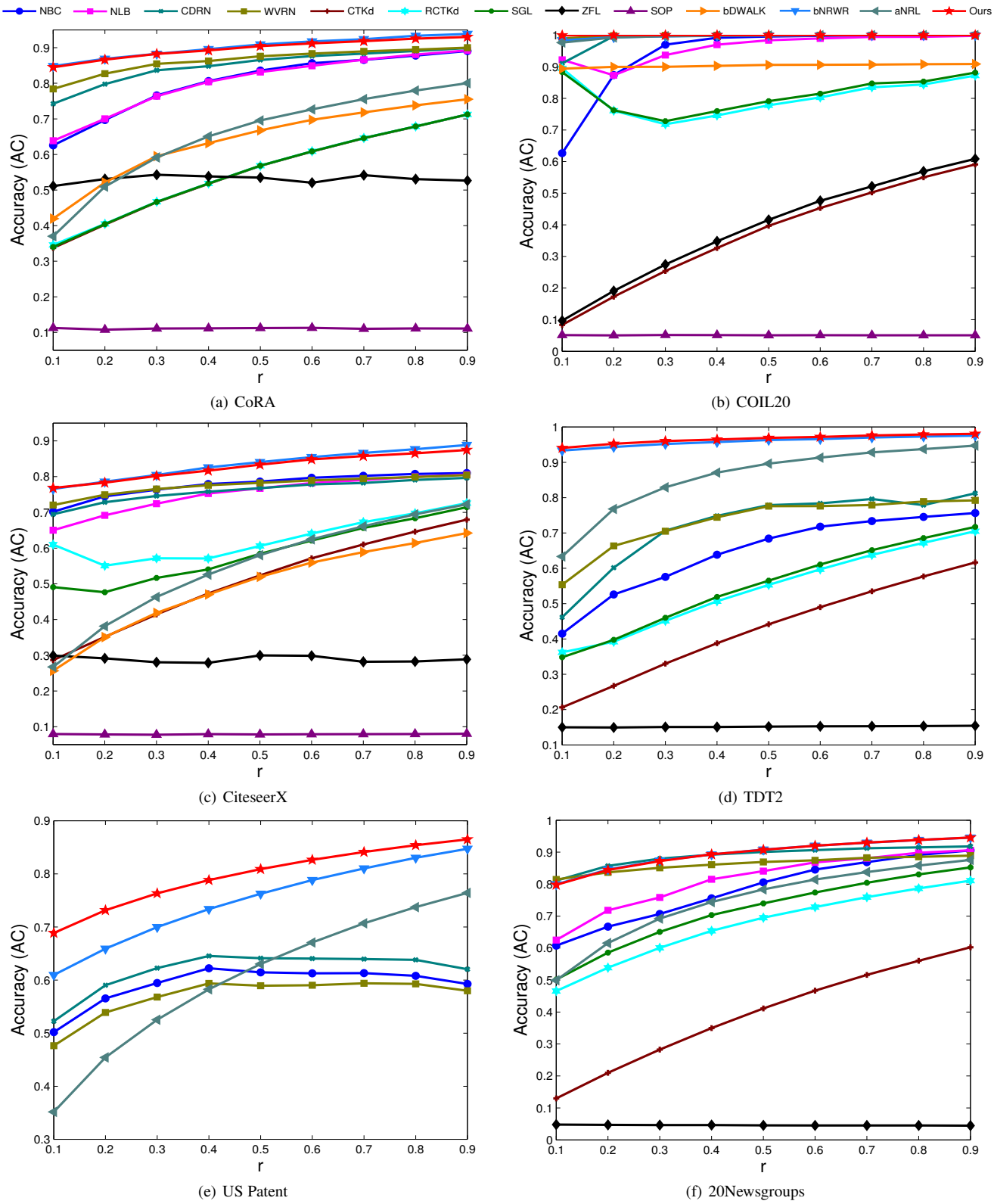
To evaluate the system performance against varying size of labeled nodes in digraphs, the following strategy is adopted: For each of the K classes, certain percentage (i.e. label ratio, also denoted as r) of instances (i.e. nodes) in this class is uniformly selected as labeled nodes – this gives one empirical data sample. This procedure is repeated 50 times to produce an averaged performance estimate. We then vary the label ratio r from 10% to 90% with 10% increment, and compare the averaged performance (AC) of competing methods as presented in the left column of Fig. 2. Note that during these experiments, when the labeled nodes are selected, the nodes with "zero-knowledge" components [9] will be temporarily removed from consideration, as these nodes that have no directed path connecting to any node in $\mathcal{V}_l$. Moreover, in TABLE 1 we present the averaged CPU time for each of the competing methods: For US Patent dataset, the timing is averaged over single runs of different label ratios, while for CoRA and CiteseerX, the timing is instead averaged over all runs and all label ratios, where there are 50 runs for each specific label ratio.

From the left column of Fig. 2, we observe that for CoRA and CiteseerX datasets, our approach and bNRWR consistently outperforms the rest of competitors, and both approaches achieve quite close accuracy. To clearly see the difference between bNRWR and our approach, we included a zoom-in figure of Fig. 2 in the supplementary file, where we see that bNRWR has slightly higher accuracy than our approach. One reason is that bNRWR employs cross-validation to select the optimal tuning parameter while our approach does not. However, cross-validation makes bNRWR much slower than our approach, as can be seen in TABLE 1. For the remaining methods, WVRN becomes the third best method, which is followed by CDRN and others, while CTKd, ZFL and SOP often produce the least favorable results. For US patent dataset our approach performs consistently the best and with a very significant advantage comparing to the others across different labeling ratios. Note the performance of SGL, a closely related method of ours, is almost at the lower end of the middle regime of performers. We attribute this to the fact that both CoRA and CitesserX are not very dense digraphs and their edge weights are quite asymmetric, which seems to be difficult for SGL, as source information is not well kept after utilizing teleporting Markov chain as well as the symmetrized graph Laplacian [24]. Our results are also aligned with existing evaluations [6], although the results are not directly comparable due to the randomized nature when sampling instances for each of the label ratios. In term of CPU time as in TABLE 1, our approach consumes significantly less time comparing to other methods, and is with a significant gap from the second best, WVRN. On the flip side, bDWALK and SOP are the most computational intensive of all, which is closely followed by SGL. In particular, our approach is shown to be very efficient when working with the large-scale US patent dataset, where it takes merely around 40 seconds for our approach to make predictions on this million-node dataset using a standard desktop.

3.2 kNN Graphs on UCI Datasets

The kNN graphs are often used in practical semi-supervised learning tasks. As they are asymmetric by nature, they can be regarded as digraphs. Therefore, we also evaluate our approach on kNN graphs constructed from the well-known UCI datasets. Directed edges of the kNN graphs are obtained as follows: There is an edge from node x_i to node x_j if and only if x_j are among the $k = 5$ nearest neighbors of x_i. The weight of the assigned edge is given by $W_{ij} = \exp\left(\frac{-\|x_i - x_j\|_2}{2}\right)$. Three directed graphs are thus obtained from three UCI datasets as follows: The kNN graph of COIL20 consists of 1,440 nodes from 20 classes and 7,200 edges. The kNN graph of TDT2 contains 10,212 nodes from 96 classes and 49,495 edges. The kNN graph of 20Newsgroups includes 18,846 nodes from 20 classes and 91,690 edges. Similar to the previous subsection, we evaluate all methods on the above kNN graphs with label ratio r varying from 10% to 90%. Experimental results are presented in TABLE 1 and the second column of Fig. 2.

6. E.g. Fig. 6 of [9] on CoRA where the best performer delivers around 0.8–0.9 by varying the label ratios.



(a) CoRA (b) COIL20

(c) CiteseerX (d) TDT2

(e) US Patent (f) 20Newsgroups

Fig. 2: Accuracy comparisons on three UCI datasets as well as the citation benchmarks including CoRA, CiteseerX, and US Patent. Here the micro-averaged accuracy (AC) is adopted as the evaluation metric. The first column presents results on CoRA, CiteseerX, and US Patent, while the second column shows results on UCI datasets COIL20, TDT2, and 20Newsgroups. In all plots, the horizontal axis denotes the label ratio (percentage of labeled nodes) varying from 10% to 90% with 10% increment. See text for details.

Our approach again achieves the best accuracy and outperforms the competitors by a large margin in terms of computational time, which is consistent with what we have already observed for the citation problem.

3.3 Social Network Application

It is a non-trivial task to identify social circles in social networks. Usually such a problem involves many different labels (circles) and is of large size. In particular, we consider the problem of

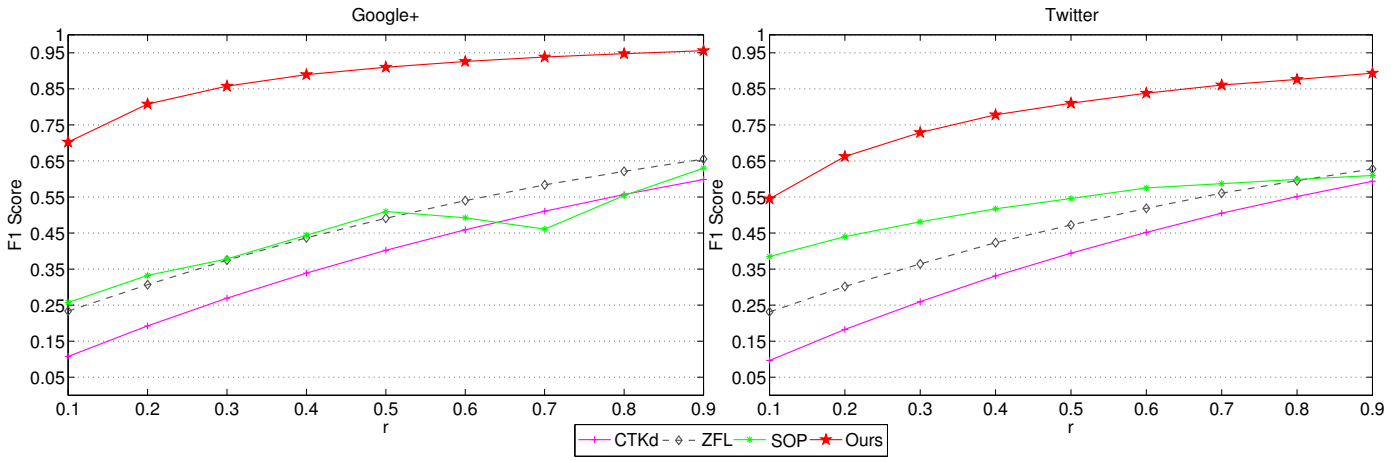


Fig. 3: Comparisons of F1-Score on the multi-label problem of Google+ and Twitter datasets. In both plots, the horizontal axis denotes the label ratio (percentage of labeled nodes) varying from 10% to 90% with 10% increment. See text for details.

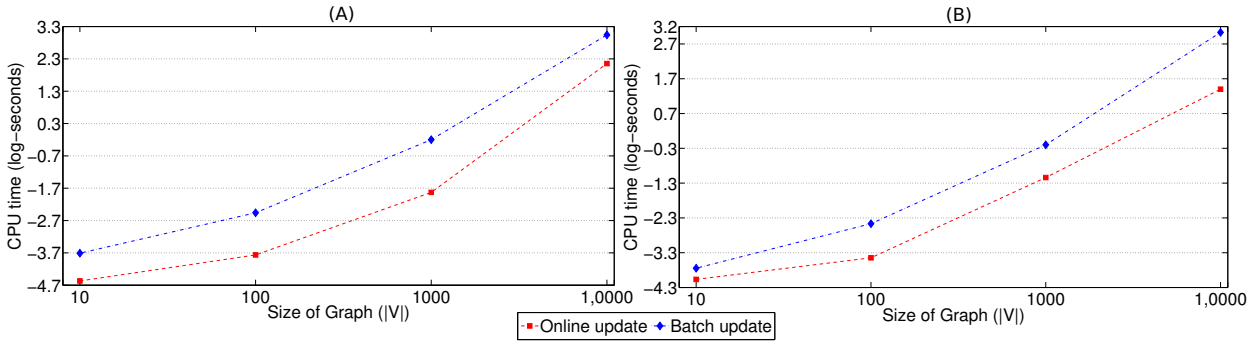


Fig. 4: Empirical time-complexity of batch update (3) vs. online updates (7) and (8) of the fundamental matrix E. (A) and (B) show the CPU-time (log-seconds) of batch update vs. online update for changing one row using (7), and for inserting / deleting a new node with (8), respectively. See text for details.

identifying 327 social circles in the Google+ dataset, and 3,127 social circles in the Twitter dataset. Both datasets are from [39]. The Google+ dataset consists of a graph with 1.4 million nodes and 30 million directed edges belonging to 133 users. As only part of the nodes have ground-truth labels, those nodes with no label information are trimmed away – we are thus left with 19,327 nodes and 3,294,465 directed edges. Similarly, the Twitter dataset has 81,306 nodes and 2.4 million directed edges from 1,000 users. After removing nodes with no label information, we obtain a digraph with 19,270 nodes and 490,667 directed edges. For experimental evaluation, the label ratio r is varied from 10% to 90% with 10% increment, and the F_1 score in (15) is computed. NBC, NLB, CDRN and WVRN are only able to work with single-label classification problem. In the meantime, the teleporting random walks introduced in SGL tends to wash away weak signals, which seems to significantly deteriorate the performance over all label ratios. As a result, our approach are compared with three methods: CTKd, ZFL and SOP, as presented in Fig. 3. Our approach clearly outperforms other three state-of-the-arts by a very large margin in both datasets. For the Google+ dataset, ours produces a series of increasing F1-scores of 0.7–0.95 with the increment of label ratio r, where ZFL and SOP are the best runner-ups with combined best performance of merely 0.25–0.65 during the same range of r. CTKd seems to perform least well. These phenomena are similarly observed for the Twitter dataset. The gap of performance in the comparison methods seems to be attributed to the combined influences of large label size and

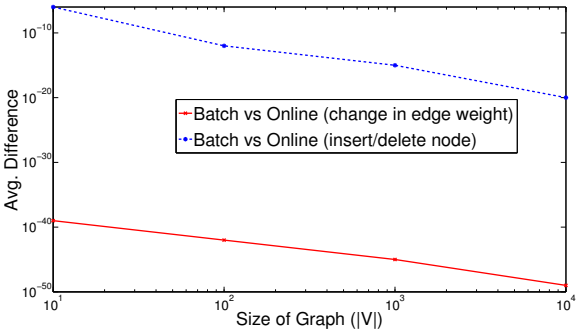
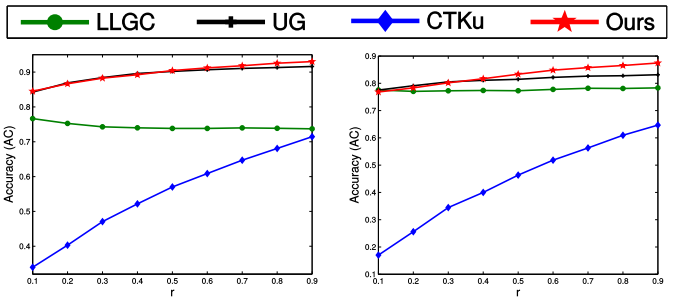


Fig. 5: Averaged absolute difference between online and batch updates. The red and the blue curves show the average difference for changing one row using (7), and for inserting/deleting a new node with (8), respectively.

large data size (In US patent dataset we also observe a rather significant margin between our approach and the best runner-up). The superior performance of our approach, on the other hand, suggests that our approach is particularly reliable when dealing with large-scale graphs with many labels.

3.4 Empirical Time-complexity of Batch vs. Online Updates

Here we focus on the dynamic graph scenario where a small fraction of the digraph structure might change over time, being either changing a single edge weight, or inserting/deleting a single



(a) CoRA (b) CiteseerX

Fig. 6: Comparison with state-of-the-art methods based on undirected graphs.

node. In our context, this boils down to efficient computation of the fundamental matrix E. Our approach is capable of addressing these changes in E, as presented in (7) and (8) for online updates, as well as in (3) for batch update. Ideally, the online updates are expected to be carried out more efficiently and the results should be the same as of batch update. To show this, we design the following synthetic experiments: The weight matrix of a sparse digraph of size n is randomly generated with its E matrix computed. This is followed by either changing a single edge weight, or inserting/deleting a single node from the digraph, which subsequently gives E'. Its online update is then computed by (7) or (8), and the batch update is computed by (3). The above process is repeated 20 times, for each of the following four different digraph sizes, namely $n \in \{10, 100, 1000, 10,000\}$, and the median running time is displayed in Fig. 4(A) and (B). Note that this comparison is not entirely fair as the implementation set-up is less favorable for the online update: To compute (3) for batch update, the MATLAB implementation of UMFPACK direct solver is highly optimized and runs on multiple cores, while our implementation of the online updates, namely (7) and (8) are in MATLAB script as is (without any optimization). Nevertheless, as presented in Fig. 4 the online update runs always an order of magnitude faster. Besides, the numerical difference between batch and online updates is negligible in practice. As displayed in Fig. 5, on average the absolute difference value is always below 10^{-5} in the above mentioned experiments. In addition, this numerical error decreases dramatically with the increase of digraph sizes.

3.5 Comparison with Undirected Graph Based Methods

So far we have compared our approach to a number of methods that can directly work with digraphs. One may still wonder how conventional undirected graph based methods would perform in our context. For this purpose, we compare our approach with two state-of-the-art such methods, namely the Learning with Local and Global Consistency (LLGC) method in [15] and the original Commute Time Kernel classifier (CTKu) [11], both operate on undirected graphs. We also compare with UG, introduced in Section 2.2 and Appendix A, which has been shown to be a special case of our approach when graphs are undirected. The comparison is conducted on CoRA and CiteseerX, where we construct undirected graphs via assigning an edge if there is at least one link between two nodes, regardless of the linking direction. Results are presented in Fig. 6 when the label ratio varies from 0.1 to 0.9. In the figure, 'Ours' denotes results of our approach on the original directed graphs. From Fig. 6, we observe that our approach has the best overall performance, followed by UG and LLGC. In addition, our approach performs better than UG which only work on undirected graphs, especially when r is large. This implies that incorporating directionality of graph into our approach improves the performance. Moreover, our approach outperforms CTKu by a large margin on both datasets, and maintains a clear performance advantage of about 10% over LLGC on CoRA dataset.

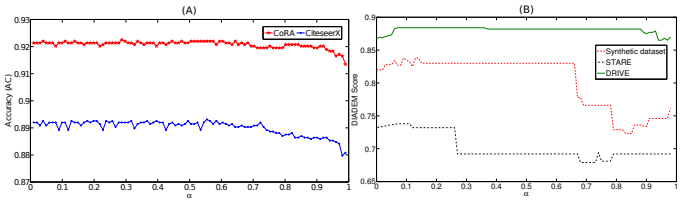


Fig. 7: Robustness of our system vs. changing α values between .01 and .99. (A) For CoRA and CiteseerX, the performance or our system is rather stable (with around .001 variation) when α is within .01 and 0.9, and start to decrease slightly (around .01 variation) when α value goes beyond .9. (B) The performance remains stable in vessel tracing problems.

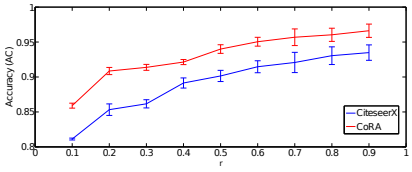


Fig. 8: Robustness of our system with respect to varying label ratio. The error bar of each labeling ratio (r) displays 5 – 95 percentile of accuracy when α values are systematically sampled between .01 and .99 with an .01 increment. The narrow deviations from median as shown in the error bar (usually less than 2%) clearly suggest that our system is rather stable against changes of α values.

3.6 The Effect of α

We also provide empirical analysis to study the effect of varying α value to the performance of the proposed system: As presented in Fig. 7, the empirical performance across a wide range of applications is relatively stable against changing α values, especially during the range of .09 to .25. This observation is further confirmed in Fig. 8 with varying label ratios, where different α values usually result in less than 2% variations in its performance. The insensitive pattern of α is experienced throughout empirical experiments. This motivate us to simply fix α to certain value (0.1) during the rest of experiments. We note in the passing that performance degradation is to be expected when α taking extreme values being too close to either 0 or 1, since which renders E to be too close to either the identity matrix or an ill-conditioned matrix, respectively.

3.7 Retinal Blood-vessel Tracing

In vessel tracing, our approach is evaluated in synthetic datasets [51], as well as two standard testbeds, DRIVE [3] and STARE [1]. The synthetic dataset is constructed in house that contains 17,000 synthesized retinal images with varying densities of retinal blood vessels (which strongly correlate with the frequency of cross-over occurrences among vessel branches). Meanwhile, DRIVE dataset contains 40 retinal fundus images, and STARE has 20 fundus images. Exemplar images of the three datasets are plotted in the first column of Fig. 11. Detailed protocol for creating the synthetic retinal images can be found in [51].

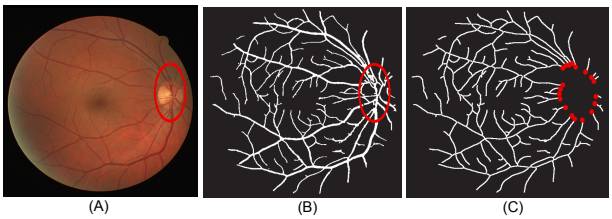


Fig. 9: Preprocessing of retinal blood vessel tracing. (A) An input image from DRIVE. (B) Binary image after segmentation. (C) Image after skeleton extraction and optical disk removal. The red elliptical area in (A) and (B) is the optical disk. The red dots in (C) are tips of the *root segments* identified as those directly contacting the optical disk. Note that each root segment induces a distinct vessel tree from the graph with itself being the tree root, due to the nature of blood flow in vessels.

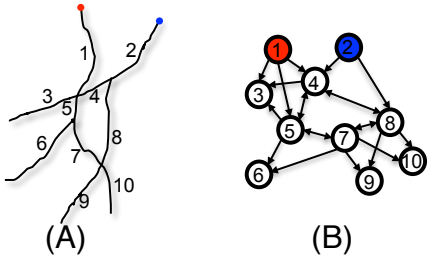


Fig. 10: From skeleton to digraph. (A) A exemplar skeleton map. (B) Its digraph G. The highlighted zone of nodes are shown as an example where the corresponding directed subgraph is formed. The segments marked with red and blue dots at their tips are the root segments, with each being regarded as the labeled node for its class. In other words, each class (corresponds to a vessel tree) has exactly its root node labeled, which corresponds to a source node in graph.

The problem of vessel tracing is to trace blood vessels by separating them into disjoint vessel trees, each starting from a unique root segment within the optical disk. The major difficulty here is to resolve the challenging cross-over issues that are abundant in the retinal datasets. This problem can be cast into a digraph-based transduction problem after the following preprocessing steps:

i) Segmentation: As illustrated in Fig. 9 (A)→(B), an input retinal image is segmented into a binary image, with vessel pixels being foreground and the remaining as background.

ii) Skeleton map: Build a skeleton map from the binary image, and remove the optical disk area as marked within red ellipse in Fig. 9(C). The tips attached to the removed optical disk are the tips of root segments, which are presented as color dots in Fig. 10(A).

iii) Skeleton to digraph: A segment is defined in the skeleton as the group of connected pixels that ends in either a junction or a tip. This segment corresponds to a node in the resulting digraph, as shown in Fig. 10(A)→(B). Two nodes are then linked with a directed edge if the two coinciding segments from the skeleton map contact and satisfy the ordering criteria of [51].

This produces a digraph as shown in Fig. 10(B), where red-colored and blue-colored nodes corresponding to the root segments in skeleton map are labeled with distinct class labels, each for one particular vessel tree. The task is to propagate class labels (tree ids) to unlabeled nodes. As reported in TABLE 2, overall our approach consistently outperforms other methods by a margin. It is followed by ZFL, SGL, and NBC, while WVRN and SOP tend to produce least accurate predictions. ZFL also performs reasonably well on vessel tracing problems, which is however cumbersome when dealing with large matrices, as it requires to work with (and even invert) dense matrices. Note the SGL method here is employed as the learning engine in [51] which is the state-of-the-art in this task. Exemplar images and results are also presented in Fig. 11 for visual inspection. It suggests that empirically our approach delivers visually plausible tracing results when compared to the ground-truths side-by-side, and errors occur at those challenging spots that are often also difficult for human observers.

3.8 Analyzing Discriminative Ability of Our Approach

Further, we compare the intra- and inter-class accumulated affinity scores over different datasets, which offers an empirical explanation for the discriminative ability of our approach. The results displayed in Fig. 12 are obtained as follows: For each non-zero (i, j)-th entry in E there is a directed path connected both nodes. Now group all entries in E into two sets: Those with both nodes belonging to the same class (i.e. intra-class), and those each of which is from a different class (i.e. inter-class). Then accumulate the scores within each set and normalize – which produces the final scores. The intra-class score is expected to outnumber the inter-class one, and the larger the gap (or ratio) between the two suggests a better discriminative ability on the particular dataset. As revealed in Fig. 12, the ratios are all very large across various datasets used in this paper, which indeed suggests that our algorithm is expected to deliver good performance regardless of any particular set of input labels.

4 Conclusion and Outlook

A novel random walk approach is proposed on digraphs that is able to preserve edge directions and is shown to perform competitively against the state-of-the-art methods. For future work, we plan to explore broader scope of applications, to generalize to work with problems with structured labels, as well as to investigate its potentials in spectral clustering on digraphs.

Acknowledgements

We would like to acknowledge support for this project from A*STAR JCO grants.

Appendix

Proof of Proposition 1

Proof We know [23] that the fundamental matrix of $\tilde{Q}$ is $E = \sum_{t=0}^{\infty} Q^t$, and are left to show that $(I - \alpha P^T)^{-1}$ exists, and $E = (I - \alpha P^T)^{-1}$. $(I - \alpha P^T)^{-1}$ exists, since its spectral radius $\rho(P)$ defined as the absolute value of its largest eigenvalue is always 1, and $\alpha < \rho(P)^{-1}$ since $\alpha \in (0, 1)$. we also have $Q^{\infty} = (\alpha P^T)^{\infty} = 0$ and the series $I + Q + Q^2 + \ldots$ will converge to $(I - \alpha P^T)^{-1}$. ■

Proof of Proposition 2

Proof (i) We focus only on the change in q_{ij}. The difference between Q and Q' is given by

$$Q' - Q = \Delta q_{ij} \varepsilon_i \varepsilon_j^T,$$

TABLE 2. Average DIADEM scores (DS) are reported for the synthetic dataset, as well as the widely used DRIVE and STARE testbeds.

	NBC	NLB	CDRN	WVRN	CTKd	SGL	SOP	ZFL	Ours
Syn [51]	0.64	0.62	0.61	0.62	0.61	0.64	0.60	0.70	**0.73**
DRIVE [3]	0.71	0.69	0.63	0.62	0.68	0.71	0.63	0.73	**0.76**
STARE [1]	0.33	0.29	0.27	0.25	0.30	0.38	0.22	0.39	**0.41**

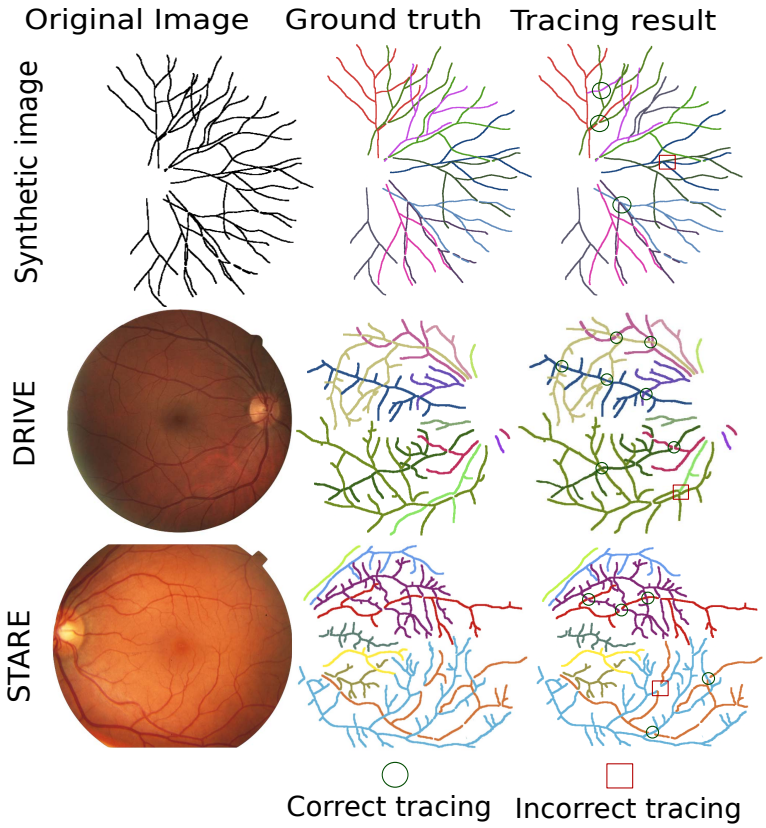


Fig. 11: Exemplar retinal tracing results on Synthetic dataset, DRIVE, and STARE. The first, second and third column shows the original images, ground-truth images and tracing results of our approach, respectively. Segments with the same color form a distinct vessel tree. Thus the number of colors equal to the number of classes (vessel trees). Selected correct (wrong) tracing segments are shown in green circles (red squares).

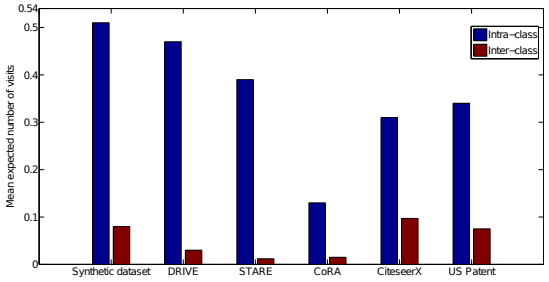


Fig. 12: Empirical discriminative ability of our approach. Intra- and inter-class accumulated affinities are displayed over different datasets. Intuitively the larger the gap between intra- and inter-class, the better its performance would be. See text for details.

where ε_i (ε_j) is a column vector with the i-th (j-th) entry being 1 and all other entries being 0. By the Sherman-Morrison-Woodbury formula [52], we have

$$\begin{aligned} E' =&(I-Q')^{-1} = (I - Q - \Delta q_{ij}\varepsilon_i\varepsilon_j^T)^{-1} \\ =&(I-Q)^{-1} \\ &+ \frac{\Delta q_{ij}}{1-\Delta q_{ij}\varepsilon_j^T(I-Q)^{-1}\varepsilon_i}(I-Q)^{-1}\varepsilon_i\varepsilon_j^T(I-Q)^{-1} \\ =&E + \frac{\Delta q_{ij}}{1-\Delta q_{ij}e_{ji}}E_{:i}E_{j:}, \end{aligned}$$

which completes the proof of part (i).

(ii) The update of E is obtained by noting that

$$Q' - Q = \varepsilon_i \Delta Q_{i:}$$

and applying the Sherman-Morrison-Woodbury formula as in part (i).

(iii) By the definition of E, we have

$$\begin{aligned} E' =&(I-Q')^{-1} = \begin{bmatrix} I-Q & -u \\ -v^T & 1-q \end{bmatrix}^{-1} \\ =& \begin{bmatrix} (I-Q)^{-1} + \gamma(I-Q)^{-1}uv^T(I-Q)^{-1} & \gamma(I-Q)^{-1}u \\ \gamma v^T(I-Q)^{-1} & \gamma \end{bmatrix} \\ =& \begin{bmatrix} E + \gamma(Eu)(v^TE) & \gamma(Eu) \\ \gamma(v^TE) & \gamma \end{bmatrix}, \end{aligned}$$

where $\gamma = \frac{1}{(1-q)-v^T(I-Q)^{-1}u} = \frac{1}{(1-q)-v^TEu}$.

■

CONNECTIONS TO GRAPH LAPLACIAN IN UNDIRECTED GRAPHS

Here we shown that when operating as random walks on undirected graphs, our algorithm is equivalent to a scaled variant of the graph Laplacian method of [15]. For an undirected graph G, denote $S = D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$, and define $P = WD^{-1}$, where $W = [w_{ij}]$ is a symmetric matrix and $D = \mathrm{diag}(d_1, \cdots, d_n)$, with $d_i = \sum_j w_{ij}$. Now consider applying our algorithm (i.e. (4) and (5)) on undirected graph G. Since

$$A = (I - \alpha P^T)^{-1}Y = D^{-\frac{1}{2}}(I - \alpha S)^{-1}D^{\frac{1}{2}}Y,$$

we have

$$D^{\frac{1}{2}}A = (I - \alpha S)^{-1}\Big(D^{\frac{1}{2}}Y\Big).$$

Notice that since the goal is to choose the best element from the current row i as in (5), the result will not change by multiplying an additional constant $d_i^{\frac{1}{2}}$ to all elements in the row. Define $\hat{\mathrm{A}} := D^{\frac{1}{2}}\mathrm{A}$, and let $\hat{Y} := D^{\frac{1}{2}}Y$ we now have

$$\hat{A} = (I - \alpha S)^{-1}\hat{Y}, \tag{A.1}$$

which recovers the update formula of [15], with the only difference that instead of Y, $\hat{Y}$ is used here as a row-wise scaled variant. In Section 3.5, we compare the method in (A.1), simply denoted as UG, with other undirected-graph based methods.

CONNECTIONS TO PARTIALLY ABSORBING RANDOM WALKS (PARW) [17]

The random walks considered in [17] deal with a *special* form of *absorbing Markov chains* where the submatrix of its weight matrix concerning transient nodes forms a symmetric non-negative matrix with diagonal entries taking zero values. More formally, denote this submatrix as

$$W_P = \begin{pmatrix} 0 & w_{1\,2} & \cdots & w_{1\,n} \\ w_{2\,1} & 0 & \cdots & w_{2\,n} \\ \cdots & \cdots & \cdots & \cdots \\ w_{n\,1} & \cdots & w_{n\,n-1} & 0 \end{pmatrix},$$

which is a $n \times n$ symmetric non-negative matrix with zero diagonal values. Let $\Lambda_P = \mathrm{diag}(\lambda_1, \cdots, \lambda_n)$ with $\lambda_i > 0\ \forall i$, and define $\lambda_{\mathbf{P}} = \mathrm{vec}(\Lambda_P)$, where the operator $\mathrm{vec}(\cdot)$ extracts the diagonal elements of the input matrix to produce a column vector. The weight matrix of the PARW family proposed in [17] can be regarded as an extended matrix of W_P by introducing an additional absorbing node, as

$$\tilde{W}_P \quad = \quad \left(\begin{array}{c|c} W_P & \lambda_{\mathbf{P}} \\ \hline 0 & 1 \end{array}\right),$$

with 0 here referring to a $1 \times n$ vector of zero values. Define the degree matrix $D_P = \mathrm{diag}(d_1, \cdots, d_n)$ with each element computed as the sum of the corresponding row of W_P, $d_i = \sum_j w_{i\,j}$. denote the (sub-) graph Laplacian $L_P = D_P - W_P$, and let $P_W = (\Lambda_P + D_P)^{-1}W_P$, and $P_\lambda = (\Lambda_P + D_P)^{-1}\Lambda_P$. At this point, we are ready to obtain the *probability transition matrix*:

$$\tilde{P}_P \quad = \quad \left(\begin{array}{c|c} P_W & \mathrm{vec}(P_\lambda) \\ \hline 0 & 1 \end{array}\right),$$

which is exactly the same form as of $\tilde{Q}$ defined earlier in Equation (1) of our approach. Note that the random walks considered in [17] is a *special* form of (1) with W_P confined to being a symmetric matrix with zero diagonal entries. Following Proposition 1, its fundamental matrix becomes

$$\begin{aligned} E_P = \sum_{t=0}^{\infty}\big(P_W\big)^t &= \big(I - (\Lambda_P + D_P)^{-1}W_P\big)^{-1} \\ &= \big(\Lambda_P + L_P\big)^{-1}\big(\Lambda_P + D_P\big). \end{aligned}$$

It is interesting to observe that the absorption probability matrix, A_P, proposed and discussed in [17] can be related to E_P as

$$A_P = \sum_{t=0}^{\infty}\big(P_W\big)^t P_\Lambda = E_P\Big[\big(\Lambda_P + D_P\big)^{-1}\Lambda_P\Big].$$

Interestingly, it corresponds to a special form of the absorbing probabilities of the Markov chain (see e.g. Theorem 3.3.7 of [23]).

REFERENCES

[1] A. Hoover, V. Kouznetsova, and M. Goldbaum, "Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response," *IEEE Trans on Med. Imag.*, vol. 19, no. 3, pp. 203 –210, 2000.

[2] E. Meijering, M. Jacob, J. Sarria, P. Steiner, H. Hirling, and M. Unser, "Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images," *Cytometry A*, vol. 58, no. 2, pp. 167–76, 2004.

[3] J. Staal, M. Abramoff, M. Niemeijer, M. Viergever, and B. van Ginneken, "Ridge based vessel segmentation in color images of the retina," *IEEE Trans Med. Imag.*, vol. 23, no. 4, pp. 501–9, 2004.

[4] E. Turetken, C. Becker, P. Glowacki, F. Benmansour, and P. Fua, "Detecting irregular curvilinear structures in gray scale and color imagery using multi-directional oriented flux," in *ICCV*, 2013.

[5] E. Turetken, F. Benmansour, B. Andres, H. Pfister, and P. Fua, "Reconstructing loopy curvilinear structures using integer programming," in *CVPR*, 2013.

[6] X. Cai, H. Wang, H. Huang, and C. Ding, "Simultaneous image classification and annotation via biased random walk on tri-relational graph," in *ECCV*, 2012.

[7] M. Chen, M. Liu, J. Liu, and X. Tang, "Isoperimetric cut on a directed graph," in *CVPR*, 2010.

[8] A. McCallum, K. Nigam, J. Rennie, and K. Seymore, "Automating the construction of internet portals with machine learning," *Inf. Retr.*, vol. 3, no. 2, pp. 127–163, 2000.

[9] S. Macskassy and F. Provost, "Classification in networked data: A toolkit and a univariate case study," *JMLR*, vol. 8, pp. 935–983, 2007.

[10] H. Wang, C. Ding, and H. Huang, "Directed graph learning via high-order co-linkage analysis," in *ECML*, 2010.

[11] F. Fouss, K. Francoisse, L. Yen, A. Pirotte, and M. Saerens, "An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification," *Neural Network*, vol. 31, pp. 53–72, 2012.

[12] O. Chapelle, B. Schölkopf, and A. Zien, Eds., *Semi-Supervised Learning*. MIT Press, 2006.

[13] X. Zhu and A. Goldberg, *Introduction to Semi-Supervised Learning*. Morgan & Claypool, 2009.

[14] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *ICML*, 2003.

[15] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Scholkopf, "Learning with local and global consistency," in *NIPS*, 2004.

[16] Y. Bengio, O. Delalleau, and N. Le Roux, "Label propagation and quadratic criterion," in *Semi-Supervised Learning*, O. Chapelle, B. Schölkopf, and A. Zien, Eds. MIT Press, 2006, pp. 193–216.

[17] X. Wu, Z. Li, A. So, J. Wright, and S. Chang, "Learning with partially absorbing random walks," in *NIPS*, 2012.

[18] J. Wang, T. Jebara, and S. Chang, "Semi-supervised learning using greedy max-cut," *J. Mach. Learn. Res.*, vol. 14, no. 1, pp. 771–800, 2013.

[19] D. Zhou, J. Huang, and B. Schölkopf, "Learning from labeled and unlabeled data on a directed graph," in *ICML*, 2005.

[20] A. Mantrach, L. Yen, J. Callut, K. Françoisse, M. Shimbo, and M. Saerens, "The sum-over-paths covariance kernel: A novel covariance measure between nodes of a directed graph," *IEEE Trans. PAMI*, vol. 32, no. 6, pp. 1112–1126, 2010.
[21] T. Gartner, Q. Le, S. Burton, A. Smola, and S. Vishwanathan, "Large-scale multiclass transduction," in *NIPS*, 2005.
[22] A. Subramanya and J. Bilmes, "Semi-supervised learning with measure propagation," *JMLR*, vol. 12, pp. 3311–70, 2011.
[23] J. G. Kemeny and J. L. Snell, *Finite Markov Chains*. Springer, 1976.
[24] F. Chung, "Laplacians and the cheeger inequality for directed graphs," *Annals of Combinatorics*, vol. 9, no. 1, pp. 1–19, 2005.
[25] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," in *WWW*, 1998.
[26] X. Zhu, A. Goldberg, J. Van, and G. D. Andrzejewski, "Improving diversity in ranking using absorbing random walks," in *HLT-NAACL*, 2007.
[27] P. Chebotarev and E. Shamis, "On proximity measures for graph vertices," *Automat. and Remote Control*, vol. 59, pp. 1443–1459, 1998.
[28] B. Schölkopf and A. Smola, *Learning with Kernels*. Cambridge, MA: MIT Press, 2002.
[29] P. Sarkar and A. Moore, "A tractable approach to finding closest truncated-commute-time neighbors in large graphs," in *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence*, R. Parr and L. van der Gaag, Eds., 2007, pp. 335–343.
[30] R. Agaev and P. Chebotarev, "Spanning forests of a digraph and their applications," *Automat. Remote Control*, vol. 62, no. 3, pp. 443–466, 2001.
[31] P. Chebotarev and E. Shamis, "The matrix-forest theorem and measuring relations in small social groups," *Automat. Remote Control*, vol. 58, no. 9, pp. 1505–14, 1997.
[32] T. Davis, "Algorithm 832: Umfpack v4.3—an unsymmetric-pattern multifrontal method," *ACM Trans. Math. Softw.*, vol. 30, pp. 196–199, 2004.
[33] J. Demmel, S. Eisenstat, J. Gilbert, X. Li, and J. Liu, "A supernodal approach to sparse partial pivoting," *SIAM J. Matr. Anal. App.*, vol. 20, pp. 720–755, 1999.
[34] C. Paige and M. Saunders, "LSQR: An algorithm for sparse linear equations and sparse least squares." *ACM Trans. Math. Softw.*, vol. 8, pp. 43–71, 1982.
[35] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed. SIAM, 2003.
[36] R. El-Yaniv and D. Pechyony, "Transductive Rademacher complexity and its applications," *J. Artif. Intell. Res.*, vol. 35, pp. 193–234, 2009.
[37] P. Bartlett and S. Mendelson, "Rademacher and Gaussian complexities: risk bounds and structural results," *JLMR*, vol. 3, pp. 463–482, 2002.
[38] L. Bégin, P. Germain, F. Laviolette, and J.-F. Roy, "PAC-Bayesian theory for transductive learning," in *International Conference on Artificial Intelligence and Statistics (AISTAT)*, 2014, pp. 105–13.
[39] J. J. McAuley and J. Leskovec, "Learning to discover social circles in ego networks," in *NIPS*, 2012, pp. 548–556.
[40] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: Densification laws, shrinking diameters and possible explanations," in *In KDD*. ACM Press, 2005, pp. 177–187.
[41] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (COIL-20)," Tech. Rep. CUCS-005-96, 1996.
[42] K. Lang, "Newsweeder: Learning to filter netnews," in *Proceedings of the 12th International Machine Learning Conference (ICML-95)*, 1995.
[43] S. Chakrabarti, B. Dom, and P. Indyk, "Enhanced hypertext categorization using hyperlinks," in *SIGMOD*, 1998.
[44] Q. Lu and L. Getoor, "Link-based classification," in *ICML*, 2003.
[45] A. Mantrach, N. van Zeebroeck, P. Francq, M. Shimbo, H. Bersini, and M. Saerens, "Semi-supervised classification and betweenness computation on large, sparse, directed graphs," *PR*, vol. 44, no. 6, pp. 1212–1224, 2011.
[46] J. Callut, K. Françoisse, M. Saerens, and P. Dupont, "Semi-supervised classification from discriminative random walks," in *Machine Learning and Knowledge Discovery in Databases*, W. Daelemans, B. Goethals, and K. Morik, Eds., 2008, pp. 162–177.
[47] P. Sen and L. Getoor, "link based classification," CS department, Univ of Maryland, Tech. Rep., 2007, cS-TR-4858.
[48] S. Godbole and S. Sarawagi, "Discriminative methods for multi-labeled classification," in *PAKDD*. Springer, 2004, pp. 22–30.
[49] T. A. Gillette, K. M. Brown, and G. A. Ascoli, "The diadem metric: comparing multiple reconstructions of the same neuron." *Neuroinformatics*, vol. 9, no. 2-3, pp. 233–245, 2011.
[50] C. Giles, K. Bollacker, and S. Lawrence, "Citeseer: an automatic citation indexing system," in *Int. Conf. on Digital Libraries*, 1998.
[51] J. De, H. Li, and L. Cheng, "Tracing retinal vessel trees by transductive inference," *BMC Bioinformatics*, vol. 15, no. 20, 2014.
[52] G. Golub and C. V. Loan, *Matrix Computations*, 3rd ed. Johns Hopkins University Press, 1996.

Jaydeep De received his B. Tech (first class honors) in Electronic Engineering from National Institute of Technology (2007) and Ph.D degree from Nanyang Technological Univ. (2015), Singapore. He is currently a Computer Vision Scientist at Naspers GmbH at Berlin, Germany. His current research involves Object Recognition for web content filtering and quality improvement. Prior to joining Naspers, he has worked with Dr. Li Cheng at Bioinformatics Institute, Singapore (2011-2015), which lead to his Ph.D dissertation.

Xiaowei Zhang is a senior postdoctoral research fellow at the Bioinformatics Institute, A*STAR, Singapore. He received the B.Sc. degree in information and computing science from the East China Normal University, Shanghai, China, in 2008, and the Ph.D. degree in applied mathematics from the National University of Singapore, in 2013. Since 2013, he has been a postdoctoral research fellow at the Bioinformatics Institute, A*STAR, Singapore. His current research interests include machine learning and its applications to computer vision, data mining, matrix computations and its applications, and numerical optimization.

Feng Lin is an Associate Professor and the Director of Biomedical Informatics Lab in the School of Computer Science and Computer Engineering, Nanyang Technological University, Singapore. His research interest includes biomedical informatics, bioimaging, computer graphics and visualization, and high performance computing. He has published 230 research works as monographs, in books, journals and conference proceedings. He is a Senior Member of IEEE.

Li CHENG is a principal investigator and group leader at Bioinformatics Institute (BII) of A*STAR, as well as a guest professor of Chongqing University. Prior to joining BII July of 2010, He worked at Statistical Machine Learning group of NICTA, Australia, TTI-Chicago, USA, and University of Alberta, Canada, where he obtained his Ph.D.in Computer Science. His research expertise is mainly on machine learning and computer vision.