World Wide Web
https://doi.org/10.1007/s11280-020-00817-8

Accurate relational reasoning in edge-labeled graphs by multi-labeled random walk with restart

Jinhong Jung[1] · Woojeong Jin[2] · Ha-myung Park[3] · U Kang[1]

Received: 30 April 2019 / Revised: 19 January 2020 / Accepted: 3 April 2020 /
Published online: 21 May 2020

Abstract
Given an edge-labeled graph and two nodes, how can we accurately infer the relation between the nodes? Reasoning how the nodes are related is a fundamental task in analyzing network data, and various relevance measures have been suggested to effectively identify relevance between nodes in graphs. Although many random walk based models have been extensively utilized to reveal relevance between nodes, they cannot distinguish how those nodes are related in terms of edge labels since the traditional surfer does not consider edge labels for estimating relevance scores. In this paper, we propose MURWR (MULTI-LABELED RANDOM WALK WITH RESTART), a novel random walk based model that accurately identifies how nodes are related with, considering multiple edge labels. We introduce a labeled random surfer whose label indicates the relation between starting and visiting nodes, and change the surfer's label during random walks for multi-hop relational reasoning. We also learn appropriate rules on changing the surfer's label from the edge-labeled graph to accurately infer relations. We develop an iterative algorithm for computing MURWR, and prove the convergence guarantee of the algorithm. Through extensive experiments, we show that our model MURWR provides the best inference performance.

Keywords Relational reasoning · Multi-labeled random walk with restart · Edge-labeled graphs

1 Introduction

How can we accurately infer the relation between two arbitrary nodes in edge-labeled graphs? Understanding how nodes are related is crucial for analyzing graph data, and many researchers have designed various relevance measures to effectively identify relevance

This article belongs to the Topical Collection: *Special Issue on Artificial Intelligence and Big Data Computing*
Guest Editors: Wookey Lee and Hiroyuki Kitagawa

✉ U Kang
ukang@snu.ac.kr

Extended author information available on the last page of the article.

between nodes in graphs. Random Walk with Restart (RWR) [38], a random surfer model, has been utilized for measuring relevance scores between nodes with considering *global network structure* [10] and *multi-faceted connections* between nodes [37]. RWR has been extensively utilized in numerous graph mining applications such as personalized ranking [38], link prediction [22], recommender systems [30], anomaly detection [31], community detection [1], etc.

Although many networks have been modeled with multiple edge labels to represent diverse relationships between nodes [2], RWR has a limitation in inferring the edge's label between two nodes in edge-labeled graphs. For example, social networks such as Slashdot [15] represent trust or distrust for users as positive or negative edges. In knowledge graphs such as WordNet [27], concepts are associated with predicates. Since RWR does not consider edge labels for its relevance, it cannot identify how nodes are related with in terms of edge labels (see Figure 1). For relation inference on multiple edge labels, there are two main approaches: path feature models and translation models. Path feature models [16] exploit paths between two nodes as features for predicting their relation. However, their reasoning focuses on relatively short paths due to the expensive cost of path enumeration. On the other hand, translation models [4, 24, 41] discover latent embeddings for relations and entities under a relational translation scenario; however, they do not take account of paths between nodes into those embeddings. Although several translation models [9, 23] have been proposed by considering paths, they still require complicated path enumeration as similar to the path feature models. These limitations make the reasoning of those approaches miss the information provided by complex and long paths between the nodes.

In this paper, we propose MuRWR (Multi-Labeled Random Walk with Restart), a novel random surfer model which accurately identifies label relevance between two nodes in an edge-labeled graph (see Figure 2). Our approaches are 1) to introduce a *labeled random surfer* whose label indicates the relation between starting and visiting nodes, 2) to change the surfer's label during random walks for multi-hop reasoning process, and 3) to learn suitable rules on how to change her label from the given edge-labeled graph. We show that letting the labeled surfer move around the graph enables our model to do accurate multi-hop relational reasoning without explicit path enumeration, as well as to generalize RWR into edge-labeled graphs. Through extensive experiments, we show that MuRWR predicts edge labels between nodes more accurately than existing models. Our main contributions are as follows:

- **Model.** We propose MuRWR (Multi-Labeled Random Walk with Restart), a novel and accurate model for relation inference in edge-labeled graphs (Definition 4). MuRWR exploits a labeled random surfer who considers edge labels to compute effective relevance scores between nodes for each edge label. We show that MuRWR is a generalized version of RWR to edge-labeled graphs (Property 1).

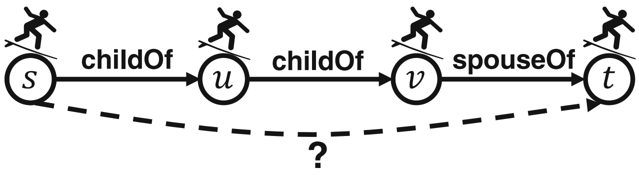


Figure 1 Limitation of a random surfer in RWR for estimating relation of s and t. Since the surfer does not consider edge labels, it cannot identify the relation

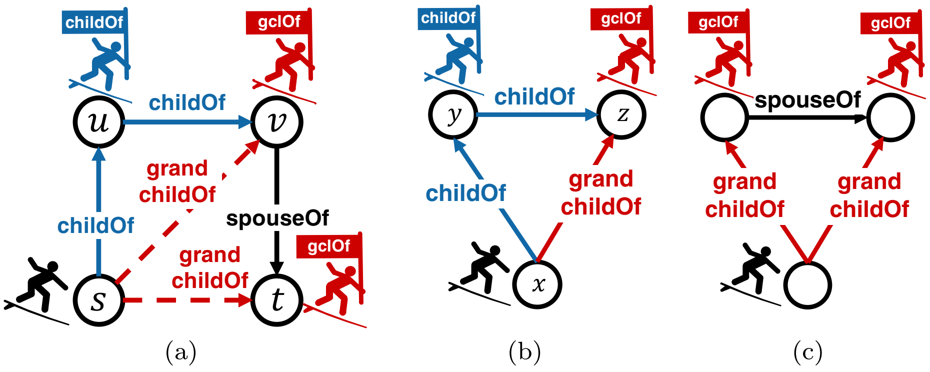


Figure 2 Examples of labeled walks and label transitive triangles. **(a)** shows how our labeled surfer walks along the path from s to t. The blue surfer is for `childOf`, the red one is for `gclOf` (`grandchildOf`), and the black one is unlabeled. In (a), a dashed line indicates two disconnected nodes are related with a label inferred by the surfer. **(b)** and **(c)** are the examples of label transitive triangles used for learning the rules on how to change the surfer's label

- **Algorithm.** We propose how to learn the surfer's label from an edge-labeled graph (Lemma 1) and an iterative algorithm for our model (Algorithm 3). We also theoretically prove the convergence of the algorithm (Theorem 1).
- **Experiment.** Our experiments show our model provides the most accurate performance for relation inference (Tables 3 and 4).

The rest of the paper is organized as follows. We first discuss related works and their limitations in Section 2. Then, we provide preliminaries on a traditional random walk based model, and formally define edge-labeled graphs in Section 3. We describe our model and algorithms for computing our model scores in Section 4. After presenting experimental results in Section 5, we conclude in Section 6. The symbols used in this paper are in Table 1.

2 Related work

There are various traditional graph measures for relevance between nodes in edge-unlabeled graphs based on random walk, e.g., PageRank [29], HITS [13], and RWR [38]. Among these models, RWR has received much attention and has been applied to many graph mining tasks. However, RWR has a limitation on predicting the relation between two nodes in edge-labeled graphs since it does not consider edge labels for its relevance (details in Section 3).

Several techniques [11, 19] have been proposed to compute RWR in heterogeneous networks. These methods focus on how to determine the weights of edges by exploiting attributes in the networks, and then construct a transition matrix with the weights to compute RWR. However, they also cannot infer the relation between the nodes in edge-labeled graphs because they produce only one relevance score between two nodes, similarly to RWR.

For relation inference, we need to obtain K relevance scores for edge labels between two nodes if a graph has K edge labels.

Many researchers have recently made great efforts to apply RWR for relevance between nodes in signed networks, a special type of edge-labeled graphs, represented by positive (trust) and negative (distrust) edges. Modified RWR (MRWR) [32] computes RWR as trust and distrust scores in positive and negative subgraphs, respectively. Although the idea is

Table 1 Symbols

Symbol	Definition
G	input edge-labeled graph
n	number of nodes in G
m	number of edges in G
K	number of edge labels in G
s	source node
c	restart probability
l_i	i-th edge label where $1 \leq i \leq K$
l_{uv}	label on the edge from node u to node v
$\overleftarrow{\mathbf{N}}_v$	set of in-neighbors of node v
$\overrightarrow{\mathbf{N}}_v$	set of out-neighbors of node v
$\mathbf{L}$	set of edge labels, i.e., $\mathbf{L} = \{l_1, \cdots, l_K\}$
$\mathbf{A}$	$(n \times n)$ labeled adjacency matrix of G
$\mathbf{A}_k$	$(n \times n)$ l_k-labeled semi-adjacency matrix
$\tilde{\mathbf{A}}_k$	$(n \times n)$ l_k-labeled semi-row-normalized matrix
$\mathbf{R}$	$(n \times K)$ relevance score matrix w.r.t. source node s
$\mathbf{R}_{vi}$	relevance score between nodes s and v for edge label l_i
$\mathbf{S}_k$	$(K \times K)$ label transition probability matrix on l_k
$\mathbf{S}_{kij}$	probability that the surfer's label changes from l_i to l_j through l_k-labeled edge, i.e., $P(l_i \xrightarrow{l_k} l_j)$
$\mathbf{1}(\cdot)$	indicator function that returns 1 if a given predicate is true, or 0 otherwise

applicable to edge-labeled graphs by computing RWR on each subgraph containing only a specific edge label, this leads to many disconnections between nodes; thus, MRWR is unable to exploit meaningful patterns from multi-hop paths. Signed RWR (SRWR) [12] measures trust and distrust scores between nodes based on random walks for signed networks. However, SRWR is inapplicable to relation inference on multiple edge labels since it is specialized for only two sign labels.

Two major approaches on relation inference for multiple edge labels are classified into *path feature model* and *translation based model.*

Path Ranking Algorithm (PRA) [16] is commonly used as a path feature model in heterogeneous networks. PRA extracts paths connecting two nodes, and exploits a random surfer to measure path probabilities which are used as features when predicting their relation. PRA, however, requires explicit path enumeration which becomes computationally problematic when it comes to long paths. Although the authors presented heuristic pruning techniques, PRA's inference has still been restricted to short paths since the path enumeration essentially has an exponential complexity to path length.

Translation based models such as TransE [4], TransR [24], and DistMult [41] have been widely utilized due to its simplicity and effectiveness for modeling relational data. They formulate the relation between two nodes as a translation between the corresponding node embeddings. However, those models consider only one directed edge at a time in training; hence, their reasoning is likely to miss the information provided by multi-hop paths between them.

To resolve this issue, PTransE [23] and COMP [9] have been proposed by utilizing multi-step paths under the translation embedding mechanism. However, these methods also suffer from the path enumeration problem as similar to PRA; thus, they have restricted the length of paths for relational reasoning to at most three steps.

Compared to the above methods, our proposed model MuRWR performs accurate multi-hop relational reasoning without complicated and heuristic path enumeration.

3 Preliminary

We provide preliminaries on relation inference in edge-labeled graphs, our target research task, and RWR in this section. Note that we use the terms *edge label* and *relation* interchangeably.

Edge-labeled Graphs Many researchers from various application areas are confronted with labeled edges which encode diverse relations between entities in graphs [2, 15, 21, 27].

Formally, an edge-labeled graph is defined as follows.

Definition 1 (Edge-labeled Graph and Labeled Adjacency Matrix) An edge-labeled graph $G = (\mathbf{V}, \mathbf{E}, \mathbf{L})$ consists of the set $\mathbf{V}$ of nodes, the set $\mathbf{E}$ of directed edges, and the set $\mathbf{L}$ of edge labels. Let $\mathbf{L} = \{l_1, \cdots, l_K\}$ where l_k is k-th label, and K is the number of edge labels. For each edge $u \rightarrow v \in \mathbf{E}$ such that $u, v \in \mathbf{V}$, the edge is associated with an edge label $l_{uv} \in \mathbf{L}$. The labeled adjacency matrix $\mathbf{A}$ of G is a sparse matrix such that $\mathbf{A}_{uv}$ is l_{uv} if there is an l_{uv}-labeled edge from u to v, and 0 otherwise.

Relation Inference Task We describe the formal problem definition of relation inference handled in this paper.

Problem 1 (Relation Inference [16]) The relation inference task is defined as follows:

- **Given**: an edge-labeled graph $G = (\mathbf{V}, \mathbf{E}, \mathbf{L})$, and two disconnected nodes s and t in $\mathbf{V}$,
- **Infer**: the edge label from source node s to target node t among edge labels in $\mathbf{L}$.

Note that Problem 1 is different from that of belief propagation (BP) [7, 14] that aims to classify labels on nodes not edges while the goal of MuRWR is to predict labels on edges.

Random Walk with Restart Given a graph and a source node s, RWR computes a relevance score between s and each node using a random surfer who starts from s. The surfer performs one of the followings at each step:

- **Random Walk.** The surfer randomly moves to one of its neighbors from the current node with probability $1 - c$.
- **Restart.** The surfer goes back to s with probability c.

The relevance score between s and node t is the stationary probability that the surfer is at t after moving around the entire graph from s.

Note that RWR considers multi-faceted paths between nodes [37] with restart probability c which controls how long paths affect the score between s and t. If c is high, the surfer frequently restarts from s; consequently, she mainly visits t via relatively short paths. Otherwise, the surfer is also able to reach t through longer paths as well as short ones.

The main challenge is to devise how to reflect multiple edge labels on relevance scores between nodes in edge-labeled graphs. Although RWR is able to identify relevance scores between nodes, it depends only on the link structure of the graph without considering labels; therefore, it cannot estimate how nodes are related in terms of edge labels. Figure 1 shows an example of representing family relationships such as `childOf` and `spouseOf`. The traditional surfer in RWR does not consider edge labels at all during its walks while the labeled edges should be interpreted differently. Thus, the information encoded in the path is ignored when measuring the relevance between nodes s and t; RWR cannot identify which edge label should be associated with nodes s and t. How can we make the surfer recognize edge labels to correctly predict their relation?

4 Proposed Method

We propose MuRWR (Multi-Labeled Random Walk with Restart), a novel random walk based model which measures relevance scores between a source node and other nodes for each edge label in an edge-labeled graph. The technical challenges and our main ideas are as follows:

- How can we make a surfer consider the edge labels? We introduce a *labeled random surfer* whose label at a node indicates the relation from the source node to that node.
- How can the surfer infer the relation between the nodes? We allow the surfer to change her label during random walks with *rules* for a multi-hop reasoning.
- How can we discover the rules? We exploit a data-driven approach to extract knowledge from the graph so that the surfer learns the rules.

In MuRWR, the surfer's movement from source node s to a node t is considered as a process of reasoning about the relation from s to t. Figure 2 depicts how MuRWR infers the relation from node s to node t. The labeled surfer has one of edge-labels such as `childOf` (blue-colored) or `grandchildOf` (red-colored) at each node except at s. Assume we have the following rules for the surfer:

Rule 1 If `childOf`-surfer moves along `childOf`-edge, the surfer's label changes to `grandchildOf`.
Rule 2 If `grandchildOf`-surfer moves along `spouseOf`-edge, the surfer's label remains the same, i.e., `grandchildOf`.

In Figure 2a, the surfer first starts from source node s without any label, and moves to node u. Then, she has `childOf` label at u since nodes s and u are directly connected with `childOf`-edge (note that her label indicates the relation between the *source* node s and the current node). After she moves to node v, her label changes to `grandchildOf` by the first rule. At this time, MuRWR infers the relation from s to v as `grandchildOf` as shown in Figure 2a. When the surfer finally arrives at node t, her label is still `grandchildOf` by the second rule, indicating that the relation from s to t is also inferred as `grandchildOf`. Thus, introducing a label to the surfer enables her to do a multi-hop relational reasoning by walking around the graph if the surfer knows appropriate rules.

Then, how can we discover the rules for the surfer? For this, we exploit a data-driven approach, which extracts knowledge embraced in the given graph. The knowledge that the surfer learns is represented as labeled transitive triangles as in Figure 2b and c. A transitive triangle is interpreted as a syllogistic knowledge, e.g., Figure 2b implies

$$\underbrace{\texttt{childOf}(x,y)}_{\text{current surfer's label}} \wedge \underbrace{\texttt{childOf}(y,z)}_{\text{edge label}} \Rightarrow \underbrace{\texttt{grandchildOf}(x,z)}_{\text{next surfer's label}}.$$

An intuitive example for the surfer's learning is as follows. Suppose an untrained surfer (black-colored) is at node x in Figure 2a. The surfer then has `childOf`-label at node y since x and y are directly related with the label. Similarly, she has `grandchildOf`-label at node z. This is interpreted as: if `childOf`-surfer moves along `childOf`-edge, her label changes to `grandchildOf`. Thus, we enumerate transitive triangles from the graph, and use them as training instances, called *label transition observations*, for the surfer.

We first describe the details on label translation observations in Section 4.1, and explain how to learn rules for our labeled surfer from the observations in Section 4.2. Then, we formally define and formulate our model MuRWR in Sections 4.3 and 4.4, respectively. We present the algorithms for computing MuRWR in Section 4.5, and prove its convergence.

4.1 Label transition observation

We describe how to collect observations for learning the surfer's rules from the graph G. We first define *label transition observation* as follows:

Definition 2 (Label Transition Observation) A label transition observation $l_i \xrightarrow{l_k} l_j$ is that when l_i-labeled surfer moves along l_k-labeled edge, her label changes to l_j.

A label transition observation is obtained from a *label transitive triangle* interpreted as a transition between edge labels. For example, suppose an untrained surfer (black-colored) is at node x in Figure 3a. When the surfer moves to node y, her label should be l_i because x and y are directly related with label l_i. Similarly, her label is l_j at node z. Then we observe how the surfer's label changes as in Figure 3c: when l_i-labeled surfer moves along l_k-labeled edge, her label changes to l_j, implying the label transition observation $l_j \xrightarrow{l_k} l_j$ in Definition 2. To collect such observations, we enumerate all transitive triangles from the graph G using an efficient triangle enumeration algorithm [17].

In this paper, we exploit only transitive triangles for learning the surfer's rules. Although other types of structures might be used for this purpose, it is not very straightforward to

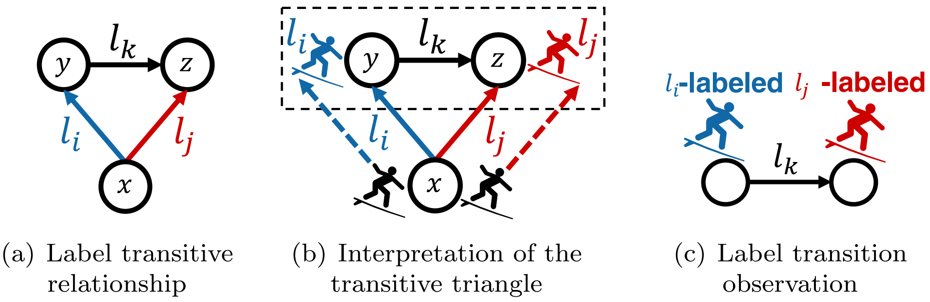

l_k
y
z
l_i
l_j
x
(a) Label transitive relationship
(b) Interpretation of the transitive triangle
l_i-labeled
l_j -labeled
(c) Label transition observation

Figure 3 Example of how to obtain label transition observations from label transitive relationships. **(a)** presents a label transitive relationship. **(b)** shows how to interpret the triangle to obtain the label transition observation $l_i \xrightarrow{l_k} l_j$ in **(c)**

extract rules from arbitrary subgraphs. Even if we could discover some rules from the subgraphs, it is not efficient and scalable to enumerate all of the subgraphs from a graph. On the other hand, triangles are easily interpreted as in Figure 3b, and efficiently enumerated. We leave the problem of considering other types of subgraphs for the surfer's rule as a future work.

4.2 Learning label transition probabilities

We explain how to learn the surfer's rules from the label transition observations. A *label transition probability* is a conditional probability $P(l_j|l_i, l_k)$ such that when l_i-labeled surfer moves along l_k-labeled edge, her label changes to l_j. For brevity, let $P(l_i \xrightarrow{l_k} l_j)$ denote $P(l_j|l_i, l_k)$. We formally define *label transition probability matrix* as follows:

Definition 3 (Label Transition Probability Matrix) Let $\mathbf{S}_k \in \mathbb{R}^{K\times K}$ be a label transition probability matrix on edge label l_k. The (i, j)-th entry $\mathbf{S}_{kij}$ of $\mathbf{S}_k$ indicates the label transition probability that the surfer's label changes from l_i to l_j through edge label l_k, i.e., $\mathbf{S}_{kij} = P(l_i \xrightarrow{l_k} l_j)$. Note that $\sum_{j=1}^{K} P(l_j|l_i, l_k) = \sum_{j=1}^{K} \mathbf{S}_{kij} = 1$.

Given label transition observations, we aim to learn label transition probabilities $\mathbf{S}_{kij}$ maximizing a likelihood function of making the observations. Suppose we are given sets $\mathcal{D}_k$ of label transition observations represented as follows:

where n_k is the number of the observations in $\mathcal{D}_k$. $\mathbf{x}_{kh}$ is a label transition observation $\mathbf{x}_{khs} \xrightarrow{l_k} \mathbf{x}_{kht}$ such that $\mathbf{x}_{khs}, \mathbf{x}_{kht} \in \mathbf{L}$ where $\mathbf{x}_{khs}$ is the surfer's source label, and $\mathbf{x}_{kht}$ is her destination label. Let $P(\mathbf{x}_{kh}; \mathbf{S}_k)$ denote the probability of $\mathbf{x}_{kh}$ with regard to the parameter $\mathbf{S}_k$, i.e., $P(\mathbf{x}_{kh}; \mathbf{S}_k) = \mathbf{S}_k k \mathbf{x}_{khs} \mathbf{x}_{kht}$. Then, the log-likelihood function $L(\mathbf{S}_k; \mathcal{D}_k)$ is represented as follows:

$$L(\mathbf{S}_k; \mathcal{D}_k) = \log \prod_{h=1}^{n_k} P(\mathbf{x}_{kh}; \mathbf{S}_k) = \sum_{h=1}^{n_k} \log P(\mathbf{x}_{kh}; \mathbf{S}_k)$$

However, maximizing the classical likelihood $L(\mathbf{S}_k; \mathcal{D}_k)$ would be unsatisfactory since it is sensitive to noises or outliers in observations, and many networks such as knowledge graphs would be noisy and incomplete [6]. Considering this issue, we adopt maximum weighted likelihood estimation (MWLE) [40] that uses weights to vary the importance of $\log P(\mathbf{x}_{kh}; \mathbf{S}_k)$ for each observation. For the purpose, we define destination label weights $\{w_j|1 \le j \le K\}$ that weigh the importance of $\log P(\mathbf{x}_{kh}; \mathbf{S}_k)$ according to destination label $\mathbf{x}_{kht}$. The intuition is that destination labels play a key role in relation inference since the relation between a source node and a destination node u is determined by the surfer's (destination) label when she arrives at u after starting from the source node. Thus, our weighted log-likelihood function is defined as follows:

$$WL(\mathbf{S}_k; \mathcal{D}_k) = \sum_{h=1}^{n_k} w_{\mathbf{x}_{kht}} \log P(\mathbf{x}_{kh}; \mathbf{S}_k) \tag{1}$$

The following lemma provides the result of MWLE on the weighted log-likelihood function $WL(\mathbf{S}_k; \mathcal{D}_k)$.

Lemma 1 (Maximum Weighted Likelihood Estimation for Label Transition Probability Matrices) *The following estimator* $\hat{\mathbf{S}}_{kij}$ *maximizes the weighted log-likelihood function* $WL(\mathbf{S}_k; \mathcal{D}_k)$ *in* (1)*:*

$$\hat{\mathbf{S}}_{kij} = \frac{w_j N_{kij}}{\sum\limits_{z=1}^{K} w_z N_{kiz}} \tag{2}$$

where $N_{kij} = \sum_{h=1}^{n_k} \mathbf{1}(\mathbf{x}_{khs} = i, \mathbf{x}_{kht} = j)$ *is the count for the label transition observations* $l_i \xrightarrow{l_k} l_j$*, and* $\mathbf{1}(\cdot)$ *returns 1 if a given predicate is true, or 0 otherwise.*

Proof See the proof in Lemma 2 of Appendix A. □

Lemma 1 indicates that $\hat{\mathbf{S}}_{kij}$ is determined by the label weight w_j and the count N_{kij} for the observations. Suppose all label weights $\{w_j\}$ are fixed to 1 (i.e., no label weights). Then $\hat{\mathbf{S}}_k ij$ depends only on the given observations. The label weights $\{w_j\}$ are interpreted as the relative importances of the labels on the label transition probabilities $\hat{\mathbf{S}}_{kij}$. If we set w_j to a high value, the probability that the surfer's label changes to label l_j increases compared to other labels during her random walk. We select proper $\{w_j\}$ that provide the best performance in the validation sets, as described in Section 5.2.

4.3 Multi-Labeled Random Walk with Restart

We describe our proposed model Multi-Labeled Random Walk with Restart (MuRWR) in the following definition:

Definition 4 (Multi-Labeled Random Walk with Restart) A labeled random surfer has label l_i among K edge labels at a node except at source node s. The surfer starts from source node s without any label. Suppose the surfer is currently at node u, and c is the restart probability with $0 < c < 1$. Then, the surfer performs one of the followings at each step:

- **Multi-Labeled Random Walk.** The surfer randomly moves from node u to a neighboring node v with probability $1 - c$ through l_{uv}-labeled edge. If her label was l_i at node u, then her label changes to label l_j at node v according to label transition probability $P(l_i \xrightarrow{l_{uv}} l_j)$.
- **Restart.** The surfer restarts at source node s with probability c. The surfer becomes unlabeled.

MuRWR measures K probabilities at each node. Let $\mathbf{R}_{ui}$ denote the probability that l_i-labeled surfer is at node u after MuRWR from s. If $\mathbf{R}_{ui}$ is higher than $\mathbf{R}_u j$ for $j \neq i$, this indicates that l_i-labeled surfer frequently visits node u, implying source node s is highly related by edge label l_i to node u. Thus $\mathbf{R}_u i$ is used for a relevance score between s and u for label l_i.

Our model MuRWR also considers multi-faceted paths between s and u, which is controlled by the restart probability c. If c is low, the labeled surfer visits u via paths of various lengths since the surfer prefers random walks to restart. On the other hand, if c is high, the surfer mainly visits u through relatively short paths due to frequent restarts. We will empirically study the effect of c on relation inference in the Experiment section (see Figure 6).

Note that when the unlabeled surfer moves from source node s to node u, her label becomes l_{su} since they are directly related with label l_{su} as depicted in Figure 2a. To make notations consistent, we add a dummy label l_d indicating *unlabeled* at source node s. Then we set label transition probabilities $P(l_d \xrightarrow{l_{su}} l_{su})$ to 1 for each out-neighbor u of source node s (line 1 in Algorithm 3).

4.4 Formulation for MuRWR

We present formulations of MuRWR before proposing an algorithm for it. We derive a recursive equation for the probabilities $\mathbf{R}_{ui}$ (MuRWR scores).

Element-wise Representation The MuRWR scores at a node are recursively determined by in-neighbors of the node. Figure 4 shows an example of the formulation for MuRWR probabilities where the figure depicts the part of a graph having two edge labels l_1 and l_2. In Figure 4, we mark an (edge label, probability) pair on each edge where the probability is for surfer's choosing the corresponding edge.

Let $\mathbf{R}_{u1}^{(t)}$ denote the probability that l_1-labeled surfer visits node u at time t after starting from source node s. In order that the surfer visits node u with label l_1 at time t, her label should be changed into l_1 when the surfer moves to node u from one of in-neighbors of u. For example, when she moves from node v to node u, her label changes to label l_1 with probability $\mathbf{R}_{v1}^{(t-1)} P(l_2 \xrightarrow{l_1} l_1) + \mathbf{R}_{v2}^{(t-1)} P(l_2 \xrightarrow{l_1} l_1)$. Considering the restart action with c, $\mathbf{R}_{u1}^{(t)}$ is determined as:

$$\begin{aligned}\mathbf{R}_{u1}^{(t)} &= (1-c)\underbrace{\left[\frac{1}{2}\left(\mathbf{R}_{v1}^{(t-1)} P(l_1 \xrightarrow{l_1} l_1) + \mathbf{R}_{v2}^{(t-1)} P(l_2 \xrightarrow{l_1} l_1)\right) + \frac{1}{3}\left(\mathbf{R}_{w1}^{(t-1)} P(l_1 \xrightarrow{l_2} l_1) + \mathbf{R}_{w2}^{(t-1)} P(l_1 \xrightarrow{l_2} l_1)\right)\right]}_{\text{Labeled Random Walk}} \\ &\quad + c\underbrace{\mathbf{1}(u = s, l_1 = l_d)}_{\text{Restart}}\end{aligned}$$

where $\mathbf{1}(\cdot)$ returns 1 if a given predicate is true, or 0 otherwise. Note that $\mathbf{R}_{u2}^{(t)}$ is also determined similarly to the above equation.

The general equation for $\mathbf{R}_{ui}^{(t)}$ is represented as:

$$\begin{aligned}\mathbf{R}_{ui}^{(t)} &= (1-c)\sum_{v \in \overleftarrow{\mathbf{N}}_u}\left(\frac{1}{|\overrightarrow{\mathbf{N}}_v|}\sum_{j=1}^{K} \mathbf{R}_{vj}^{(t-1)} P(l_j \xrightarrow{l_{vu}} l_i)\right) \\ &\quad + c\mathbf{1}(u = s, l_i = l_d)\end{aligned} \tag{3}$$

where $\overleftarrow{\mathbf{N}}_u$ is the set of in-neighbors of node u, and $\overrightarrow{\mathbf{N}}_v$ is the set of out-neighbors of node v. Note that $\mathbf{R}_{ur}^{t}$ is the accumulated result of MuRWR until step t with decaying factor $1 - c$, as interpreted similarly in PageRank or RWR.

Matrix Representation We represent (3) in a matrix form using symbols in the following definitions:

Definition 5 (Labeled Semi-adjacency Matrix) The l_k-labeled semi-ad-jacency matrix $\mathbf{A}_k$ is a matrix such that (u, v)-th entry $\mathbf{A}_{kuv}$ of $\mathbf{A}_k$ is 1 if the label of the edge $u \rightarrow v$ is l_k, or 0 otherwise.

Definition 6 (Labeled Semi-row-normalized Matrix) Let $\mathbf{D}$ be the out-degree diagonal matrix of a graph G where $\mathbf{D}_{uu}$ is the out-degree of node u. Then l_k-labeled semi-row-normalized matrix $\tilde{\mathbf{A}}_k$ is defined by $\tilde{\mathbf{A}}_k = \mathbf{D}^{-1}\mathbf{A}_k$. In other words, $\tilde{\mathbf{A}}_{kuv} = |\overrightarrow{\mathbf{N}}_u|^{-1} = \mathbf{D}_{uu}^{-1}$ if there is l_k-labeled edge from node u to node v in the graph G, or 0 otherwise, where $\overrightarrow{\mathbf{N}}_u$ is the set of out-neighbors of node u.

Based on Definitions 3, 5 and 6, (3) is rewritten as follows:

$$\mathbf{R}^{(t)} = \underbrace{(1-c)\sum_{k=1}^{K}(\tilde{\mathbf{A}}_k^{\top}\mathbf{R}^{(t-1)}\mathbf{S}_k)}_{\text{Labeled Random Walk}} + \underbrace{c\mathbf{Q}_s}_{\text{Restart}} \tag{4}$$

where $\mathbf{R}^{(t)} \in \mathbb{R}^{n\times K}$ is an MuRWR score matrix such that each entry $\mathbf{R}_{ui}^{(t)}$ is a score between source node s and node u for edge label l_i at step t, and $\mathbf{Q}_s \in \mathbb{R}^{n\times K}$ is a single entry matrix whose (s, d)-th entry is 1, and other entries are 0, where the index d is for the dummy label l_d. We provide the detailed derivation from (3) to (4) in Lemma 3 of Appendix B.

Algorithm 1 Learning phase for MuRWR

```
Input: labeled adjacency matrix $\mathbf{A}$, and label weights $\{w_j\}$
Output: label transition probability matrices $\mathbf{S}_k$ for $1 \le k \le K$
1: enumerate all transitive triangles in the graph $G$ represented by $\mathbf{A}$ using a triangle enumeration algorithm [17] to collect label transition observations $\mathcal{D}_k$ for $1 \le k \le K$
2: set $N_{kij} \leftarrow 0$ for $1 \le i, j, k \le K$
3: for $k = 1$ to $K$ do
4:     for each $\mathbf{x}_{kh} = (\mathbf{x}_{khs} \xrightarrow{l_k} \mathbf{x}_{kht}) \in \mathcal{D}_k$ do
5:         set $i \leftarrow \mathbf{x}_{khs}$, $j \leftarrow \mathbf{x}_{kht}$, and $N_{kij} \leftarrow N_{kij} + 1$
6:     end for
7:     $\mathbf{S}_{kij} \leftarrow \frac{w_j N_{kij}}{\sum_{z=1}^{k} w_z N_{kiz}}$ for $1 \le i, j \le K$
8: end for
       return $\mathbf{S}_k$ for $1 \le k \le K$
```

Algorithm 2 Normalization phase for MuRWR

```
Input: labeled adjacency matrix $\mathbf{A}$
Output: labeled semi-row-normalized matrices $\tilde{\mathbf{A}}_k$ for $1 \le k \le K$
1: for each edge label $l_k$, construct semi-adjacency matrix $\mathbf{A}_k$ from
2: the graph $G$ represented by $\mathbf{A}$
3: compute adjacency matrix $\mathbf{A}' = \sum_k \mathbf{A}_k$ and out-degree diagonal matrix $\mathbf{D}$ such that $\mathbf{D}_{uu} = \sum_v \mathbf{A}'_{uv}$
4: for each edge label $l_k$, compute semi-row-normalized matrix $\tilde{\mathbf{A}}_k$,
5: i.e., $\tilde{\mathbf{A}}_k = \mathbf{D}^{-1}\mathbf{A}_k$
       return $\tilde{\mathbf{A}}_k$ for $1 \le k \le K$
```

Note that MuRWR is a generalized version of RWR, i.e., MuRWR works on edge-labeled graphs as well as plain graphs without edge labels, which is proved in the following.

Property 1 MuRWR produces the same result with RWR in a plain graph.

Proof This case is equivalent to when the number K of edge labels is 1. Then, $\mathbf{R}^{(t)}$ and $\mathbf{Q}_s$ are written as vectors $\mathbf{r}^{(t)}$ and $\mathbf{q}_s$ in $\mathbb{R}^{n\times1}$, respectively, where n is the number of nodes. Since $K = 1$, there is only one type of labeled transitive triangle; thus, $\mathbf{S}_1 \in \mathbb{R}^{1\times1} = 1$. Then, (4) is exactly the same with $\mathbf{r}^{(t)} = (1-c)\tilde{\mathbf{A}}^\top\mathbf{r}^{(t-1)} + c\mathbf{q}_s$, the recursive equation of RWR [38]. □

4.5 Algorithm for MuRWR

Learning Phase (Algorithm 1) Given a labeled adjacency matrix $\mathbf{A}$, the learning phase learns the label transition probability matrices $\mathbf{S}_k$. We enumerate all transitive triangles from the graph represented by $\mathbf{A}$ (line 1) using a triangle enumeration algorithm [17]. Based on the enumerated transitive triangles, we estimate the label transition matrices $\mathbf{S}_k$ using (2) (lines 2∼8).

Normalization Phase (Algorithm 2). This phase produces the semi-row-normalized matrices $\tilde{\mathbf{A}}_k$ for $1 \leq k \leq K$ from the labeled adjacency matrix $\mathbf{A}$ after building semi-adjacency matrices $\mathbf{A}_k$ by Definitions 5 and 6 (lines 1∼3).

Algorithm 3 Iterative Algorithm for MuRWR

```
Input: semi-row-normalized matrices $\tilde{\mathbf{A}}_k$, label transition probability matrices $\mathbf{S}_k$ for $1 \leq k \leq K$, source node $s$, restart probability $c$, and error tolerance $\epsilon$
Output: MuRWR relevance score matrix $\mathbf{R}$
1: for each node $u \in \overrightarrow{\mathbf{N}}_s$, set $\mathbf{S}_{kij} \leftarrow 1$ where $k$ and $j$ are the indices
2: for $l_{su}$, and $i$ is for $l_d$, i.e., set $P(l_d \xrightarrow{l_{su}} l_{su}) = 1$
3: set starting matrix $\mathbf{Q}_s$ from source node $s$ and $\mathbf{R}^{(0)} \leftarrow \mathbf{Q}_s$
4: set step $t \leftarrow 0$
5: repeat
6:     update $t \leftarrow t + 1$
7:     compute $\mathbf{R}^{(t)} \leftarrow (1-c)\sum_{k=1}^{K}(\tilde{\mathbf{A}}_k^\top \mathbf{R}^{(t-1)}\mathbf{S}_k) + c\mathbf{Q}_s$
8:     compute residual $\delta^{(t)} = \|\mathbf{R}^{(t)} - \mathbf{R}^{(t-1)}\|_{1,1}$
9: until $\delta^{(t)} < \epsilon$
       return MuRWR relevance score matrix $\mathbf{R} \leftarrow \mathbf{R}^{(t)}$
```

Iteration Phase (Algorithm 3) This phase computes the MuRWR relevance score matrix $\mathbf{R}$ w.r.t. source node s. We first set $P(l_d \xrightarrow{l_{su}} l_{su}) = 1$ for each $u \in \overrightarrow{\mathbf{N}}_s$ for the dummy label l_d (line 1). After setting starting matrix $\mathbf{Q}_s$ and initializing $\mathbf{r}^{(0)}$ to $\mathbf{Q}_s$ (line 2), we repeat the update for $\mathbf{r}^{(t)}$ based on (4) until convergence (lines 4∼8). We compute residual $\delta^{(t)}$ between $\mathbf{r}^{(t)}$ and $\mathbf{r}^{(t-1)}$ which is the result from the previous iteration (line 7) where $\delta^{(t)}$ is measured by entry-wise L1 matrix norm, i.e., $\|\mathbf{A}\|_{1,1} = \sum_{i,j}|\mathbf{A}_{ij}|$. This stops when $\delta^{(t)}$ is smaller than error tolerance ϵ.

Convergence Analysis We prove the convergence guarantee of the iterative method (Algorithm 3) of MuRWR in Theorem 1.

Theorem 1 (Convergence of MuRWR) *Suppose* $\mathbf{r}^{(t)} = vec(\mathbf{R}^{(t)})$ *and* $\mathbf{q}_s = vec(\mathbf{Q}_s)$ *where* $\mathbf{R}^{(t)}$ *is the* MuRWR *score matrix at step t in Algorithm 3, and* $vec(\cdot)$ *is the vec-operator which converts a matrix into a vector* [18]. *Then the residual* $\delta^{(t)} \leq 2(1-c)^t$, *and* $\mathbf{r}^{(t)}$

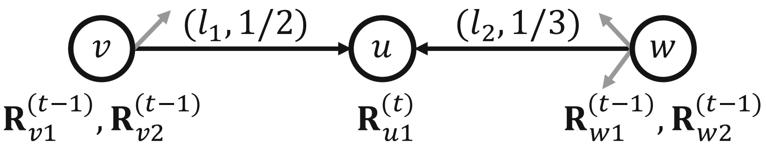

$(l_1, 1/2)$
$(l_2, 1/3)$
v
u
w
$\mathbf{R}_{v1}^{(t-1)}, \mathbf{R}_{v2}^{(t-1)}$
$\mathbf{R}_{u1}^{(t)}$
$\mathbf{R}_{w1}^{(t-1)}, \mathbf{R}_{w2}^{(t-1)}$

Figure 4 Example of the formulation for the probability $\mathbf{R}_{u1}^{(t)}$ that l_1-labeled surfer visits node u at time t

converges to $\mathbf{r} = c(\mathbf{I} - (1-c)\tilde{\mathbf{B}}^\top)^{-1}\mathbf{q}_s$ *where* $\tilde{\mathbf{B}}^\top = \sum_{k=1}^{K} \mathbf{S}_K^\top \otimes \tilde{\mathbf{A}}_k^\top$ *and* $\otimes$ *is Kronecker product.*

Proof Equation (4) is vectorized by $vec(\cdot)$ as follows:

$$\begin{aligned}
vec(\mathbf{R}^{(t)}) &= (1-c)\sum_{k=1}^{K} vec(\tilde{\mathbf{A}}_k^\top \mathbf{R}^{(t-1)}\mathbf{S}_k) + c(vec(\mathbf{Q}_s)) \\
&= (1-c)\sum_{k=1}^{K} (\mathbf{S}_k^\top \otimes \tilde{\mathbf{A}}_k^\top) vec(\mathbf{R}^{(t-1)}) + c(vec(\mathbf{Q}_s)) \\
\Leftrightarrow \mathbf{r}^{(t)} &= (1-c)\tilde{\mathbf{B}}^\top \mathbf{r}^{(t-1)} + c\mathbf{q}_s \qquad (5)
\end{aligned}$$

where $\tilde{\mathbf{B}}^\top = \sum_{k=1}^{K} \mathbf{S}_k^\top \otimes \tilde{\mathbf{A}}_k^\top$. The second equation is derived by $vec(\mathbf{ABC}) = (\mathbf{C}^\top \otimes \mathbf{A})vec(\mathbf{B})$ [18].

Residual analysis The residual $\delta^{(t)} = \|\mathbf{R}^{(t)} - \mathbf{R}^{(t-1)}\|_{1,1}$ is equal to $\|\mathbf{r}^{(t)} - \mathbf{r}^{(t-1)}\|_1$ since $\|\mathbf{A}\|_{1,1} = \|vec(\mathbf{A})\|_1$ [18]. Thus, the residual is bounded as follows:

$$\begin{aligned}
\delta^{(t)} &= \|\mathbf{r}^{(t)} - \mathbf{r}^{(t-1)}\|_1 = \|(1-c)\tilde{\mathbf{B}}^\top \mathbf{r}^{(t-1)} - (1-c)\tilde{\mathbf{B}}^\top \mathbf{r}^{(t-2)}\|_1 \\
&\le (1-c)\|\tilde{\mathbf{B}}^\top\|_1 \|\mathbf{r}^{(t-1)} - \mathbf{r}^{(t-2)}\|_1 \\
&\le (1-c)\|\mathbf{r}^{(t-1)} - \mathbf{r}^{(t-2)}\|_1 \le \cdots \\
&\le (1-c)^{t-1}\|\mathbf{r}^{(1)} - \mathbf{r}^{(0)}\|_1 = (1-c)^t \|\tilde{\mathbf{B}}^\top \mathbf{q}_s - \mathbf{q}_s\|_1 \le 2(1-c)^t
\end{aligned}$$

Note that $\|\tilde{\mathbf{B}}^\top \mathbf{q}_s - \mathbf{q}_s\|_1 \le 2$ since $\|\tilde{\mathbf{B}}^\top \mathbf{q}_s - \mathbf{q}_s\|_1 \le \|\tilde{\mathbf{B}}^\top \mathbf{q}_s\|_1 + \|\mathbf{q}_s\|_1 \le \|\tilde{\mathbf{B}}^\top\|_1 \|\mathbf{q}_s\|_1 + 1 \le 2$ where $\|\mathbf{q}_s\|_1 = 1$, and $\|\tilde{\mathbf{B}}^\top\|_1 \le 1$ by Lemma 6. Hence, the bound for the residual is $\delta^{(t)} \le 2(1-c)^t$.

Convergence analysis Equation (5) is also represented as follows:

$$\begin{aligned}
\mathbf{r}^{(t)} &= (1-c)\tilde{\mathbf{B}}^\top \mathbf{r}^{(t-1)} + c\mathbf{q}_s \\
&= \left((1-c)\tilde{\mathbf{B}}^\top\right)^2 \mathbf{r}^{(t-2)} + c\left((1-c)\tilde{\mathbf{B}}^\top + \mathbf{I}\right)\mathbf{q}_s = \cdots \\
&= \left((1-c)\tilde{\mathbf{B}}^\top\right)^t \mathbf{r}^{(0)} + c\sum_{j=0}^{t-1}\left((1-c)\tilde{\mathbf{B}}^\top\right)^j \mathbf{q}_s
\end{aligned}$$

The spectral radius $\rho((1-c)\tilde{\mathbf{B}}^\top) \le (1-c) < 1$ when $0 < c < 1$ since $\rho(\tilde{\mathbf{B}}^\top) \le 1$ according to Lemma 6 of Appendix C. Hence, $\lim_{t\to\infty}((1-c)\tilde{\mathbf{B}}^\top)^t \mathbf{r}^{(0)} = \mathbf{0}$ [35] and $\lim_{t\to\infty} \mathbf{r}^{(k)}$ converges as:

$$\lim_{t\to\infty} \mathbf{r}^{(t)} = c\sum_{j=0}^{\infty}\left((1-c)\tilde{\mathbf{B}}^\top\right)^j \mathbf{q}_s = c(\mathbf{I} - (1-c)\tilde{\mathbf{B}}^\top)^{-1}\mathbf{q}_s \qquad (6)$$

$\sum_{j=0}^{\infty}((1-c)\tilde{\mathbf{B}}^{\top})^j$ is a geometric series of the matrix $(1-c)\tilde{\mathbf{B}}^{\top}$, and it converges to $(\mathbf{I}-(1-c)\tilde{\mathbf{B}}^{\top})^{-1}$ since the spectral radius of $(1-c)\tilde{\mathbf{B}}^{\top}$ is less than 1 [35]. Note that $\mathbf{I}-(1-c)\tilde{\mathbf{B}}^{\top}$ is invertible when the spectral radius $\rho((1-c)\tilde{\mathbf{B}}^{\top}) \le (1-c) < 1$ [35]. □

Theorem 1 implies that as step t increases, the residual $\delta^{(t)}$ in Algorithm 3 monotonically decreases, and converges to zero since $0 < c < 1$ as specified in Definition 4. Also, $\mathbf{r}^{(t)}$ converges to a unique solution $\mathbf{r} = c(\mathbf{I}-(1-c)\tilde{\mathbf{B}}^{\top})^{-1}\mathbf{q}_s$.

According to Theorem 1, MuRWR scores in $\mathbf{r} = vec(\mathbf{R})$ can be directly obtained from (6). However, the matrix inversion of $\mathbf{I}-(1-c)\mathbf{r}^{(B)}$ requires $O((Kn)^3)$ time because the dimension of $\mathbf{r}^{(B)}$ is $Kn \times Kn$, and matrix inversion exhibits cubic time (i.e., $O(t^3)$ for $t \times t$ matrix) [35] where K is the number of edge labels and n is the number of nodes. Thus, the matrix inversion is not scalable, especially when n is large. On the other hand, our iterative approach in Algorithm 3 shows a linear scalability w.r.t. m and n, i.e., $O(T(Km + K^3n))$ where T is the number of iterations and m is the number of edges, which is much more efficient than the matrix inversion. We analyze the detailed time complexities of the algorithms of MuRWR in Appendix D.

5 Experiment

We perform experiments to answer the following questions:

- **Q1. Performance of Relation Inference (Section 5.2).** How accurately does MuRWR predict edge labels between nodes compared to other existing methods?
- **Q2. Effects of Label Weights (Section 5.3).** How does the label weights w_j in MuRWR affect the predictive performance of MuRWR for the relation inference task?
- **Q3. Effects of Restart Probability (Section 5.4).** How does the restart probability c in MuRWR affect the inference performance of MuRWR?
- **Q4. Convergence (Section 5.5).** Does the iterative algorithm for MuRWR converge? How does the restart probability c affect the convergence behavior of the algorithm?

5.1 Experimental settings

Datasets The datasets used for our experiments are summarized in Table 2. In the Epinions [25] and the Slashdot [15] datasets, users rate each other positively or negatively. The WN11 [33] and WN18 [3] datasets are from WordNet [27], a knowledge graph of words where an edge label is a relation between words. The WikiVote dataset is a signed network where users vote positively or negatively on their candidates [20]. The Advogato dataset is a social network where an edge label indicates a level of trust [26].

Competitors We compare our proposed model MuRWR to other existing methods which are categorized as follows:

- **Random**: Random predicts the label of a test edge randomly, which provides the worst performance of each dataset.
- **LINE** [36] and **node2vec** [8]: We exploit well-known embedding models LINE and node2vec as baseline methods, although they are not designed for relation inference. Since they are designed for plain networks, we first extract an embedding vector of each

Table 2 Dataset statistics. n is the number of nodes, m is the number of edges, and K is the number of edge labels

Dataset	n	m	K	Description
Epinions[1]	131,828	841,372	2	Signed social network
Slashdot[2]	79,120	515,397	2	Signed social network
WN18[3]	40,943	151,433	18	Knowledge graph
WN11[4]	38,588	138,887	11	Knowledge graph
WikiVote[5]	7,118	103,675	2	Signed voting network
Advogato[6]	6,541	47,135	3	Social network

[1]http://www.trustlet.org/wiki/Extended_Epinions_dataset

[2]http://dai-labor.de/IRML/datasets

[3]https://everest.hds.utc.fr/doku.php?id=en:transe

[4]http://cs.stanford.edu/~danqi/data/nips13-dataset.tar.bz2

[5]http://snap.stanford.edu/data/wiki-Vote.html

[6]http://konect.uni-koblenz.de/networks/advogato

node using those methods in a given network without edge labels. Next, we convert l_i-edge between two nodes into a training instance, i.e., the feature is the concatenation of the embeddings of those nodes, and the label is l_i. We perform multinomial logistic regression based on a softmax function to predict the edge label.

– **MRWR** [32] and **SRWR** [12]:
  We compare MuRWR to MRWR and SRWR which are RWR based variants. Although MRWR is originally designed for signed networks, it is easy to make the method work on edge-labeled graphs by computing RWR on each subgraph containing only a specific edge label. Note that SRWR cannot work on general edge-labeled graphs since it is designed only for signed networks (i.e., $K = 2$); hence, the results of SRWR for other edge-labeled graphs (i.e., $K > 2$) are omitted in Tables 3 and 4.
– **PRA** [16]: PRA is a path feature model used for relation inference in edge-labeled graphs. PRA extracts path features between two nodes using a random surfer, and exploits those features with logistic regressors.
– **TransE** [4] and **TransR** [24]: TransE is a translation based method which considers the relation l between nodes s and t as a translation between the corresponding node embeddings. Specifically, TransE discovers embeddings $\mathbf{s}$, $\mathbf{l}$, and $\mathbf{t}$ minimizing $f(s, l, t) = \|\mathbf{s} + \mathbf{l} - \mathbf{t}\|$. Given nodes s and t, it predicts their relation l that minimizes $f(s, l, t)$. TransR models entities and relations in distinct spaces, i.e., $\mathbf{s}_l = \mathbf{s}\mathbf{M}_l$ and $\mathbf{t}_l = \mathbf{t}\mathbf{M}_l$ where $\mathbf{M}_l$ is projection matrix for l. Then, TransR minimizes $f(s, l, t) = \|\mathbf{s}_l + \mathbf{l} - \mathbf{t}_l\|$.
– **DistMult** [41]: DistMult is a general framework modeling triples (s, l, t); it unifies existing neural tensor and translating models for relational learning.
– **PTransE** [23]: PTransE extends TransE, and models relational paths between entities. Suppose entities s and t are connected by a path $s \xrightarrow{l_1} e_1 \xrightarrow{l_2} t$. PTransE regards this path as another relation consisting of l_1 and l_2 (i.e., $p = (l_1, l_2)$), and minimizes $f(s, p, t) = \|\mathbf{s} + (\mathbf{l}_1 \circ \mathbf{l}_2) - \mathbf{t}\|$ where $\circ$ is an operation such as addition or multiplication to join these relations. PTransE exploits a network flow technique to extract relational paths to be trained.

Table 3 Performance of relation inference in terms of accuracy. The best method is in bold, and the second best method is in italic. Our proposed model MuRWR (marked †) shows the best performance in accuracy

Methods	WikiVote	Slashdot	Epinions	Advogato	WN11	WN18
Random	0.497	0.500	0.493	0.340	0.090	0.078
LINE [36]	0.781	0.771	0.903	0.552	0.489	0.404
node2vec [8]	0.779	0.765	0.905	0.586	0.426	0.401
MRWR [32]	0.805	0.769	0.890	0.550	0.194	0.342
SRWR [12]	*0.825*	0.790	0.906	–	–	–
PRA [16]	0.813	0.804	0.913	*0.683*	0.580	0.556
TransE [4]	0.793	0.802	0.902	0.644	*0.617*	*0.653*
TransR [24]	0.800	0.757	0.874	0.672	0.609	0.530
PTransE [23]	0.775	0.754	0.864	0.639	0.611	0.607
COMP [9]	0.775	0.752	0.868	0.633	0.601	0.610
DistMult [41]	0.799	*0.811*	*0.921*	0.615	0.612	0.621
MuRWR[†]	**0.830**	**0.820**	**0.929**	**0.727**	**0.641**	**0.689**

– **COMP** [9]: COMP is a unified framework to model relational paths by generalizing existing embedding approaches such as TransE and bilinear models. COMP samples relational paths by performing random walks on an edge-labeled graph.

5.2 Relation inference task

We evaluate our proposed model MuRWR on a relation inference task defined as follows: given an edge-labeled graph containing missed labels of edges, predict those edge labels. We randomly select 500 source nodes and choose 20% of out-going edges from each source node as a validation set which is used for selecting proper hyper-parameters of each method. For a test set, we randomly select another 500 source nodes and choose 20% of out-going

Table 4 Performance of relation inference in terms of macro F1-score. The best method is in bold, and the second best method is in italic. Our proposed model MuRWR (marked †) shows better inference performance than other methods on most datasets in terms of macro F1-score

Methods	WikiVote	Slashdot	Epinions	Advogato	WN11	WN18
Random	0.504	0.502	0.501	0.334	0.094	0.059
LINE [36]	0.524	0.592	0.706	0.479	0.204	0.202
node2vec [8]	0.519	0.583	0.673	0.532	0.256	0.206
MRWR [32]	0.703	0.661	0.809	*0.698*	0.149	0.131
SRWR [12]	*0.742*	0.730	0.822	–	–	–
PRA [16]	0.733	0.692	0.815	0.682	0.569	0.536
TransE [4]	0.725	0.687	0.821	0.648	*0.571*	*0.642*
PTransE [23]	0.635	0.674	0.702	0.574	0.539	0.565
COMP [9]	0.615	0.616	0.641	0.581	0.525	0.567
DistMult [41]	0.730	**0.770**	**0.841**	0.594	0.559	0.566
MuRWR[†]	**0.746**	*0.748*	*0.830*	**0.723**	**0.594**	**0.660**

edges from each source node. We then remove each selected edge $s \rightarrow t$, and predict the edge's label using relevance scores w.r.t. source node s, i.e., the predicted label $\hat{l}$ is decided as follows: $\hat{l} = \arg\max_{1 \le k \le K} \mathbf{R}_{tk}$. We measure the prediction accuracy for the test dataset, which is defined as follows: *accuracy = # of correct predictions/# of test edges*. This task is also considered as multi-class classification (i.e., classify the label of a test edge); thus, we measure macro F1-score which is a multi-class classification accuracy [34]. We repeat the above procedure 10 times, and report the average accuracy over the multiple runs for each method.

For the experiment, we select label weights w_j which provide the best inference in the validation set. To search for the label weights, we adopt the forward stepwise selection strategy [28]. Suppose we consider three weights w_1, w_2, and w_3 initialized to 1. For w_1, we fix other weights w_2 and w_3, and choose the value of w_1 which provides the best accuracy on the validation set when varying w_1 in the range [0, 2] by step size 0.2. With the selected value of w_1, we fix w_3, and repeat the above procedure for w_2. We finally perform the stepwise search for w_3 with the selected values of w_1 and w_2. For other methods, we select proper hyper-parameters of the methods via a grid search, which provide the best accuracy on the validation set. For PRA, PTransE, and COMP, we set the maximum path length to 3 Note that we cannot perform relation inference using relevance scores measured by standard RWR as described in Section 1.

Tables 3 and 4 show the performance of MuRWR compared to other methods. Our model MuRWR provides the best accuracy for predicting edge labels: MuRWR obtains $0.7 \sim 6.1\%$ relative improvement on accuracy over the second best method. Also, the performance of MuRWR is higher than that of other methods in terms of macro F1-score for most datasets. MuRWR outperforms the second best method by up to $0.5 \sim 4\%$ on the datasets except the Slashdot and Epinions datasets. For these two datasets, although the F1-score of DistMult is higher than that of MuRWR, our method exhibits the second best performance.

These results indicate that relevance scores computed by MuRWR are effective for predicting edge labels.

5.3 Effects of label weights in MuRWR

We examine the impact of label weights in MuRWR on the performance of the relation inference task. We use MuRWR-F (a version of MuRWR where all weights $\{w_j\}$ are fixed to 1) and MuRWR using the forward stepwise strategy as described above.

We measure the performance of the relation inference task in terms of accuracy and F1-score.

As seen in Figure 5, the inference performance of MuRWR is better than that of MuRWR-F in terms of accuracy and F1-score, respectively.

This indicates that adjusting the label weights is helpful for the performance of MuRWR in the relation inference task.

5.4 Effects of restart probability in MuRWR

We investigate the effect of restart probability c of MuRWR. We measure the inference performance of MuRWR in terms of accuracy varying c from 0.01 to 0.99. As shown in Figure 6, the performance of MuRWR with $c = 0.15 \sim 0.3$ is better than that of MuRWR when c is too low or high. Note that c controls how far the surfer walks from source node s. If a value of c is high, the surfer frequently jumps back to the source node; thus, the

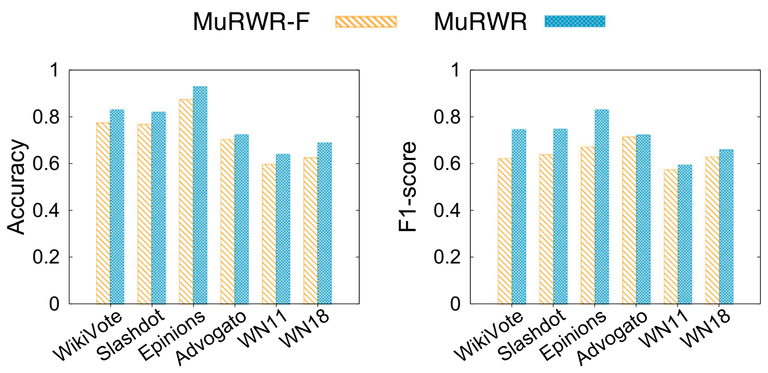


Figure 5 Effect of the label weights in MuRWR. The performance of MuRWR is better than that of MuRWR-F (i.e., MuRWR where all label weights are fixed to 1), verifying that introducing the weights improves the performance of MuRWR

relevance score between node s and a target node t are highly affected by paths of short length from s to t, restricting the model's complexity severely. Hence, an extremely high value of c such as 0.99 has a bad influence on the performance of MuRWR. On the other hand, a too low value of c such as 0.01 also does not have a positive impact on relational reasoning since such a low value effectively ignores the source vertex. With a proper value of c between 0.15 and 0.3, MuRWR provides the best performance, and outperforms other methods as shown in Tables 3 and 4.

5.5 Convergence of MuRWR

We explore the convergence behavior of MuRWR described in Theorem 1. We vary the restart probability c between 0.1 and 0.9, and measure the residual $\delta^{(t)}$ in Algorithm 3 until convergence with error tolerance $\epsilon = 10^{-14}$. As seen in Figure 7, the residual $\delta^{(t)}$ monotonically decreases for $0 < c < 1$ as step t increases, and then, $\delta^{(t)}$ finally becomes less than ϵ. Another observation is that a high value of c accelerates the convergence rate for the residual $\delta^{(t)}$, e.g., $c = 0.9$ makes MuRWR converge faster than $c = 0.1$. The reason is that by Theorem 1, $\delta^{(t)} \leq 2(1-c)^t$; thus, the higher c is, the faster the residual $\delta^{(t)}$ goes

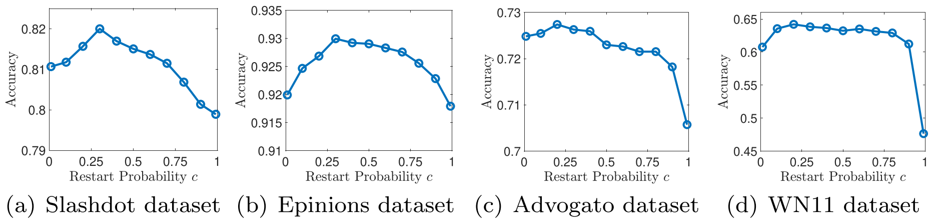
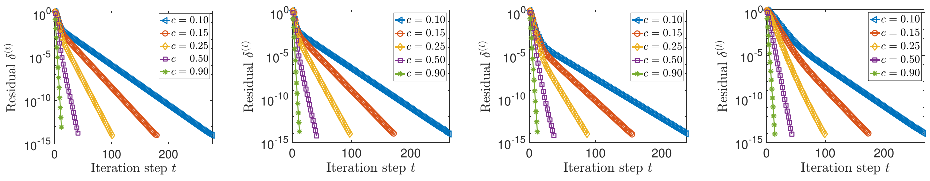


Figure 6 Effect of the restart probability c in MuRWR. The inference performance of MuRWR with $c = 0.15 \sim 0.3$ is better than that of MuRWR when c is too low or high



(a) Slashdot dataset (b) Epinions dataset (c) Advogato dataset (d) WN11 dataset

Figure 7 Convergence of MuRWR. The residual $\delta^{(t)}$ in Algorithm 3 monotonically decreases in the datasets for $0 < c < 1$ where c is the restart probability in MuRWR

toward zero. Note that an extremely high value of c such as 0.99 degrades the performance of MuRWR as shown in Figure 6, while a too low value of c requires many iterations to converge as shown in Figure 7. A value of c between 0.15 and 0.3 provides a good trade-off between inference performance and the number of iterations to converge.

6 Conclusion

We propose MuRWR (Multi-Labeled Random Walk with Restart), a novel random walk based model which accurately infers edge labels between nodes by computing relevance scores between a source node and other nodes for each edge label in edge-labeled graphs. We introduce a labeled random surfer to consider labeled edges for multi-hop relational reasoning. We provide a learning procedure based on label transitive relationships inherent in a given graph, and theoretically analyze our method including its convergence. We also show that MuRWR is a generalized version of RWR.

Experiments show MuRWR gives the best accuracy in the relation inference task. Future works include learning the label weights of MuRWR from a given edge-labeled graph, and extending the method for graphs with complex node labels.

Acknowledgements This work was supported by the ICT R&D program of MSIT/IITP (No.2017-0-01772, Development of QA systems for Video Story Understanding to pass the Video Turing Test). The Institute of Engineering Research at Seoul National University provided research facilities for this work. The ICT at Seoul National University provides research facilities for this study.

Appendix A: Lemma for convexity and solution of label transition probabilities

Lemma 2 *The estimator in* (2) *maximizes the weighted log-likelihood function* $WL(\mathbf{S}_k; \mathcal{D}_k)$ *in* (1).

Proof Our goal is to find $\mathbf{S}_k$ that maximizes $WL(\mathbf{S}_k; \mathcal{D}_k)$, which is equivalent to minimizing $-WL(\mathbf{S}_k; \mathcal{D}_k)$. The probability $P(\mathbf{x}_{kh}; \mathbf{S}_k)$ is written as follows:

$$P(\mathbf{x}_{kh}; \mathbf{S}_k) = \mathbf{S}_{k\mathbf{x}_{khs}\mathbf{x}_{kht}} = \prod_{i=1}^{K}\prod_{j=1}^{K}(\mathbf{S}_{kij})^{\mathbf{1}(\mathbf{x}_{khs}=i, \mathbf{x}_{kht}=j)}$$

Then $-WL(\mathbf{S}_k;\mathcal{D}_k)$ is represented as follows:

$$\begin{aligned}-WL(\mathbf{S}_k;\mathcal{D}_k) &= -\sum_{h=1}^{n_k} w_{\mathbf{x}_{kht}} \log P(\mathbf{x}_{kh};\mathbf{S}_k)\\ &= -\sum_{h=1}^{n_k}\sum_{i=1}^{K}\sum_{j=1}^{K} w_j \mathbf{1}(\mathbf{x}_{khs}=i,\mathbf{x}_{kht}=j)\log \mathbf{S}_{kij}\\ &= -\sum_{i=1}^{K}\sum_{j=1}^{K} w_j \left(\sum_{h=1}^{n_k}\mathbf{1}(\mathbf{x}_{khs}=i,\mathbf{x}_{kht}=j)\right)\log \mathbf{S}_{kij}\\ &= -\sum_{i=1}^{K}\sum_{j=1}^{K} w_j N_{kij}\log \mathbf{S}_{kij}\end{aligned}$$

where $N_{kij}=\sum_{h=1}^{n_k}\mathbf{1}(\mathbf{x}_{khs}=i,\mathbf{x}_{kht}=j)$ is the count of the label transition observations. Then the minimization problem is represented as follows:

$$\begin{aligned}\underset{\mathbf{S}_{kij}}{\text{minimize}} - WL(\mathbf{S}_k;\mathcal{D}_k) &= -\sum_{i=1}^{K}\sum_{j=1}^{K} w_j N_{kij}\log \mathbf{S}_{kij}\\ \text{subject to}\mathbf{S}_{kij} &\geq 0 \text{ for } 1\leq i,j\leq K,\\ \sum_{j=1}^{K}\mathbf{S}_{kij} &= 1 \text{ for } 1\leq i\leq K. \end{aligned}\tag{7}$$

Note that the above problem is convex (see Lemma 3); thus, the optimization problem is solved by the KKT theorem [5], and the solution of the problem is represented as (2) (details in Lemma 4. □

Lemma 3 *The optimization problem in* (7) *is convex.*

Proof The objective function is convex since the negative log functions $-\log \mathbf{S}_{kij}$ are convex, and the sum of non-negatively weighted convex functions is convex (i.e., $w_j N_{kij}\geq 0$) [5]. Let $\mathbf{C}$ be a set of $\mathbf{S}_k$ satisfying the constraints, i.e., $\mathbf{C}=\{\mathbf{S}_k|\mathbf{S}_k\mathbf{1}=\mathbf{1}, \mathbf{S}_{kij}\geq 0, \text{ for } 1\leq i,j\leq K\}$. For $\mathbf{S}_{k_1},\mathbf{S}_{k_2}\in\mathbf{C}$ and $\theta_1+\theta_2=1$ such that $\theta_1,\theta_2\geq 0$, let $\mathbf{S}_{k_3}=\theta_1\mathbf{S}_{k_1}+\theta_2\mathbf{S}_{k_2}$. Then $\mathbf{S}_{k_3}\mathbf{1}=(\theta_1\mathbf{S}_{k_1}+\theta_2\mathbf{S}_{k_2})\mathbf{1}=\mathbf{1}$ indicating $\mathbf{S}_{k_3}\in\mathbf{C}$. Thus $\mathbf{C}$ is convex by the definition of convex set [5]. □

Lemma 4 *The solution of the optimization problem in* (7) *is represented as* (2).

Proof The lagrangian $\mathcal{L}(\cdot)$ of the objective function in (7) is represented as follows:

$$\begin{aligned}\mathcal{L}(\mathbf{S}_k,\lambda,\nu) = -\sum_{i=1}^{K}\sum_{j=1}^{K} w_j N_{kij}\log \mathbf{S}_{kij} + \sum_{i=1}^{K}\sum_{j=1}^{K} -\lambda_{ij}\mathbf{S}_{kij}\\ + \sum_{i=1}^{K}\nu_i\sum_{j=1}^{K}\left(\mathbf{S}_{kij}-1\right)\end{aligned}$$

where λ and ν are inequality and equality lagrange multipliers, respectively. Let $\hat{\mathbf{S}}_{kij}$ be the solution that minimizes (7). λ^* and ν^* denote the optional points for λ and ν, respectively.

The stationarity condition $\nabla_{\mathbf{S}_k}\mathcal{L}(\hat{\mathbf{S}}_k, \lambda^*, \nu^*) = 0$ implies the following equation:

$$\frac{\partial\mathcal{L}(\hat{\mathbf{S}}_k, \lambda^*, \nu^*)}{\partial\mathbf{S}_{kij}} = -\frac{w_j N_{kij}}{\hat{\mathbf{S}}_{kij}} - \lambda^*_{ij} + \nu^*_i = 0 \Leftrightarrow \hat{\mathbf{S}}_{kij} = \frac{w_j N_{kij}}{\nu^*_i - \lambda^*_{ij}}$$

By the complementary slackness $\lambda^*_{ij}\hat{\mathbf{S}}_{kij} = 0$, primal feasibility $\hat{\mathbf{S}}_{kij} \geq 0$, and dual feasibility $\lambda^*_{ij} \geq 0$,

- $\hat{\mathbf{S}}_{kij} > 0 \Rightarrow \lambda^*_{ij} = 0 \Leftrightarrow \hat{\mathbf{S}}_{kij} = \dfrac{w_j N_{kij}}{\nu^*_i} > 0 \Leftrightarrow w_j N_{kij} \neq 0$
- $\lambda^*_{ij} > 0 \Rightarrow \hat{\mathbf{S}}_{kij} = 0 \Leftrightarrow \hat{\mathbf{S}}_{kij} = \dfrac{w_j N_{kij}}{\nu^*_i - \lambda^*_{ij}} = 0 \Leftrightarrow w_j N_{kij} = 0$

For the case that $\hat{\mathbf{S}}_{kij} > 0$, ν^*_i is obtained from the equality constraint $\sum_{z=1}^{K}\hat{\mathbf{S}}_{kiz} = 1$ as follows:

$$\sum_{z=1}^{K}\hat{\mathbf{S}}_{kiz} = \sum_{\{z|\hat{\mathbf{S}}_{kiz}>0\}}\hat{\mathbf{S}}_{kiz} = \sum_{\{z|\hat{\mathbf{S}}_{kiz}>0\}}\frac{w_z N_{kiz}}{\nu^*_i} = 1 \Leftrightarrow$$

$$\nu^*_i = \sum_{\{z|\hat{\mathbf{S}}_{kiz}>0\}} w_z N_{kiz} = \sum_{\{z|\hat{\mathbf{S}}_{kiz}>0\}} w_z N_{kiz} + \sum_{\{z|\hat{\mathbf{S}}_{kiz}=0\}} w_z N_{kiz} = \sum_{z=1}^{K} w_z N_{kiz}$$

Hence, $\hat{\mathbf{S}}_{kij} = w_j N_{kij}/\nu^*_i = w_j N_{kij}/(\sum_{z=1}^{K} w_z N_{kiz})$. □

Appendix B: Lemma for recursive equation of MuRWR score matrix

Lemma 5 *Equation* (3) *is represented as* (4).

Proof In (3), let l_p denote l_{vu}. For edge $v \to u$,

$$l_i) = \sum_{j=1}^{K}\mathbf{R}_{vj}P(l_j \xrightarrow{l_p} l_i) = \sum_{j=1}^{K}\mathbf{R}_{vj}\mathbf{S}_{pji}$$

where $\mathbf{S}_{pji}$ is the label transition probability $P(l_j \xrightarrow{l_p} l_i)$. By Definition 6, $\tilde{\mathbf{A}}_{pvu} = |\overrightarrow{\mathbf{N}}_v|^{-1} = \tilde{\mathbf{A}}^\top_{puv}$ for all p when $\tilde{\mathbf{A}}_{puv}$ is non-zero. Hence,

$$\sum_{v\in\overleftarrow{\mathbf{N}}_u}\left(\frac{1}{|\overrightarrow{\mathbf{N}}_v|}\sum_{j=1}^{K}\mathbf{R}_{vj}\mathbf{S}_{pji}\right) = \sum_{v\in\overleftarrow{\mathbf{N}}_u}\tilde{\mathbf{A}}^\top_{puv}\sum_{j=1}^{K}\mathbf{R}_{vj}\mathbf{S}_{pji} \tag{8}$$

Let $\overleftarrow{\mathbf{N}}^{(i)}_u$ be the set of in-neighbors of node u such that node $v \in \overleftarrow{\mathbf{N}}^{(i)}_u$ is connected to node u with edge label l_i. Then $\overleftarrow{\mathbf{N}}b_u$ is represented as $\overleftarrow{\mathbf{N}}_{uh} = \overleftarrow{\mathbf{N}}^{(1)}_u \cup \cdots \cup \overleftarrow{\mathbf{N}}^{(K)}_u$.

If there is no l_i-labeled edge to node u from any in-neighbor node, Thus (8) is represented as follows:

$$\begin{aligned}
&\sum_{v \in \overleftarrow{\mathbf{N}}_u} \tilde{\mathbf{A}}^\top_{puv} \sum_{j=1}^{K} \mathbf{R}_{vj} \mathbf{S}_{pji} \\
&= \sum_{v \in \overleftarrow{\mathbf{N}}_u^{(1)}} \tilde{\mathbf{A}}^\top_{1uv} \sum_{j=1}^{K} \mathbf{R}_{vj} \mathbf{S}_{1ji} + \cdots + \sum_{v \in \overleftarrow{\mathbf{N}}_u^{(K)}} \tilde{\mathbf{A}}^\top_{Kuv} \sum_{j=1}^{K} \mathbf{R}_{vj} \mathbf{S}_{Kji} \\
&= \sum_{k=1}^{K} \left(\sum_{v \in \overleftarrow{\mathbf{N}}_u^{(k)}} \tilde{\mathbf{A}}^\top_{kuv} \sum_{j=1}^{K} \mathbf{R}_{vj} \mathbf{S}_{kji} \right) \qquad (9)
\end{aligned}$$

Let $(\cdot)_{ij}$ be (i, j)-th entry of a matrix. Then, $\sum_{v \in \overleftarrow{\mathbf{N}}_u^{(k)}} \tilde{\mathbf{A}}^\top_{kuv} \sum_{j=1}^{K} \mathbf{R}_{vj} \mathbf{S}_{kji}$ in the above equation is written as:

$$\sum_{v \in \overleftarrow{\mathbf{N}}_u^{(k)}} \tilde{\mathbf{A}}^\top_{kuv} \sum_{j=1}^{K} \mathbf{R}_{vj} \mathbf{S}_{kji} = \sum_{v \in \overleftarrow{\mathbf{N}}_u^{(k)}} \tilde{\mathbf{A}}^\top_{kuv} (\mathbf{R}\mathbf{S}_k)_{vi} = (\tilde{\mathbf{A}}_k^\top \mathbf{R}\mathbf{S}_k)_{ui}$$

Then (9) is represented as follows:

$$\sum_{k=1}^{K} \left(\sum_{v \in \overleftarrow{\mathbf{N}}_u^{(k)}} \tilde{\mathbf{A}}^\top_{kuv} \sum_{j=1}^{K} \mathbf{R}_{vj} \mathbf{S}_{kji} \right) = \sum_{k=1}^{K} (\tilde{\mathbf{A}}_k^\top \mathbf{R}\mathbf{S}_k)_{ui} = \left(\sum_{k=1}^{K} \tilde{\mathbf{A}}_k^\top \mathbf{R}\mathbf{S}_k \right)_{ui}$$

Thus $\mathbf{R}_{ui}$ in (3) is written as follows:

$$\mathbf{R}_{ui} = (1-c) \left(\sum_{k=1}^{K} \tilde{\mathbf{A}}_k^\top \mathbf{R}\mathbf{S}_k \right)_{ui} + c\mathbf{1}(u = s, l_i = l_d)$$

For $1 \le u \le n$ and $1 \le i \le K$ where n is the number of nodes, the above equation is represented as (4). □

Appendix C: Lemma for spectral radius in convergence theorem

Lemma 6 *Suppose* $\tilde{\mathbf{B}}^\top = \sum_{k=1}^{K} \mathbf{S}_k^\top \otimes \tilde{\mathbf{A}}_k^\top$ *where* $\tilde{\mathbf{A}}_k$ *is k-th labeled semi-row-normalized matrix, and* $\mathbf{S}_k$ *is k-th label transition probability matrix. Then,* $\|\tilde{\mathbf{B}}^\top\|_1 \le 1$*, and the spectral radius of* $\tilde{\mathbf{B}}^\top$ *is bounded as follows:* $\rho(\tilde{\mathbf{B}}^\top) \le 1$.

Proof According to spectral radius theorem [39], $\rho(\tilde{\mathbf{B}}^\top) \le \|\tilde{\mathbf{B}}^\top\|_1$

where $\|\tilde{\mathbf{B}}^\top\|_1$ is the maximum absolute column sum of $\tilde{\mathbf{B}}^\top$. Since each entry of $\tilde{\mathbf{B}}^\top$ is non-negative, $\|\tilde{\mathbf{B}}^\top\|_1$ is equal to the maximum value of the column sums of the matrix. The column sums are represented as follows:

$$\begin{aligned}
(\mathbf{1}^\top \otimes \mathbf{1}^\top)\tilde{\mathbf{B}}^\top &= (\mathbf{1}^\top \otimes \mathbf{1}^\top) \left(\sum_{k=1}^{K} \mathbf{S}_k^\top \otimes \tilde{\mathbf{A}}_k^\top \right) \\
&= \sum_{k=1}^{K} (\mathbf{1}^\top \otimes \mathbf{1}^\top)(\mathbf{S}_k^\top \otimes \tilde{\mathbf{A}}_k^\top) = \sum_{k=1}^{K} \mathbf{1}^\top \mathbf{S}_k^\top \otimes \mathbf{1}^\top \tilde{\mathbf{A}}_k^\top
\end{aligned}$$

According to Definition 3, the sum of each row of $\mathbf{S}_k$ is 1, i.e., $\mathbf{S}_k\mathbf{1} = \mathbf{1} \Leftrightarrow \mathbf{1}^\top\mathbf{S}_k^\top = \mathbf{1}^\top$. Hence,

$$\sum_{k=1}^{K} \mathbf{1}^\top\mathbf{S}_k^\top \otimes \mathbf{1}^\top\tilde{\mathbf{A}}_k^\top = \sum_{k=1}^{K} \mathbf{1}^\top \otimes \mathbf{1}^\top\tilde{\mathbf{A}}_k^\top = \mathbf{1}^\top \otimes \sum_{k=1}^{K} \mathbf{1}^\top\tilde{\mathbf{A}}_k^\top$$
$$= \mathbf{1}^\top \otimes \mathbf{1}^\top \sum_{k=1}^{K} \tilde{\mathbf{A}}_k^\top$$

Note that $\tilde{\mathbf{A}}_k^\top = \mathbf{A}_k^\top\mathbf{D}^{-1}$ according to Definition 5. Suppose $\mathbf{A}' = \sum_{k=1}^{K}\mathbf{A}_k$ is the adjacency matrix of the graph G without edge labels. Then, $\mathbf{1}\sum_{k=1}^{K}\tilde{\mathbf{A}}_k^\top = \mathbf{1}\sum_{k=1}^{K}\mathbf{A}_k^\top\mathbf{D}^{-1} = (\mathbf{A}'\mathbf{1})^\top\mathbf{D}^{-1}$. The u-th row of $\mathbf{A}'\mathbf{1}$ indicates the out-degree of node u, denoted by deg_u. If node u is a deadend node, then $deg_u = 0$. Otherwise, $deg_u > 0$. Note that $\mathbf{D}$ is the out-degree diagonal matrix of G and $\mathbf{D}_{uu}^{-1} = 1/deg_u$ if node u is not a deadend. Otherwise, $\mathbf{D}_{uu}^{-1} = 0$. Thus, $(\mathbf{A}'\mathbf{1})^\top\mathbf{D}^{-1} = \mathbf{b}^\top$ where u-th entry of $\mathbf{b}$ is 1 if node u is non-deadend, or 0 otherwise. Hence, $(\mathbf{1}\otimes\mathbf{1})\tilde{\mathbf{B}}^\top = \mathbf{1}\otimes\mathbf{b}^\top$ which is the column sum vector of $\tilde{\mathbf{B}}^\top$, and the maximum value of the vector is less than or equal to 1. Therefore, $\|\tilde{\mathbf{B}}^\top\|_1 \leq 1$, implying $\rho(\tilde{\mathbf{B}}^\top) \leq \|\tilde{\mathbf{B}}^\top\|_1 \leq 1$. □

Appendix D: Lemma for Complexity Analysis

Lemma 7 *The time complexity of Algorithms 1 and 2 is $O(m^{1.5} + K^3)$ where m is the number of edges, and K is the number of edge labels.*

Proof In Algorithm 1, it takes $O(m^{1.5})$ time to enumerate all transitive triangles in the given graph G using a triangle enumeration algorithm [17] (line 1 in Algorithm 1). Also, estimating $\mathbf{S}_k$ requires $O(K^3)$ time (lines 2 $\sim$ 8 in Algorithm 1). Algorithm 2 takes $O(m)$ time for counting out-degrees of nodes (line 2 in Algorithm 2) and computing $\tilde{\mathbf{A}}_k = \mathbf{D}^{-1}\mathbf{A}_k$ for $1 \leq k \leq K$ (line 3 in Algorithm 2).

□

Lemma 8 *The time complexity of Algorithm 3 is $O(T(Km + K^3n))$ where $T = \log_{(1-c)}\frac{\epsilon}{2}$ indicates the number of iterations for convergence, ϵ is an error tolerance, m is the number of edges, n is the number of nodes, and K is the number of edge labels.*

Proof Let m_k denote the number of non-zeros in k-th semi-row normalized matrix $\tilde{\mathbf{A}}_k$ stored in a sparse matrix format such as compressed column storage (CCS). For each iteration, it takes $O(Km_k + K^2n)$ time to compute $\tilde{\mathbf{A}}_k^\top\mathbf{R}^{(t-1)}\mathbf{S}_k$ since the sparse matrix product $\tilde{\mathbf{A}}_k^\top\mathbf{R}^{(t-1)}$ requires $O(Km_k)$ time, and the dense matrix product $(\tilde{\mathbf{A}}_k^\top\mathbf{R}^{(t-1)})\mathbf{S}_k$ takes $O(K^2n)$ time. Thus, computing $\sum_{k=1}^{K}(\tilde{\mathbf{A}}_k^\top\mathbf{R}^{(t-1)}\mathbf{S}_k)$ takes $O(\sum_{k=1}^{K}(Km_k + K^2n)) = O(Km + K^3n)$ where $\sum_{k=1}^{K}m_k = m$ (line 6). Note that when $2(1-c)^t \leq \epsilon$, $\mathbf{R}^{(t)}$ converges since $\delta^{(t)} \leq 2(1-c)^t$ by Theorem 1. Hence, for $t \geq \log_{(1-c)}\frac{\epsilon}{2}$, the iteration is necessarily terminated. Thus, the number of iterations for convergence is estimated at $\log_{(1-c)}\frac{\epsilon}{2}$, and the total time complexity is $O((\log_{(1-c)}\frac{\epsilon}{2})(Km + K^3n))$. □

References

1. Avron, H., Horesh, L.: Community detection using time-dependent personalized pagerank. In: ICML (2015)
2. Bonchi, F., Gionis, A., Gullo, F., Ukkonen, A.: Distance oracles in edge-labeled graphs. In: EDBT (2014)
3. Bordes, A., Glorot, X., Weston, J., Bengio, Y.: A semantic matching energy function for learning with multi-relational data. Mach. Learn. **94**(2), 233–259 (2014)
4. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NIPS (2013)
5. Boyd, S., Vandenberghe, L.: Convex optimization. Cambridge University Press, Cambridge (2004)
6. Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., Zhang, W.: Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In: SIGKDD. ACM (2014)
7. Eswaran, D., Günnemann, S., Faloutsos, C., Makhija, D., Kumar, M.: Zoobp: Belief propagation for heterogeneous networks. In: Proceedings of the VLDB Endowment, vol. 10, pp. 625–636 (2017)
8. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: SIGKDD. ACM (2016)
9. Guu, K., Miller, J., Liang, P.: Traversing knowledge graphs in vector space. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 318–327 (2015)
10. He, J., Li, M., Zhang, H.-J., Tong, H., Zhang, C.: Manifold-ranking based image retrieval. In: Proceedings of the 12th Annual ACM International Conference on Multimedia. ACM (2004)
11. Jiang, Z., Liu, H., Fu, B., Wu, Z., Zhang, T.: Recommendation in heterogeneous information networks based on generalized random walk model and bayesian personalized ranking. ACM (2018)
12. Jung, J., Jung, W.J., Sael, L., Kang, U.: Personalized ranking in signed networks using signed random walk with restart. In: ICDM. IEEE (2016)
13. Jon, M.: Kleinberg. Authoritative sources in a hyperlinked environment. J. ACM (JACM) **46**(5), 604–632 (1999)
14. Koutra, D., Ke, T.-Y., Kang, U., Chau, D.H.P., Pao, H.K.K., Faloutsos, C.: Unifying guilt-by-association approaches: Theorems and fast algorithms. In: ECML, pp. 245–260. Springer (2011)
15. Kunegis, J., Lommatzsch, A., Bauckhage, C.: The slashdot zoo: mining a social network with negative edges. In: WWW. ACM (2009)
16. Lao, N., Mitchell, T., Cohen, W.W.: Random walk inference and learning in a large scale knowledge base. In: EMNLP. ACL (2011)
17. Latapy, M.: Main-memory triangle computations for very large (sparse (power-law)) graphs. Theor. Comput. Sci. **407**(1–3), 458–473 (2008)
18. Laub, A.J.: Matrix analysis for scientists and engineers, vol. 91 Siam (2005)
19. Lee, S., Park, S., Kahng, M., Lee, S.: Pathrank: a novel node ranking measure on a heterogeneous graph for recommender systems. In: CIKM. ACM (2012)
20. Leskovec, J., Huttenlocher, D.P., Kleinberg, J.M.: Governance in social media: A case study of the wikipedia promotion process. In: ICWSM (2010)
21. Leskovec, J., Mcauley, J.J.: Learning to discover social circles in ego networks. In: NIPS (2012)
22. Li, L., Yao, Y., Tang, J., Fan, W., Tong, H.: Quint: on query-specific optimal networks. In: SIGKDD. ACM (2016)
23. Lin, Y., Liu, Z., Luan, H., Sun, M., Rao, S., Liu, S.: Modeling relation paths for representation learning of knowledge bases. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 705–714 (2015)
24. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: AAAI, vol. 15, pp. 2181–2187 (2015)
25. Massa, P., Avesani, P.: Controversial users demand local trust metrics: An experimental study on epinions. com community. In: AAAI (2005)
26. Massa, P., Salvetti, M., Tomasoni, D.: Bowling alone and trust decline in social network sites. In: DASC. IEEE (2009)
27. Miller, G.A.: Wordnet: a lexical database for english. Commun. ACM **38**(11), 39–41 (1995)
28. Murphy, K.: Machine learning: a probabilistic approach. Massachusetts Institute of Technology, 1–21 (2012)
29. Page, L., Brin, S., Motwani, R., Winograd, T.: The Pagerank Citation Ranking: Bringing Order to the Web. Technical report, Stanford InfoLab (1999)
30. Park, H., Jung, J., Kang, U.: A comparative study of matrix factorization and random walk with restart in recommender systems. In: BigData. IEEE (2017)
31. Perozzi, B., Schueppert, M., Saalweachter, J., Thakur, M.: When recommendation goes wrong: Anomalous link discovery in recommendation networks. In: SIGKDD. ACM (2016)

32. Shahriari, M., Jalili, M.: Ranking nodes in signed social networks. Soc. Netw. Anal. Min. **4**(1), 172 (2014)
33. Socher, R., Chen, D., Manning, C.D., Ng, A.: Reasoning with neural tensor networks for knowledge base completion. In: NIPS (2013)
34. Sokolova, M., Lapalme, G.: A systematic analysis of performance measures for classification tasks. Inform. Process. Manag. **45**(4), 427–437 (2009)
35. Strang, G.: Linear Algebra and Its Applications. Thomson Brooks/Cole, Pacific Groove (2006)
36. Tang, J.M.Q., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. In: WWW. International World Wide Web Conferences Steering Committee (2015)
37. Tong, H., Faloutsos, C.: Center-piece subgraphs: problem definition and fast solutions. In: SIGKDD. ACM (2006)
38. Tong, H., Faloutso, C., Pan, J.-Y.: Fast random walk with restart and its applications. In: ICDM. IEEE (2006)
39. Trefethen, L.N., Bau, D. III.: Numerical linear algebra, vol. 50, Siam (1997)
40. Wang, X., van Eeden, C., Zidek, J.V.: Asymptotic properties of maximum weighted likelihood estimators. J. Stat. Plan. Infer. **119**(1), 37–54 (2004)
41. Yang, B., Yih, W.-T., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. In: 3rd International Conference on Learning Representations, ICLR 2015, San diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015)

Affiliations

Jinhong Jung[1] · Woojeong Jin[2] · Ha-myung Park[3] · U Kang[1]

Jinhong Jung
jinhongjung@snu.ac.kr

Woojeong Jin
woojeong.jin@usc.edu

Ha-myung Park
hmpark@kookmin.ac.kr

[1] Seoul National University, Seoul, South Korea

[2] University of Southern California, Los Angeles, CA, USA

[3] Kookmin University, Seoul, South Korea