

Resolving Identity Uncertainty with Learned Random Walks

Ted Sandler, Lyle H. Ungar
Computer and Information Science
University of Pennsylvania
{tsandler, ungar}@cis.upenn.edu

Koby Crammer*
Department of Electrical Engineering
Technion - Israel Institute of Technology
koby@ee.technion.ac.il

Abstract

A pervasive problem in large relational databases is identity uncertainty which occurs when multiple entries in a database refer to the same underlying entity in the world. Relational databases exhibit rich graphical structure and are naturally modeled as graphs whose nodes represent entities and whose typed-edges represent relations between them. We propose using random walk models for resolving identity uncertainty since they have proven effective for finding points which are proximately located in a network. Because not all types of relations are equally helpful in alleviating identity uncertainty, we develop a supervised approach to learning the usefulness of different database relations from a training set of database entries whose true identities are known. When tested on the task of resolving uncertainty of ambiguously named authors in bibliographical data, the learned random walk models yield performance superior to support vector machines, and to a related spectral clustering method.

1. Introduction

Many authors have proposed using random walk models for inducing similarities over relational data represented as a network or graph [26, 21, 17]. Such models capture an intuitive notion of graph similarity such that the similarity of nodes grows as a function of the number of short paths between them. Consequently, two nodes are similar if they lie in nearby, well-connected regions of the network.

Here we propose using random walk models for resolving identity uncertainty in bibliographic data. Identity uncertainty is a problem which occurs when names in a database are not in one-to-one correspondence with the entities they refer to [27]. Random walk models are well suited to resolving identity uncertainty since coreferent database entries—i.e. entries which refer to the same entity

in the world—will possess similar or identical relationships to other coreferent entries. Hence there will be many chains of commonality linking them, or linking the items to which they are linked. For example, duplicate entries of an article in a bibliographic database might possess fields indicating that their titles are similar, that they share some coauthors in common, and that they were published in the same journal. Consequently, when viewed as nodes in a graph, these duplicates will have many edges which connect them.

While other authors have proposed using random walks for resolving identity uncertainty [21, 26], we use machine learning to learn the importance of different database relations in terms of how helpful they are at resolving identity uncertainty. Such learning is critical in feature rich databases which contain many different relationships, each indicating potential similarities between database entries.

Identity uncertainty is a problem faced when trying to extract even the simplest information from many real-world, bibliographic databases. For example, it is common to measure the impact of an author by the number of articles that he or she has published. But as discussed by Han et al., highly prolific authors often turn out to be amalgams of several individuals, all with the same or similar names [15]. A related problem occurs in biomedicine where, due to the uncoordinated efforts of different research communities, distinct genes are given the same name or abbreviation. Chen et al. found that in a sample of nearly 150,000 official gene symbols collected from 21 species, identical symbols existed in one of the other 20 species roughly 14% of the time [7]. The converse problem, in which a distinct gene possesses multiple names is also widespread; it is common for genes to have both long-form and abbreviated names, as well as to be called by different names by different research communities. Bioinformatic analysis is slowed by difficulties in determining which genes are meant by which names [32].

The task of resolving name ambiguity has itself been called by many different names, including identity uncertainty, reference reconciliation, record linkage, the merge-purge problem, and entity resolution [11, 13, 16, 5]. Early

*A Horev Fellow - supported by the Taub Foundation

work on entity resolution treated the problem as a binary classification task such that matching decisions were considered in isolation or in sequence [13, 6, 3]. More recent work has framed the problem as a collective classification problem where a global cost is defined over the entire set of match decisions [4, 8, 29, 10]. In richly relational data, a collective approach is appropriate since different matching decisions need to be considered simultaneously in order to preserve the relational semantics of the data. Unfortunately, minimizing the global cost of these collective models is NP-complete, so optimization must be done in a series of local refinement steps.

The method we propose shares features of both the collective and the independent approach. Similar to collective approaches, we model the relational structure of the data using graphs whose nodes represent entity references and whose edges represent typed relations between them. However, rather than performing classification directly on this graph, as the collective approaches do, we instead use the graph to induce a random-walk based similarity, or distance, on the entity references. We then use this similarity to make independent classification decisions. The similarity that we define between references is the probability that a random walk started from one reference ends at another.

As mentioned above, we propose a method for learning the weight of each relation type. Specifically, each relation type represents a type of transition in an absorbing Markov chain and, given training data, we show how to learn the importance of these transition types so that random walks will connect references which refer to the same entity. Due to the parametric form of our model, the similarity we learn is symmetric and induces a distance on the graph known as the resistance distance [33]. We note that while some authors have used random walk models or related spectral clustering methods for resolving identity uncertainty [15, 21, 26], our work addresses the problem of *learning* the importance of different transition types, a problem which has received less attention in the literature. When tested on the task of resolving uncertainty among ambiguously named authors in bibliographical data, our method of learned random walks gives superior performance to support vector machines and to a related spectral clustering method [22].

2 Entity Resolution on Graphs

In what follows, we assume we are given an undirected graph $G = (V, E)$ whose vertices are entity references and whose edges represent similarities between those references in terms of how likely they are to corefer. The edges of the graph have weights which reflect the amount of similarity. Greater weight means greater similarity while a weight of zero means no similarity. The graph's edge weights are stored in a matrix W whose entry W_{uv} denotes the weight

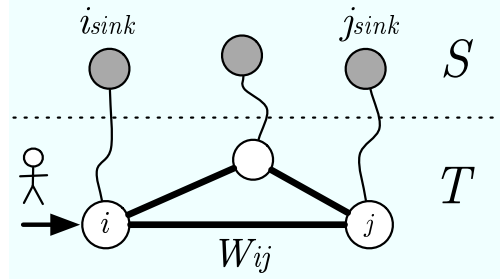


Figure 1. The graph is extended such that a sink node (gray) is attached to each transient node (white). S denotes the set of sink nodes and T denotes the set of transient nodes.

of the edge between references u and v .

In the problem of author-name disambiguation, each vertex represents a reference to an author on a particular paper. So if u refers to the i 'th author on the p 'th paper of the citation database, and v refers to the j 'th author of the q 'th paper of the database, then W_{uv} reflects the degree to which we believe that u and v refer to the same person in the world. Factors which would determine the value of W_{uv} might include the string edit-distance between the i 'th name on the p 'th paper and the j 'th name on the q 'th paper (i.e., the string edit distance between the names associated with u and v), the number of words shared between the titles of papers p and q , whether p cited q (or vice versa), and whether or not p and q were published in the same journal.

2.1 Random Walk Similarity

Given an undirected, weighted graph with weight matrix W , we can construct a transitive similarity score between author references by considering random walks of unbounded length on the graph. We define this similarity score in terms of an absorbing Markov chain [12], or equivalently, a lazy random walk on the graph. To do this, we construct a probability transition matrix P by normalizing the rows of W so that they sum to one: $P_{ij} = W_{ij} / \sum_k W_{ik}$. The entry $P_{ij} = P(j|i)$ denotes the probability that a random walk at vertex i will be at vertex j in the next time-step. The transitive similarity score that we wish to induce from P is the probability that a random walk started from vertex i ends at vertex j after a sequence of steps. However, to make this definition precise, we need to impose some stopping conditions so that walks actually terminate.

We do this by attaching a terminal node, or “sink,” to each vertex and requiring that a walk stops once it hits a sink. That is, for each vertex i , we create a sink node i^{sink} and link i to i^{sink} with an edge of weight one. A walk at a

non-sink vertex i is allowed to transition to another non-sink node j with probability $P(j|i) = W_{ij}/(1 + \sum_k W_{ik})$, or it can exit to its sink node i^{sink} with probability $P(i^{sink}|i) = 1/(1 + \sum_j W_{ij})$ and terminate. Nodes which are not sinks are called “transient” nodes. Figure 1 depicts the construction.

With this construction in hand, we now define the transitive/multistep similarity between two nodes to be the probability that a random walk started from one node ends at the other node’s sink. That is, for two nodes i and j , we define the transitive similarity between i and j to be

$$A_{ij} = P(j^{sink}|i). \quad (1)$$

We call “ A ” the absorbing probability matrix and its entries A_{ij} give the probability of landing in sink node j^{sink} when from transient node i .

2.2 Computing the Random Walk Similarity

We now show how to compute A analytically. For notational convenience, it is easiest to assume that the vertex set V , edge set E , and weight matrix W , have all been extended to contain the added sink nodes and the transitions to them. Partitioning V into the set of sinks, S , and transient nodes, T , and ordering the sink nodes ahead of the transients, W can be written in block form as

$$W = \begin{pmatrix} W_{SS} & W_{ST} \\ W_{TS} & W_{TT} \end{pmatrix}. \quad (2)$$

Because the nodes of S are sinks, they have no outgoing transitions, or equivalently, they only transition back to themselves. Thus $W_{SS} = I$, the identity matrix, and $W_{ST} = 0$, the zero matrix. Additionally, if we assume an ordering of the sink nodes such that the i ’th vertex of S is the sink for the i ’th vertex of T , the matrix W_{TS} can also be written as the identity matrix, I , because we assumed that the edges connecting transients to sink nodes had weight one. Finally, we can analogously decompose P into P_{SS} , P_{ST} , P_{TS} and P_{TT} by normalizing the rows of W as it is written in equation (2). This allows us to compute the absorbing probability matrix, A , as

$$A = \left[\sum_{k=0}^{\infty} (P_{TT})^k \right] P_{TS} = (I - P_{TT})^{-1} P_{TS}. \quad (3)$$

Here we have used the fact that $(P^k)_{ij}$ is the probability that a random walk starting at node i lands in node j at k time-steps in the future.

Intuitively, equation (3) says that A_{ij} is the probability that a random walk started at transient node i bounces around the transient nodes for some number of steps, lands

at transient node j and then exits to sink j^{sink} . The matrix inverse in equation (3) arises from the sum of a convergent geometric series and is well defined because the rows of P_{TT} sum to less than one, making $(I - P_{TT})$ diagonally dominant. Since a random walk on the graph must eventually terminate at some sink node¹, the matrix A is itself a probability transition matrix.

Surprisingly, A is also symmetric. If we define D to be the diagonal degree matrix where $D_{ii} = \sum_j W_{ij}$ and rewrite P as $D^{-1}W$, it is easy to show that

$$A = (D_T - W_{TT})^{-1} W_{TS}, \quad (4)$$

where we have written D as $D = \begin{pmatrix} D_S & 0 \\ 0 & D_T \end{pmatrix}$. The matrix $(D_T - W_{TT})$ is symmetric and therefore so is its inverse. W_{TS} is equal to the identity matrix which yields the desired result.

As discussed in [33], the matrix A is an inner product on the vertices of the graph and therefore has an associated distance called the resistance distance:

$$\text{dist}(i, j) = \sqrt{A_{ii} + A_{jj} - 2A_{ij}}.$$

Consequently, either the similarities given by A , or the associated distances, can be used to resolve identity uncertainty.

2.3 Random Walks with Multiple Relations

In practice, we are not simply given a graph corresponding to an entity resolution task but instead must construct one from the relational database. A typical database contains many types of relations, only some of which provide relevant information for solving the entity resolution problem. We make the assumption that these relations are represented as a set of symmetric similarity matrices $\mathcal{W} = \{W^{(1)}, \dots, W^{(R)}\}$ which provide different similarity measures between the entity references of the database. To take our running example of ambiguous author references in a bibliographic database, one similarity matrix, $W^{(r)} \in \mathcal{W}$, will contain the string edit-distances between the names associated with the author references, while another similarity matrix, $W^{(s)} \in \mathcal{W}$, will be the binary-valued matrix whose entries $W_{ij}^{(s)}$ are zero or one depending on whether the article associated with author reference i cited the article associated with author reference j (or vice versa).

From the collection of similarity matrices, \mathcal{W} , we construct the weight matrix W_{TT} so that its entries, (i, j) , are a pointwise log-linear combination of the (i, j) ’th entries of the matrices in \mathcal{W} . That is,

$$W_{TT}(i, j) = \exp \left(\sum_{r=1}^R \theta_r W_{ij}^{(r)} \right). \quad (5)$$

¹The probability of an infinite step random walk is zero provided that $W_{TS}(i, i) > 0$ for all i .

The parameter $\theta \in \mathbb{R}^R$ is a vector of mixing coefficients which determines how much emphasis is placed on each similarity matrix, $W^{(r)}$. This parametric form for W_{TT} can be viewed as a set of experts who vote independently on which type of transition to take and has been used extensively in the spectral clustering literature [1, 22, 34]. Additionally, it later allows us to perform unconstrained optimization in order to learn the coefficient vector, θ , since it ensures that the entries of W_{TT} are never negative. We similarly define

$$W_{TS} = \exp(\theta^{sink}) \cdot I \quad (6)$$

where $\exp(\theta^{sink})$ can be interpreted as a bias weight which determines how likely a random walk is to exit to a sink-node at each step. The mixing coefficients θ and θ^{sink} can either be set by hand using domain expertise, or as we show next, they can be optimized through supervised learning.

3 The Learning Framework

For databases containing a small number of entity relations, reasonable values for the mixing coefficients can be chosen by hand. However, when there are many attributes, manually choosing optimal values for θ and θ^{sink} becomes arduous. Thus, we give a supervised learning procedure for choosing these values when the true identities are known for a small set of entity references in the database.

We assume that this knowledge is given to the learning procedure as a training set $\langle V_L, K \rangle$, where $V_L \subseteq V$ is the subset of nodes in the graph whose true identities are known and $K : V \times V \rightarrow \{-1, 0, +1\}$ is the labeling function which labels pairs of nodes as being coreferent or not:

$$K(i, j) = \begin{cases} +1 & \text{if } i, j \in V_L \text{ refer to the same entity} \\ -1 & \text{if } i, j \in V_L \text{ do not refer to the same entity} \\ 0 & \text{if } i \text{ or } j \text{ are not in } V_L \text{ or if } i = j. \end{cases} \quad (7)$$

We define $K(i, i) = 0$ because we know that a node is coreferent with itself and do not need to learn this information in training.

We denote the set of pairs of nodes which are known to be coreferent as $K^+ = \{(i, j) \mid K(i, j) = +1\}$ and similarly, the set of nodes known not to be coreferent as $K^- = \{(i, j) \mid K(i, j) = -1\}$. Additionally, we define $K_i^+ = \{j \mid (i, j) \in K^+\}$ and $K_i^- = \{j \mid (i, j) \in K^-\}$.

Given the set of weight matrices \mathcal{W} and a training set $\langle V_L, K \rangle$, the learning task is to find a vector θ which increases the probabilities A_{ij} for all $(i, j) \in K^+$ and decreases the probabilities A_{ij} for $(i, j) \in K^-$. To do this, we take the straight-forward approach of maximizing the sum of the log-probabilities $\sum_{(i,j) \in K^+} \log A_{ij}$. However, because walks can exit at unlabeled nodes, maximizing the

probabilities for pairs in K^+ is not guaranteed to minimize the probabilities of pairs in K^- . Consequently, it is necessary to add an explicit penalty on the entries A_{ij} for $(i, j) \in K^-$.

We give two related objectives for doing this. The first is analogous to way we handled pairs in K^+ . For all $(i, j) \in K^-$, we try to maximize the quantity, $\log(1 - A_{ij})$, which is large when A_{ij} is small. Incorporating this into our objective gives us the following function to maximize:

$$J_1(V_L, K, \theta) = \frac{\alpha}{|K^+|} \sum_{(i,j) \in K^+} \log(A_{ij}) + \frac{1 - \alpha}{|K^-|} \sum_{(i,j) \in K^-} \log(1 - A_{ij}) - \beta \|\theta\|^2. \quad (8)$$

Here we have included the ridge penalty $\|\theta\|^2 = \sum_r \theta_r^2$ to prevent the entries of θ from getting too big—a common problem in training log-linear models. The hyper-parameter α controls how much the objective focuses on K^+ versus K^- , and the hyper-parameter β controls the contribution of the ridge penalty to the overall objective.

In our second version of the objective function, rather than minimize the individual entries A_{ij} for $(i, j) \in K^-$, we minimize the overall probability of ending up at a non-coreferent node. That is, for all nodes i we minimize $P(K_i^- | i) = \sum_{j \in K_i^-} A_{ij}$. This is the probability of starting a walk at node i and ending up any node j which is known not to be coreferent with i . This objective therefore controls for the overall likelihood of a bad event rather than controlling for individual bad events. Under this alternate objective, we try to maximize the function:

$$J_2(V_L, K, \theta) = \frac{\alpha}{|K^+|} \sum_{(i,j) \in K^+} \log(A_{ij}) + \frac{1 - \alpha}{|V_L|} \sum_{i \in V_L} \log(1 - P(K_i^- | i)) - \beta \|\theta\|^2. \quad (9)$$

As in equation (8), maximizing $\log(1 - P(K_i^- | i))$ makes $P(K_i^- | i)$ small. The second summation is normalized by $|V_L|$ because it is a sum over $|V_L|$ terms.

In our experiments, we maximize J_1 or J_2 using gradient ascent. The gradient is relatively straight-forward to calculate using the chain rule. The only tricky part is the partial derivative, $\partial J / \partial A$, which is found to be

$$\frac{\partial A}{\partial \theta_r} = N \left(\frac{\partial D_T}{\partial \theta_r} - \frac{\partial W_{TT}}{\partial \theta_r} \right) A - N \frac{\partial W_{TS}}{\partial \theta_r} \quad (10)$$

where $N = (D_T - W_{TT})^{-1}$ is the inverse of the graph Laplacian, but where the rows of W_{TT} sum to less than the corresponding rows of D_T , due to the fact that each transient node has an additional edge connecting it to its sink.

When the input similarity matrices $W^{(r)}$ are dense, equation (10) scales as $O(R|T|^3)$ since the matrix multiplications are $O(|T|^3)$ and the derivative needs to be computed with respect to the $R = |\mathcal{W}|$ matrices. In practice, most of the similarity matrices are extremely sparse so the cost of equation (10) is closer to $O(R|T|^2)$. However, inverting the matrix $(D_T - W_{TT})$ is $O(|T|^3)$ and thus approximations must be used for graphs containing more than a few thousand nodes. The datasets in our experiments were small enough to compute the matrix inverse directly.

4 Evaluation

Identity uncertainty in bibliographic databases is a well-documented problem, especially for author names [20]. Because each author of an article is represented only by a text string spelling his or her name, confusion arises when two or more authors publish under the same name or when a single author publishes under slightly different names. For example, there are over 25 authors in the DBLP publishing under the name “Wei Li,” while the publications of the single author “Fernando C. N. Pereira” are attributed to three separate individuals: “Fernando C. N. Pereira,” “Fernando C. Pereira,” and “Fernando Pereira.” Additionally, there are at least four other individuals named “Fernando Pereira” in the DBLP. Such ambiguity makes it difficult to search for articles by a specific author, properly assess an author’s impact, or discover research communities [14].

We formulate the problem of resolving author identity uncertainty as learning the weight matrix W so that citations containing mentions of the same author will have high random-walk similarity. Each citation is represented by a transient node in the graph and is linked to its corresponding sink via the edges in W_{TS} (see section 2.1). The adjacency matrices in \mathcal{W} correspond to particular relations between citations such as the number of coauthor names that two citations have in common, the TF-IDF similarity of document titles, and the TF-IDF similarity between venue names. We also extract adjacency matrices corresponding to individual words in document titles and to similarities between document titles and journal names which have been projected into a low-dimensional space.

We learn and evaluate our random-walks approach on eight of the citation data sets used by Han et al. (see table 1) [15]. Each dataset contains citations for authors possessing the same first initial and last name. To create further ambiguity, Han et al. replaced first and middle name information with only the first initial of the first name. The size of the data sets in terms of number of citations, number of actual authors, and number of similarity matrices in \mathcal{W} is presented in table 1.

For all datasets, we extracted adjacency matrices representing TF-IDF similarities between coauthor names, be-

DATA SET	CITATIONS	AUTHORS	$ \mathcal{W} $
A. KUMAR	244	14	133
M. JONES	260	13	161
K. TANAKA	280	10	173
D. JOHNSON	368	15	170
J. SMITH	927	31	6
Y. CHEN	1264	71	6
J. LEE	1417	100	6
S. LEE	1458	86	6

Table 1. Citation data set statistics.

tween document titles, and between venue names. For the four smallest datasets, we also extracted adjacency matrices for each word that occurred in at least five document titles or venue names. For example, in the “A. Kumar” dataset, the word “network” occurs in six document titles. Thus the adjacency matrix for the word “network” contains edges connecting all six of these citations. Words in all three of these citation fields were tokenized on non-alphabetical characters, lower-cased, and stripped of accents. Stop-words were removed.

For the four large datasets, creating edges for each word in the vocabulary created too many adjacency matrices, and as shown in section 3, the gradient scales with the number of input similarity matrices, R . Therefore, to prevent this feature blow-up, we reduced the dimensionality of the vocabulary to 10 dimensions by applying non-negative matrix factorization [19] to a Word \times Venue matrix which we extracted from the DBLP database. In this matrix, each of the 2,500 venues is treated as a long document consisting of the words appearing in the titles of articles published in it. Non-negative matrix factorization was applied to this matrix to obtain a 10-dimensional vector representation of each word. Document titles and venue names were then obtained by the summing the 10-dimension vectors for the words they contained. By projecting titles and venue names into this space, we were able to replace the many vocabulary adjacency matrices with only two adjacency matrices: one consisting of the dot-products between the 10-dimensional title vectors and one consisting of the dot-products between the 10-dimensional venue vectors.

4.1 Experiments

We trained and evaluated random walk models on the eight datasets described above. For each dataset, we learned and computed the similarity matrix A and generated precision-recall curves by plotting precision and recall at each level of similarity. That is, at each threshold of similarity t , if two citations i and j have similarity $A_{ij} > t$,

we predict that i and j contain mentions of the same author. Precision is computed as the number of such predictions that are correct divided by the total number of predictions. Recall is computed as the number of correct predictions divided by the number of pairs of citations wherein both citations refer to the same author.

4.1.1 Learned versus Unlearned Random Walks

We first tested how much benefit is obtained from learning the weights of the different link types. To do this, we split the citations of the small datasets into training and testing sets and compared the performance of models learned on the training sets to “unlearned” models whose feature weights were all set equal to the same constant “1.”

Parameters for the random walk models were optimized via conjugate gradient with line search. The hyperparameters α and β in equations (8) and (9) were chosen using a grid search over $\alpha, \beta \in [0, 1]$ and picking the combination which yielded the highest average precision on the training set. Results for the four smaller datasets are averages over five-fold cross-validation. Training plus grid search took roughly half an hour per fold on a machine with an Intel Xeon 3GHz processor. On the four larger data sets, five-fold cross-validation was impractical so we evaluated on a single train-test split wherein 500 citations were sampled for training and the rest were used for testing. Training on the larger data sets took between 9 and 23 hours on the same machine. Typical values chosen for α were between 0.1 and 0.5 while β ranged between 0.01 and 0.1.

Precision and recall curves are given in figure 2 comparing the learned and unlearned models. The results shown were obtained using objective function J_2 which consistently out-performed J_1 . In every case learning the relative importance of different link types greatly improves both precision and recall. In particular since link types were created for many of the words found in the document titles and venue names, learning is critical for detecting the important words which distinguish authors.

4.1.2 Necessity of Penalizing Probabilities for K^-

Next, to assess the benefit obtained by penalizing the set of dissimilar citations, K^- , we compared our full model to a model which lacked this penalty. The results shown in table 2 demonstrate that including the penalty on K^- improves the performance of our model. The results here are reported in terms of average precision, i.e. the average of precision scores at each level of recall.

4.1.3 Comparisons against Support Vector Machines

To compare against an external baseline, we trained support vector machines (SVMs) on the entity resolution task,

DATA SET	K^- PENALTY	AVG. PREC.	STD. ERR.
A. KUMAR	✓	0.86	0.09
	⊙	0.80	0.10
M. JONES	✓	0.74	0.05
	⊙	0.73	0.03
K. TANAKA	✓	0.91	0.03
	⊙	0.68	0.04

Table 2. Average precision with “✓” and without “⊙” the penalty on pairs in K^- .

treating it as a binary classification problem in which the object is to predict whether two citations refer to the same author [6]. Since SVMs have smaller time complexity than our method with respect to the number of features used, all features were used in training the support vector machines. This can only help the support vector methods since the number of features (similarity matrices) used is too small relative to the number of training pairs to create overfitting (the number of features is given in table 1).

We used two methods to generate precision and recall curves for the SVM output. In the first, each prediction was taken independently and precision and recall curves were generated as above using the SVM prediction score as the similarity. In the second method, the transitive closure of predictions was computed at each threshold [24]. That is, if citations i and j were predicted to corefer and citations j and k were also predicted to corefer, then citations i and k would be predicted to corefer. Such a strategy adds transitivity to the SVM predictions. Note that this strategy is not needed for random walks since they are inherently transitive. In figure 3, the two strategies are called “indep” and “closure.” We used SVMLight [18] and trained it with a polynomial kernel of degree two. Linear and higher order polynomials were also tried, as were Gaussian kernels, but none consistently out-performed the chosen kernel. A range of values were tried for the slack-variable penalty, C , all of which yielded similar performance. Therefore we conservatively picked C to be the inverse of the 90-th percentile of the inner-product values, $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ for all i and j .

4.1.4 Comparison with Learned Spectral Clustering

We also compare our method against the method for learning random walks proposed by Meila and Shi [22], which learns a transition matrix P from random walks consisting of only one step—i.e., walks are required to stop after one step. In their method, the smallest eigenvectors of the learned graph are used to cluster the graph’s nodes. Instead of clustering, we use these eigenvectors to form a low-

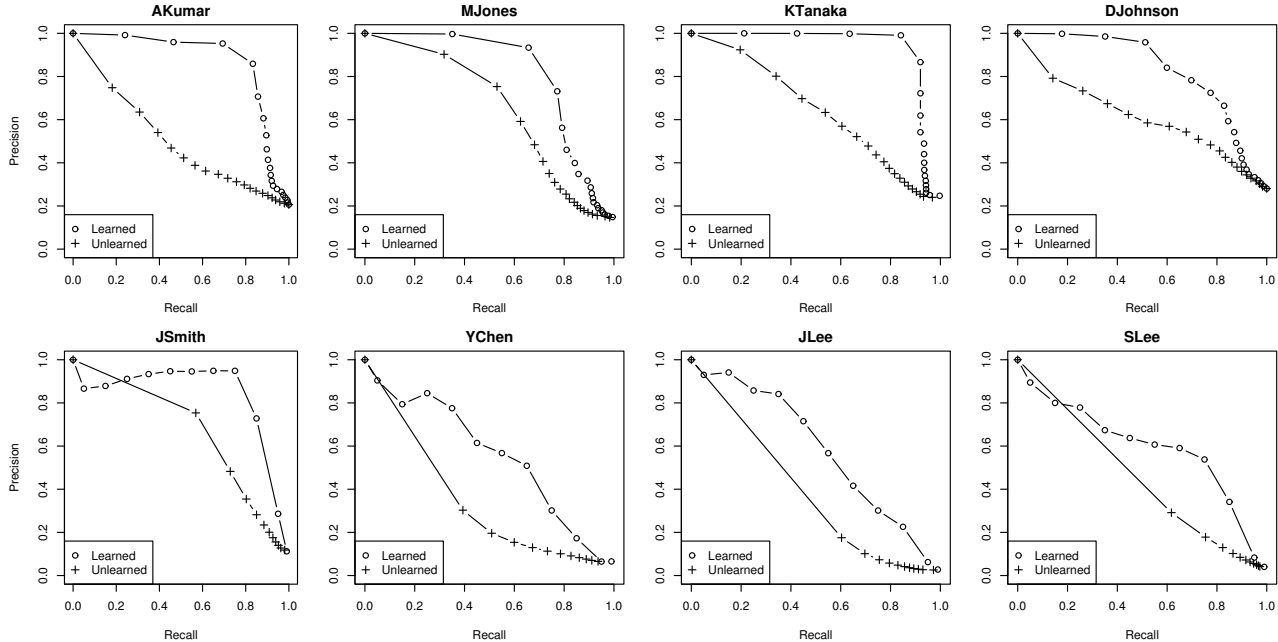


Figure 2. Precision and recall curves for learned and “unlearned” random walk models.

dimensional embedding of the nodes and compute a similarity from the embedding coordinates. Let $W^{(ms)}$ be the weight matrix learned from their method and let $L^{(ms)}$ be its associated Laplacian matrix. From $L^{(ms)}$, we compute the ten smallest eigenvectors of the generalized eigenvector/value problem, $L^{(ms)}\mathbf{v} = \lambda\mathbf{v}$ (see [28] and [33] for details), and use them to find the embedding coordinate vectors, $\mathbf{y}_i \in \mathbb{R}^{10}$ for each node i in the graph. The similarity between nodes i and j is taken to be the inner product, $\langle \mathbf{y}_i, \mathbf{y}_j \rangle$. Results for the spectral learning method are given in figure 3.

The results of figure 3 show that on the smaller datasets, random walks perform comparably to, or better than, the SVM and spectral methods. The spectral method, on the whole, performs worse than either the random walk models or the SVMs. On the larger datasets where dimensionality reduction is used to limit the number of input similarity matrices, the learned random walk models still perform as well as, or better than, the SVMs trained with all features. Finally we note that taking the transitive closure seems to improve the performance of SVMs, but it is not clear how great the benefit is due to the resulting jaggedness of the precision-recall curves.

5 Related Work

There has been extensive work on entity resolution involving both supervised and unsupervised learning of prob-

abilistic relational models [4, 10, 27, 29]. These approaches provide powerful frameworks for resolving identity uncertainty. However, they differ from the work presented here in that they are not based on random walks.

Malin et al. do use random walks to disambiguate actor names within the Internet Movie Database (IMDB) but they do not learn the link weights [21]. Minkov et al. use random walks to disambiguate names in email messages. Instead of learning link weights, they use absorbing probabilities as a feature in a linear classifier [26]. More recent work by Minkov and Cohen uses error back-propagation to learn the connection strengths of different links in the graph [25]. Additionally, they minimize squared error rather than log-loss. It remains an area of future research to determine which approach is more appropriate for entity resolution. Toutanova et al. learn transition probabilities in a random walk model for prepositional phrase attachment [30]. Their loss function is tailored to the particular natural language task and is quite different from the objective function we propose here.

Zhu et al. apply absorbing random walks to the problem of semi-supervised learning on a partially labeled graph [34]. They work in a classification setting where the number of classes is known at training time. This is unlike the entity resolution task where the true number of entities (classes) is unknown. Zhu et al. do learn link weights, but their objective minimizes the label entropy of nodes in the test set rather than minimizing a loss computed on nodes

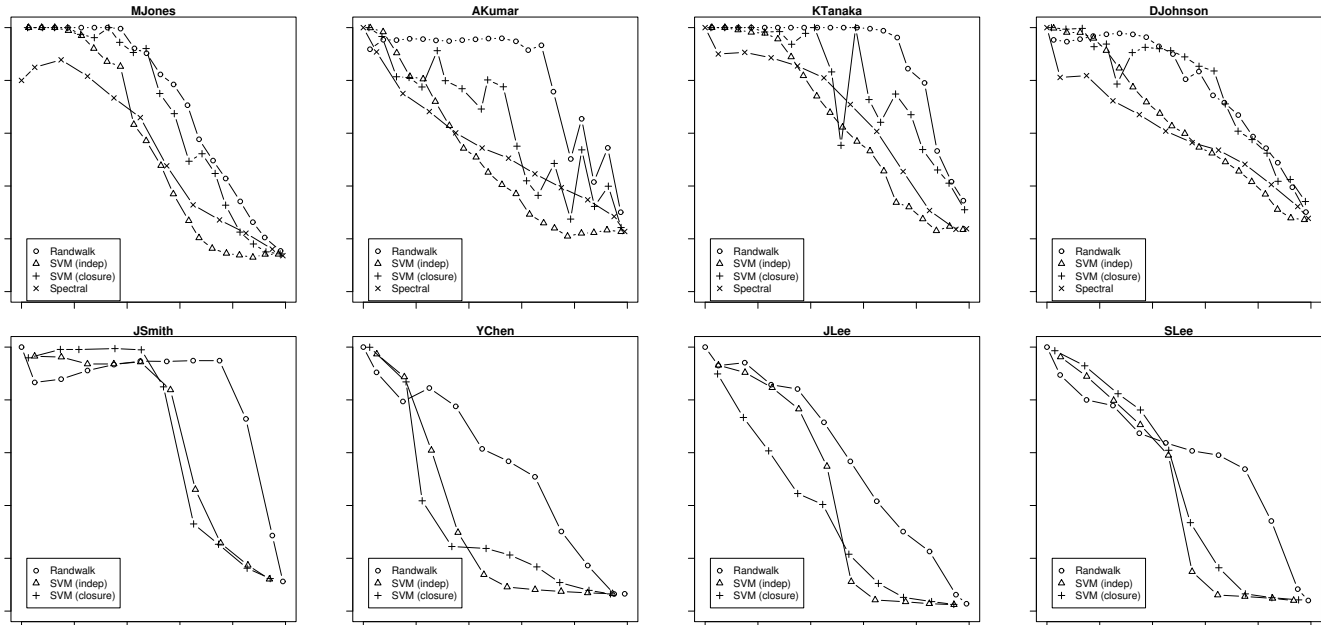


Figure 3. Precision and Recall Curves. The ‘x’ and ‘y’ axes range from zero to one. Row 1: Random walks, SVMs, and learned spectral clustering run on the four small datasets. Row 2: Random walks and SVMs run on the four large datasets.

of the training set. The work of Tsuda et al. [31] is similar to that of Zhu et al. in that they perform semi-supervised classification on graphs. The authors propose an objective function for learning link weights which minimizes prediction errors on the training set and maximizes the smoothness of the prediction function over the entire graph. Again, their objective function is quite different from ours.

Finally, several authors have incorporated learning into spectral clustering with a loss on the eigenvectors and eigenvalues of the normalized graph Laplacian [2, 9, 23]. We differ in taking a probabilistic approach which avoids some of the difficulties involved in differentiating with respect to eigenvectors and eigenvalues.

6 Conclusion

We have proposed a method based on absorbing random walks for entity resolution. Unlike other random-walk based approaches to this problem, we give a supervised learning procedure for learning the transition probabilities from training data. Large numbers of features can be a bottleneck for our model but we show how dimensionality reduction can be used to circumvent this problem. The learned random walk models out-perform support vector machines and a related spectral clustering method on the task of resolving author ambiguities in bibliographic data. Finally, the learned random walk models achieve greater

classification accuracy than the unlearned models and induce a distance on the nodes of the graph.

References

- [1] F. R. Bach and M. I. Jordan. Learning Spectral Clustering. In *Advances in Neural Information Processing Systems (NIPS)*, 2004.
- [2] F. R. Bach and M. I. Jordan. Learning spectral clustering, with application to speech separation. *Journal of Machine Learning Research*, 7:1963–2001, 2006.
- [3] I. Bhattacharya and L. Getoor. Iterative record linkage for cleaning and integration. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD)*, 2004.
- [4] I. Bhattacharya and L. Getoor. A latent dirichlet model for unsupervised entity resolution. In *SIAM Conference on Data Mining (SDM)*, April 2006.
- [5] I. Bhattacharya and L. Getoor. Collective Entity Resolution In Relational Data. *ACM Transactions on Knowledge Discovery from Data*, 1(1):1–36, March 2007.
- [6] M. Bilenko, R. Mooney, W. W. Cohen, P. Ravikumar, and S. Fienberg. Adaptive Name Matching in Information Integration. *IEEE Intelligent Systems*, 18(5):16–23, 2003.
- [7] L. Chen, H. Liu, and C. Friedman. Gene name ambiguity of eukaryotic nomenclatures. *Bioinformatics*, 21(2):248–256, 2005.
- [8] W. Cohen, H. Kautz, and D. McAllester. Hardening Soft Information Sources. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, 2000.

- [9] T. Cour, N. Gogin, and J. Shi. Learning Spectral Graph Segmentation. In *Artificial Intelligence and Statistics (AISTATS)*, 2005.
- [10] A. Culotta and A. McCallum. Joint deduplication of multiple record types in relational data. In *International Conference on Information and Knowledge Management (CIKM)*, 2005.
- [11] X. Dong, A. Halevy, and J. Madhavan. Reference reconciliation in complex information spaces. In *International Conference on Management of Data (SIGMOD)*, 2005.
- [12] P. G. Doyle and J. L. Snell. Random Walks and Electric Networks. *The Mathematical Association of America*, 22, 1984.
- [13] I. P. Fellegi and A. B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64(328):1183–1210, 1969.
- [14] C. L. Giles, K. D. Bollacker, and S. Lawrence. CiteSeer: an automatic citation indexing system. In *International Conference on Digital Libraries (ICDL)*, 1998.
- [15] H. Han, H. Zha, and C. L. Giles. Name disambiguation in author citations using a k-way spectral clustering method. In *Joint Conference on Digital Libraries (JCSDL)*, 2005.
- [16] M. Hernández and S. Stolfo. The merge/purge problem for large databases. In *International Conference on Management of Data (SIGMOD)*, 1995.
- [17] J. Huang, T. Zhu, R. Greiner, D. Schuurmans, and D. Zhou. Information Marginalization on Subgraphs. In *Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 2006.
- [18] T. Joachims. *Advances in Kernel Methods - Support Vector Learning*, chapter Making large-Scale SVM Learning Practical. B. Schölkopf, C. Burges and A. Smola. MIT-Press, 1999.
- [19] D. D. Lee and H. S. Seung. Algorithms for Non-negative Matrix Factorization. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.
- [20] M. Ley. The DBLP Computer Science Bibliography: Evolution, Research Issues, Perspectives. In *International Symposium on String Processing and Information Retrieval (SPIRE)*, 2002.
- [21] B. Malin, E. Airoldi, and K. Carley. A network analysis model for disambiguation of names in list. *Computational and Mathematical Organization Theory*, 11:119–139, 2005.
- [22] M. Meilă and J. Shi. Learning Segmentation by Random Walks. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.
- [23] M. Meilă, S. Shortreed, and L. Xu. Regularized Spectral Learning. In *Artificial Intelligence and Statistics (AISTATS)*, 2005.
- [24] B. Mikhail and R. J. Mooney. On evaluation and training-set construction for duplicate detection. In *KDD-2003 workshop on data cleaning, record linkage, and object consolidation*, 2003.
- [25] E. Minkov and W. W. Cohen. Learning Graph Walk Based Similarity Measures for Parsed Text. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2008.
- [26] E. Minkov, W. W. Cohen, and A. Y. Ng. Contextual Search and Name Disambiguation in Email using Graphs. In *Conference on Research and Development in Information Retrieval (SIGIR)*, 2006.
- [27] H. Pasula, B. Marthi., B. Milch, S. Russell, and I. Shpitser. Identity uncertainty and citation matching. In *Advances in Neural Information Processing Systems (NIPS)*, 2003.
- [28] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, August 2000.
- [29] P. Singla and P. Domingos. Entity Resolution with Markov Logic. In *International Conference on Data Mining (ICDM)*, 2006.
- [30] K. Toutanova, C. D. Manning, and A. Y. Ng. Learning random walk models for inducing word dependency distributions. In *International Conference on Machine Learning (ICML)*, 2004.
- [31] K. Tsuda, H. Shin, and B. Schölkopf. Fast protein classification with multiple networks. *Bioinformatics*, 21(2):59–65, 2005.
- [32] O. Tuason, L. Chen, H. Liu, J. Blake, and C. Friedman. Biological nomenclatures: a source of lexical knowledge and ambiguity. In *Pacific Symposium on Biocomputing*, volume 2004, pages 238–49, 2004.
- [33] U. von Luxborg. A Tutorial on Spectral Clustering. Technical Report 149, Max Planck Institute for Biological Cybernetics, August 2006.
- [34] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. In *International Conference on Machine Learning (ICML)*, 2003.