Applying Link-based Classification to Label Blogs[*]

Smriti Bhagat
smbhagat@cs.rutgers.edu

Irina Rozenbaum
rozenbau@cs.rutgers.edu

Graham Cormode
graham@dimacs.rutgers.edu

ABSTRACT

In analyzing data from social and communication networks, we encounter the problem of classifying objects where there is an explicit link structure amongst the objects. We study the problem of inferring the classification of all the objects from a labeled subset, using *only* the link-based information amongst the objects.

We abstract the above as a labeling problem on multigraphs with weighted edges. We present two classes of algorithms, based on local and global similarities. Then we focus on multigraphs induced by blog data, and carefully apply our general algorithms to specifically infer labels such as age, gender and location associated with the blog based only on the link-structure amongst them. We perform a comprehensive set of experiments with real, large-scale blog data sets and show that significant accuracy is possible from little or no non-link information, and our methods scale to millions of nodes and edges.

Categories and Subject Descriptors: I.2.6 [Learning]: Graphs

General Terms: Algorithms, Experimentation.

Keywords: Graph labeling, Relational learning, Social Networks.

1. INTRODUCTION

In recent years there has been rapid growth in the massive networks which store and encode information—most obviously, the emergence of the World Wide Web, but also in the social networks explicit or implicit from collections of emails, phone calls, blogs, and services such as myspace and facebook. It is now feasible to collect and store such data on a scale many orders of magnitude greater than initial hand-curated collections of friendship networks by sociologists.

Many fundamental problems in analyzing such data can be modeled as instances of classification. That is, we wish to attach a label to each entity in the data based on a small initially labeled subset. This is motivated by applications as varied as (web) search, marketing, expert finding, topic analysis, fraud detection, network service optimization, customer analysis and even matchmaking. The problem of "link-based classification" [10], or "relational learning" [19], has received significant study: we later survey the most relevant works. However, such studies have typically been over relatively small collections, i.e. thousands of carefully curated data items such as movies, companies, and university web pages.

In this paper, we address the challenge of labeling network data in the face of substantially larger datasets, motivated by the examples of large social networks. In particular, we study the networks induced by blogs, and the rich structure they create, although our methods and observations are sufficiently general to be more widely applicable. Our choice of blogs is motivated by our earlier experiences with this data source [3]: working closely with this data reveals questions to serious analysis, such as:

- How to work across multiple networks?

Typically, entities of interest do not reside within a single network, but are spread across multiple interlinked networks. For example, blogs are hosted by multiple different providers, and link offsite to websites; in the telecoms world, calls are handled by a mixture of long distance and wireless providers. Although one can focus in on just a single network, this will often lose vital information, and lead to a disconnected view of the data. Instead, one needs to work across networks (and hence develop data collection techniques for each network type) to see a fuller picture, and recognize that different (sub)networks have different behavior.

- What information may be used to infer labels?

We focus on situations where there are explicit links between objects, and study how to use these to propagate labels. We use the fact that there is often some amount of initially labeled data, such as customer information in telecoms, or profile information for blogs. In some applications, there may be additional features, such as text in blogs. However, in many cases we can obtain no other features apart from graph edges: for technical, legal, or privacy reasons we often see only the links, and little additional data associated with them. In this study, we explore the power of inference based solely on link information, and conclude that for many applications this alone can give a powerful result.

- How to collect data and ensure data quality?

Data collection is a non-trivial part of applying any classification algorithm. Especially when collecting data self-reported by individuals, (e.g. from web repositories), there are issues of reliability in trustworthiness in the collected values. Additionally, one has to be aware of an inherent bias in the data due to the perspective of the observer, since some networks are easier to poll, or yield more information, than others. For example, we must choose which subset of blog hosting sites to crawl and write parsers for, and what crawling strategy to employ.

- How to scale to large data sizes?

With numbers of nodes in the hundreds of thousands, and edges in the millions, many sophisticated structural analysis techniques, such as matrix-style decompositions, fail to scale. Moreover, we also need methods which can provide relatively simple explana-

[*]Supported by DyDAn, ONR grant number N00014-07-1-0150.

tions of their inferences if we are to justify them to decision makers. Here, we seek to understand the power of relatively simple methods whose computational cost is near linear in the input size.

Our Contributions. Our study here is on the problem of inferring the classification labels for objects with explicit structure linking them. We can view this a problem over enormous graphs, or more correctly, multigraphs, due to the multiple object types, edge types, and multiple edges inherent in any detailed study of such networks. Our main contributions are as follows:
1. We formalize our problem of graph labeling as an instance of (semi-)supervised learning, and introduce two simple classes of algorithms, *local iterative* and *global nearest neighbor*.
2. We give a detailed description of data collection and modeling in order to apply these algorithms to blog data and to infer labels such as age, location and gender, given the challenges outlined above.
3. We show experimental results with large scale blog data that demonstrate these methods are quite accurate using link information only, and are highly scalable: for some tasks we can accurately assign labels with accuracy of 80-90% on graphs of hundreds of thousands of nodes in a matter of tens of seconds.

Our chosen methods succeed due to inherent structure in the blog network: people tend to link to others with similar demographics, or alike people link to similar objects. It remains a challenge to extend them to situations where such behavior is less prevalent, and understand fully when they are and are not applicable.

Outline. In what follows, we will describe the abstract problem of labeling (multi)graphs and our two classes of solutions in Section 2. In Section 3, we will describe the study case of blogs and describe how our algorithms are applicable for inferring labels on blogs. In Section 4, we present extensive experiments with large scale blog data to infer age/location/gender of blogs. Related work is discussed in Section 5 and concluding remarks are in Section 6.

2. GRAPH LABELING

2.1 Problem Formulation

We now define the problems of graph and multigraph labeling. Given a graph, with a subset of nodes labeled, our goal is to infer the labels on the remaining nodes. More formally,

DEFINITION 1. *Let G be a partially labeled directed graph $G = (V, E, M)$, where V is the set of vertices or nodes and E is the set of edges as usual. Let $L = \{\ell_1, \ell_2, ..., \ell_c\}$ be the set of labels, where a label ℓ_k can take an integer or a nominal value, and $|L|$ is the number of possible labels. $M : V \rightarrow L \cup \{0\}$ is a function which gives the label for a subset of nodes $W \subset V$; for nodes $v \notin W$, $M(v) = 0$, indicating that v is initially unlabeled. The* Graph Labeling Problem *is, given the partially labeled graph G, to complete the labeling: i.e., to assign labels to nodes in $U = V \backslash W$.*

This abstract problem captures our core question: how to use the link structure of the graph and the partial labels in order to infer the remaining labels. This can be seen as an instance of Relational Learning [10], since a graph can be encoded as a number of (many-to-many) relations. However, we find it more useful to phrase the problem in terms of graph nodes and edges, since this is how our input is presented. This is also fundamentally a *semi-supervised* classification problem, since in our motivating applications it is rare to find any representative completely labeled graph; instead, we are given one large graph with a subset of its nodes labeled. Connections to related work are discussed further in Section 5.

In practice, the data that we collect is richer than this, and does not fit into the simplistic graph model. We may see different kinds of nodes, relating to different entities. For example, blogs have links to webpages which are quite different from blog pages. When we have multiple types of node, and different kinds of edge connecting them, we have a *multigraph* version of the problem. An important detail is that for some node types, we may have more or less information than others. This is a result of how much can be sampled or observed in the world. For example, in the telecommunications world, service providers can observe both incoming and outgoing calls for their customers, but do not see calls between customers of other providers. As a result the graph a provider sees may not contain all the outgoing/incoming edges of some of the nodes. Likewise in blog or web analysis, one may know all outgoing edges for each page, but not all the incoming edges. This gives an unavoidable bias that is due to problems of *observability* (some links are not observable) and *collectability* (it is not feasible to collect and store every last link).

Formally, we may have a multigraph, $G^+ = (V^+, E^+)$, where V^+ is partitioned into p sets of nodes of different types, $V^+ = \{V_1, V_2, ..., V_p\}$, and E^+ is a (weighted) collection of sets of edges $E^+ = \{E_1, E_2, ..., E_q\}$. We may have additional features F on each node, and a function w giving the weight of each edge or set of edges. Other variations, such as features and labels on edges are possible, but we omit them to keep focus on the underlying problems. Some examples of multigraphs in different settings are:

- **Telecommunications:** The nodes of the multigraph may represent distinct phone numbers, and the edges represent telephone calls made between two phone numbers (clearly directional). One node type may represent 1-800 numbers that can only receive calls, while the other nodes are consumer accounts. There can be multiple edges between nodes and multiple kinds of edges (long distance calls, local calls and toll free calls). A suitable label in this example is to classify the numbers as business/non-business. Typically telephone companies have a business directory to populate labels on a subset of nodes, and in some cases, use human evaluation to label some nodes too.
- **IP networks:** In the IP network setting, a node could represent a distinct IP address, a segment of IP addresses or an ISP. An edge between two nodes may signify any kind of IP traffic detected between the two nodes, traffic belonging to a certain application or protocol, certain types of messages, etc. A suitable label in this case is based on the network node's function as a server or a client. Typically Internet Service Providers have a list of known or suspected servers which is the initial set of labels from which we need to infer the classification of server/client for remaining nodes.
- **Web:** The World Wide Web can be represented by a multigraph, where nodes are webpages that can be further categorized by ownership, functionality or topic [5], and an edge between two nodes signifying an HTML link from one web page to another. Links could also be categorized: e.g., an edge from a site to a commercial company website could signify presence of the company's advertisement on the website. In this setting, suitable node labels could be based on the site being public or commercial, or the site's function (portal, news, encyclopedia, etc). The class of some nodes are known, and these can be used to label the remaining nodes. □

2.2 Algorithmic Overview

We principally study two classes of algorithms for predicting labels on multigraphs, namely, *Local Iterative* and *Global Nearest Neighbor*. These classes have their antecedents in prior work on relational learning [12] and the well-known general purpose classifiers such as Nearest Neighbors. They are designed to be relatively simple to implement and scale to very large graph data. Both the local and global approaches use the link structure and neighborhood

Algorithm 2.1: LOCALITERATIVE(E, M, s)

```
$B^0 \leftarrow \mathbf{0}$
$M^0 \leftarrow M$
for $t \leftarrow 1$ to $s$
  do
    for $i \in U$
      do
        for $(i,j) \in E$
          do
            $k \leftarrow M^{t-1}(j)$
            if $k \neq 0$
              then $B^t_{ik} \leftarrow B^t_{ik} + 1$
    for $j \in V$
      do
        if $j \in U$
          then $M^t(j) \leftarrow M(j)$
          else $M^t(j) \leftarrow \mathrm{voting}(B^t_{(j)})$
    $B^{t+1} \leftarrow B^t$
```

information to infer labels. For each unlabeled node they examine a set of labeled nodes and select one of these labels as the new label. The approaches differ in the set of nodes considered for inferring labels. We call these 'classes of algorithms' as there are many feasible variations in each class which do not change the fundamental character of the algorithms.

Preliminaries. Our algorithms take as input a description of the (multi)graph and any features and labels. Since the graphs we consider may become very large in size, typically the description will be in the form of an adjacency list or similar set of edges. However, it is often convenient to give the description of the algorithms in adjacency matrix notation. Let A be an $n \times n$ adjacency matrix representing a graph $G = (V, E)$, where $a_{ij} = 1$ if $(i,j) \in E$ and 0 otherwise (more generally, a_{ij} can be a weight of an edge from i to j if any); $n = |V|$ is the number of nodes. Let $A_{(i)}$ denote the i^{th} row of matrix A and $A^{(j)}$ denote the j^{th} column of A, where $i, j \in [1, n]$. For a function f whose domain is $\mathrm{dom}(f)$ we use the notation $\mathrm{diag}(f)$ as shorthand for the $\mathrm{dom}(f) \times \mathrm{dom}(f)$ matrix such that $\mathrm{diag}(f)_{ii} = f(i)$, and is zero elsewhere.

The neighborhood of each i node, defined by the immediately adjacent nodes, is encoded as a *feature vector*, $B_{(i)}$, based on the link structure of i (in general, the feature vector could also include other features of the node, but here we focus on a core link-based classification problem). Typically, this feature vector is initialized to represent the frequency of the labels on the nodes in its neighborhood. From these vectors, we create an $n \times c$ feature matrix B. Given a function f mapping from n to c, let $\chi(f)$ denote the characteristic matrix of f, i.e. $\chi(f)_{il} = 1$ iff $f(i) = l$. We can write $B = A\chi(M)$, where M is the initial labeling. In such graph labeling problems we potentially *change* the feature vectors as we make inferences and label nodes—when the neighbor of node i is labeled, the feature vector of i changes to reflect this new label—to enable the propagation of labels to all nodes.

We consider only directed graphs, since the natural extension of these algorithms to the undirected case is equivalent to the directed case where each directed edge is present in the reverse direction. However, because of this directionality, nodes with no incoming edges have no neighborhood information for us to predict from (they have an empty feature vector); hence such nodes may remain unlabeled, or can be labeled with most likely label from the prior distribution. In some networks, link reciprocity (a bi-directional link) is an important indicator of a stronger connection; this can be captured giving an increased weight to reciprocal links.

2.3 Local Iterative Methods

With only a graph and some initial labels, there is limited information available to help us propagate the labeling. We therefore make some assumptions about how nodes attain their labels. Our local methods assume the existence of *homophily*: that links are formed between similar entities so that the label of a node is a function of the labels of nearby nodes [13]. The algorithms succeed if the following hypothesis holds:

$$\boxed{\text{Nodes link to other nodes with similar labels.}}$$

Thus, we view each incoming edge in a directed graph as representing a "vote" by an adjacent node. Let ℓ_k be the label on node u. An edge from node u to node v implies node u is a vote for the label ℓ_k for node v. For each unlabeled node, the votes from its neighbors are combined to derive its label by a function $\mathrm{voting}(B_{(i)})$. When performed iteratively, recalculating feature vectors each step, this leads to propagation of labels. There are many variations, based on the voting function used to assign the label based on the feature vector, the neighborhood used to generate the feature vector, and so on. Intuitively, the simplest voting scheme is "plurality voting" which chooses the most frequently voted label by the immediately adjacent nodes of an unlabeled node (as in [12]). We can also consider other functions, such as taking the median or average label drawn from an ordered domain, and so on.

In each iteration, we visit each node $i \in U$, and determine a label for it based on its current neighborhood, rather than fixing the label for an unlabeled node once and for all. A node may receive different labels in different steps, with different levels of confidence in each based on the round in which the label is assigned.

Formal Definition. Let A be the adjacency matrix representation of the graph. At each iteration, we compute a new labeling function M^t such that for every (unlabeled) node $i \in U$, the label determined based on voting by its neighbors is assigned to $M(i)$. To label the nodes, at iteration t, M^t is defined by:

$$M^t(i) \leftarrow \mathrm{voting}(B^t_{(i)})$$

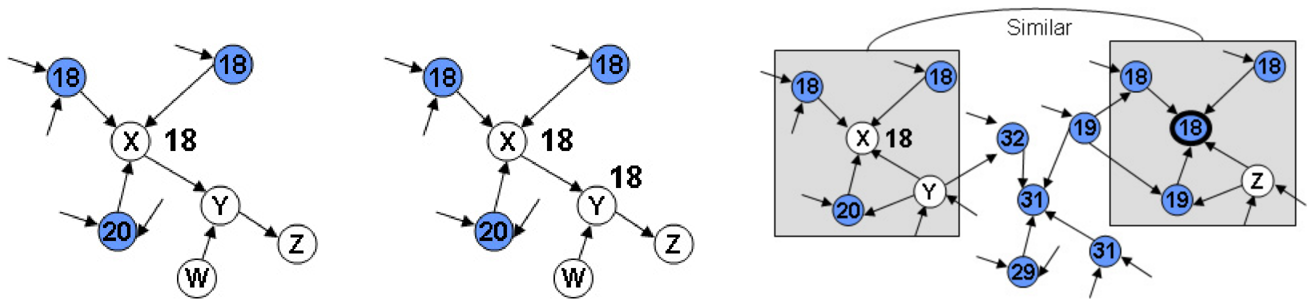
where B^t is the feature matrix for the t-th iteration, defined by:

DEFINITION 2. *Let $M^t : V \to L \cup \{0\}$ denote the labeling function on the t-th iteration (we insist that $M^t(i) = M(i)$ for $i \in W$). Let $\mathrm{conf}^t : V \to \mathbb{R}$ be a function from nodes denoting the relative confidence of the algorithm in its labeling at the t-th iteration. We set $M^0 = M$ and $\mathrm{conf}^0(i) = 1$ for all $i \in W$, zero otherwise. Lastly, let $\mathrm{decay} : \mathbb{N} \to \mathbb{R}$ be a function which returns a weighting for all labels assigned a iterations ago. We define the* iterative feature vector at iteration *t, B^t as*

$$B^t = A \sum_{t'=0}^{t-1} \mathrm{decay}(t - t')\chi(M^{t'})\,\mathrm{diag}(\mathrm{conf}^{t'}) \qquad (1)$$

We consider a simple setting of these functions: our instantiation assigns equal confidence to every labeling (i.e. $\mathrm{conf}^t(i) = 1$ for all i, t), and equal weighting for all label values (i.e. $\mathrm{decay}(x) = 1$ for all x), which favors labels assigned earlier in the process. This makes B^t convenient to compute: (1) simplifies to $B^t = B^{t-1} + AM^{t-1}$, and so can be computed in place without keeping a history of prior values of $B^{t'}$. Note that the absolute values of entries in B^t are unimportant, just the relative values, due to our application of voting via $\arg\max$ (thus one can normalize and view B^t as a distribution on labels, and the voting take the maximum likelihood label). We set a bound on the number of iterations performed before terminating as s. Algorithm 2.1 shows pseudocode for the local iterative method and we summarize its performance (proofs omitted for brevity):

LEMMA 2.1. *The local iterative algorithm can be implemented so each iteration runs in time linear in the input size, $O(|V|+|E|)$.*



(a) Local: Iteration 1 (b) Local: Iteration 2 (c) Global Nearest Neighbor

Figure 1: Graph Labeling Examples

Example. Figures 1(a) and 1(b) show the working of the local iterative approach on a small example, with $\arg\max$ as the voting function. Nodes X, Y, Z and W are unlabeled. The feature of vector of node X records a frequency of 2 for label '18', and 1 for label '19'. So in the first iteration, node X is labeled '18' based on the votes by its neighbors. In the following iterations, labels are propagated to nodes Y and Z. Since node W does not have any incoming edges, it remains unlabeled. $\square$

Iterations. The number of iterations should be chosen large enough so that each node has the chance to get labeled. This is bounded by the diameter of the (directed) graph and the graphs we consider are typically "small world", and so have a relatively small diameter—often, $O(\log |V|)$. Additional iterations increase our confidence in the labeling, and allow the labeling to stabilize. Although is possible to create adversarial examples where the iterative process never stabilizes (because the label of a node cycles through several possible labelings), such extreme cases do not occur in our experiments.

Multigraph Case. We highlight some important issues when there are multiple node and edge types:

• **Pseudo-labels.** In multigraphs, we may have multiple classes of nodes, where the label only applies to certain of them. In our telecoms example, it is not meaningful to classify the calling patterns of certain numbers (such as 1-800 numbers which are for incoming calls only). Instead of omitting such nodes, we use the notion of *pseudo-labels*: allocating labels to nodes by the iterative method even if these labels are not wholly meaningful, as a means to the end of ensuring the nodes of interest do receive meaningful labels. This generalization turns out to be very important in propagating labels to otherwise isolated nodes.

• **Edge weights.** With different types of edge, and different numbers of edges between nodes, it is natural to introduce edge weights, and modify the feature vectors by using these weights to scale the votes. These details are mostly straightforward.

• **Additional Features.** In some cases we have additional features F attached to certain nodes or edges. It is not completely obvious how to cleanly extend the iterative approach to incorporate these. One direction is to use an appropriate distance function to measure the similarity between the features of pairs of nodes connected by an edge, and re-weight the voting based on the similarity: the votes of more similar nodes count more than others. However, the specification of the distance and weighting schemes is non-trivial, and is left for future work.

2.4 Global Nearest Neighbor

The global nearest neighbor family of methods uses a different notion of proximity to the local algorithm to find labels. By analogy with the traditional k-Nearest Neighbor classifier, we consider the set of labeled nodes and take the labels of the k-best matches. The matching is based on the similarity of the feature vectors, i.e. the labels of the neighboring nodes. The underlying hypothesis is one of *co-citation regularity* [10], that is:

Algorithm 2.2: GLOBAL1NN(E, M)

```
$B_{n\times c} \leftarrow \mathbf{0}$
$S_{n\times N} \leftarrow \mathbf{0}$
for $(i,j) \in E$
    do { $k \leftarrow M(j)$
         if $k \neq 0$
           then $B_{ik} \leftarrow B_{ik} + 1$
for $i \in U$
    do { for $j \in W$
           do {$S_{ij} \leftarrow sim(B_{(i)}, B_{(j)})$}
         $k \leftarrow \arg\max(S_{(i)})$
         $M(i) \leftarrow M(k)$
```

Nodes with similar neighborhoods have similar labels.

Example. Figure 1(c) gives an example of labeling by the global 1-nearest neighbor approach. Of all labeled nodes, the neighborhood of nodes X is most similar to that of the highlighted node with label 18. The algorithm assigns the label 18 to node X. A similar nearest neighbor search is repeated for all unlabeled nodes. $\square$

Formal Description. As before, consider the adjacency matrix A and an initial labeling function M, which define a feature matrix B. Let $S_{n\times n}$ be a similarity matrix. For each unlabeled node i, compute the similarity coefficient S_{ij} between $B_{(i)}$ and $B_{(j)}$, for each labeled node j. Node i is assigned the most frequent label of the k nodes with the highest similarity coefficients i.e. the label of node $\arg\max(S_{(i)})$.

Choice of Similarity Function. Given two vectors x and y, there are many possible choices, such as the L_p distances: Euclidean distance, $\|x-y\|_2$, and Manhattan distance, $\|x-y\|_1$. We employ Pearson's correlation coefficient,

$$C(x,y) = \frac{n x\cdot y - \|x\|_1\|y\|_1}{\sqrt{n\|x\|_2^2 - \|x\|_1^2}\sqrt{n\|y\|_2^2 - \|y\|_1^2}}.$$

Intuitively, the correlation coefficient is preferred over Euclidean distance when the shape of the vectors being compared is more important than the magnitude.

In the multigraph case, we can naturally take into account different nodes and edges (V^+, E^+) and features F by keeping the algorithm fixed and generalizing the similarity function. For set valued features, we can compare sets X and Y using measures such as Jaccard ($J(X,Y) = \frac{|X\cap Y|}{|X\cup Y|}$). The similarity function can combine the similarities of each feature. For example, we later utilize a weighted combination of Jaccard coefficient (for features represented as sets) and correlation coefficient (for vector features).

Algorithm 2.2 shows the pseudocode for the global nearest neighbor method, and we summarize:

LEMMA 2.2. *The total running time for the global nearest neighbor method is $O(|U||W||L| + |E|)$.*

Approximate Nearest Neighbors. The above lemma assumes a naive exhaustive comparison of every labeled node with every unlabeled node. For appropriate similarity functions, this can be accelerated via dimensionality reduction and approximate nearest neighbors algorithms to find nodes that are approximately the closest [9].

Multi-pass Generalization. As defined above, the global nearest neighbor algorithm takes a single pass and attempts to assign a label to every unlabeled node based on the initially labeled neighborhoods. This could result in poor inferences when a node has few, if any labeled neighbors. As in the iterative case, it is possible to define a multi-pass algorithm, which bases its conclusions on labels (and confidences) defined in the previous iteration; here we focus on the single-pass version for brevity.

3. LABELING BLOGS

3.1 Anatomy of a Blog

A blog is typically a web-based journal, with entries (posts) displayed in reverse chronological order. Postings are publicly viewable, and readers may provide immediate feedback by adding a comment. Websites offer blog hosting with a variety of user interfaces and features. Blogs commonly include information about the owner/author in the form of a *profile*, in addition to the blog entries themselves.

User Profile. When users opens accounts at a blog hosting site, they are asked to fill out a user profile form with age, gender, occupation, location, interests (favorite music, books, movies, etc.). In some cases, the user can also provide an email address, URL of a personal website, Instant Messenger ID's, etc. Most of this information is optional. Some services only reveal some information to a set of "friends" (accounts on the same service); this list of friends may be visible to all.

Blog Entries. The blog owner posts blog entries which contain text, images, links to other websites and multimedia etc. They are typically accompanied by the date and time each entry was made. Blog postings often reference other blogs and websites (as illustrated in Figure 2(a)). Bloggers can also utilize special blog sections to display links of particular interest to them, such as "friends", "links", "subscriptions", etc.

3.2 Modeling Blogs As Graphs

There are many ways to extract a (multi)graph from a collection of blog data. We outline some of the choices in the modeling and extraction of features.

Nodes. We can encode blogs as graph nodes at several granularities: we can treat each blog posting and comment as separate nodes, or consider all postings within a single blog as constituting a single node. There is also some subtlety here, since some blogs may have multiple authors, and single authors may contribute to multiple blogs. However, the common case is when a single author has a single blog. We adopt the model where the entirety of a blog is represented by a single node. Additional nodes represent webpages connected to blogs.

Edges. We use the (web)links to define the edges in the blog graph: a directed edge in the blog graph corresponds to a link from the blog to another blog or website. We can automatically categorize these links according to the destination and where they appear within the blog pages: a link appearing in a blog entry, a link appearing in a comment posted as a response to a blog entry, a link in the "friends" category, etc. These define various sets of edges: E_F, based on explicit friend links, E_B, containing all other links to blogs, and E_W, containing links from blogs to websites.

Labels. Having defined the nodes and edges, we consider a variety of labels. In full generality, we can consider almost any label that can be attached to a subset of the nodes and propagated by our algorithms. In our study, we restrict ourselves to labels based on components of the user profile. These cover a broad set of different label types (binary, categorical, continuous), and ensure that we can use collected data to define training and test data. We consider the following labels:

• **Age**— Blog profiles typically invite the user to specify their date of birth, and a derived age is shown to viewers. But the "age" we attach to a blog can have multiple interpretations: the actual age of the blog author, the "assumed" age of the author, the age of the audience, and so on. We will evaluate our algorithms at matching given age labels that are withheld from the algorithm.

• **Gender**— Gender is another natural profile entry to attempt to propagate. Prior work has looked for text and presentation features [18] in order to predict gender; here, we aim to use link information only. Gender has multiple interpretations similar to age.

• **Location**— Lastly, we include the (stated) location of the author, at the granularity of continents (category with seven values) or country (category with over two hundred possible values).

Many other labels are possible, but we focus on these three, since they demonstrate common label types, and are available in many blog profiles for evaluation. Figure 2 illustrates possible graphs on the same set of nodes, showing different labels.

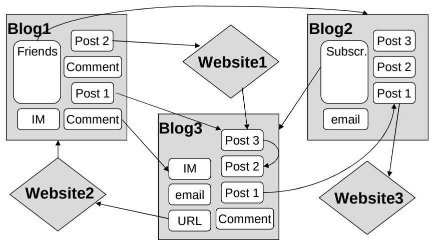
3.3 Algorithms Applied to the Blog Graph

We represent a collection of blogs and the links between them by a graph over a set of nodes V_B. Each node $v \in V_B$ corresponds to a blog user (identified by a unique user id). We examined the three label types: age, gender and location (by country or by continent). Label values are extracted from the blog owner's profile if present. In applying our algorithms, different issues arise for each label:

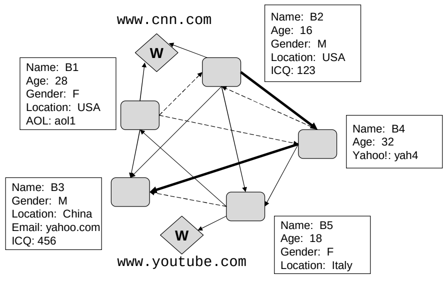
Age. When working with age label, our hypotheses translate to *bloggers tend to link to other bloggers of their own age* (local iterative) and *bloggers of the same age link to bloggers of similar age distributions* (global nearest neighbor). Both of these seem plausible, but distinct. The initial feature matrix, $B_{n\times 120}$, encodes the frequency of adjacent ages in years for each node. Because age is a continuous attribute, we smoothed each feature vector by convolution with the triangular kernel $[0.2, 0.4, 0.6, 0.8, 1.0, 0.8, 0.6, 0.4, 0.2]$. This improved the quality of the observed results, so all experiments shown use this kernel. Our default similarity method is the correlation coefficient.

Location. The hypotheses for the location label are that *bloggers tend to link to other bloggers in their vicinity* (local) and *bloggers in the same locale link to similar distributions of locations* (global). The former hypothesis seems more intuitively defensible. Feature vectors encode 245 countries belonging to seven continents.

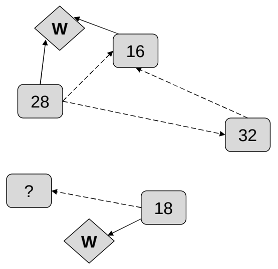
Gender. For gender, the iterative assumption is *bloggers link to bloggers of their own gender* (local) or *bloggers of the same gender link to similar patterns of genders*. Neither seems particularly convincing, and indeed for gender we saw the worst experimental results. This is partly due to using the desired label as the (only) feature in the classification. As noted in Section 2.3, our meth-



(a) Links between blogs and websites

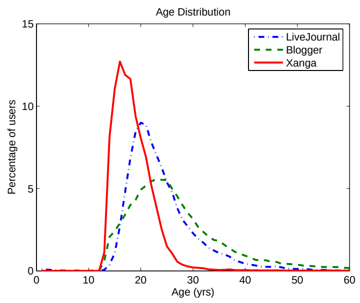


(b) Blog Multigraph and user profiles

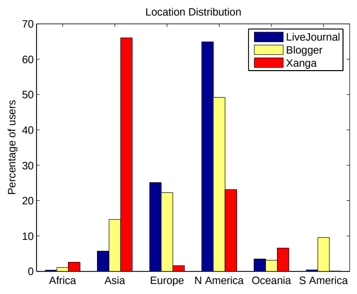


(c) Subgraph with Age labels

Figure 2: Multigraphs extracted from blog data



(a) Age Distribution



(b) Location Distribution

		Blogger	LiveJournal	Xanga
Blog	Nodes	453K	293K	784K
Blog	Edges	534K	411K	2,997K
Web	Nodes	990K	289K	74K
Web	Edges	2,858K	1,089K	895K
Friend Edges		—	3,977K	64K
	Age	47K	124K	493K
Labels	Gender	113K		580K
	Location	111K	242K	430K

(c) Summary of data used for blog graphs

Figure 3: Label and Data Collection Summary

ods allow other features to be included and we hypothesize that including additional features (such as the age and location, if known) could improve the learning. We defer such studies to future work.

Multigraph Edges. We tried several weighting schemes for edges to reflect the relative importance attached different link types (blog, friends, web), and studied the impact this can have on the quality of the resultant labeling. Many weightings are possible; for brevity we report only the following settings: Blog-blog links only (all edges in E_B have weight 1, all other edges have weight 0); Friends only (E_F edges weight 1, others 0); Blogs and friends (E_B and E_F have weight 1); Blogs and web (E_B and E_W have weight 1).

Multigraph Nodes. In addition to nodes corresponding to blogs, we include additional nodes V_W corresponding to (non-blog) websites. These turn out to be vital in propagating labels to otherwise weakly linked subgraphs. To make use of the webpage nodes, we replicate edges to webpages in the reverse direction. All webpage nodes are initially unlabeled, and the local iterative method therefore assigns a *pseudo-label* to all of these nodes in the course of its operation. Although some labels such as Location or Age have unclear semantics when applied to webpages, it is possible to interpret them as a function of the location and age of the bloggers linking to the webpage. This can be viewed as applying co-citation regularity in the iterative model, allowing labels to be transfered from one blog to another via these intermediaries.

The Global Nearest Neighbor algorithm takes a different approach to using the nodes in V_W. Since no such nodes are initially labeled, they would play no part in the (single-pass) algorithm even if we assign them pseudo labels. Instead, we effectively treat the links to nodes in V_W as defining a set of (sparse, high dimensional) binary features. We therefore extend the similarity function between two nodes, as suggested in Section 2.4 as a weighted sum of the (set) similarity between V_W neighborhoods and (vector) similarity between V_B neighborhoods. For nodes $i \in U$ and $j \in W$ the similarity coefficient is:

$$S_{ij} = \alpha \times C(B_{(i)}, B_{(j)}) + (1-\alpha) \times J(V_{W(i)}, V_{W(j)})$$

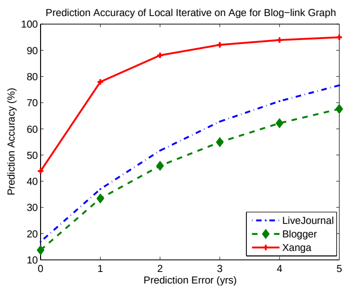
for some $0 \le \alpha \le 1$, where $B_{(i)}$ is the feature vector of the node i, and $V_{W(i)}$ is the set of web nodes linked to the blog node i.

4. EXPERIMENTS

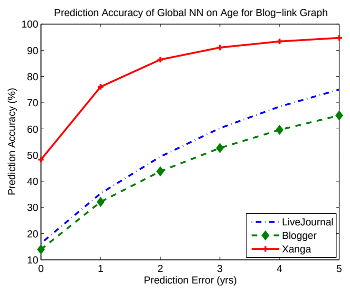
4.1 Experimental Setup

Data Collection. In our experiments we used data collected in Summer 2006 by crawling three blog hosting sites: Blogger, LiveJournal and Xanga. The data consists of two main categories: user profiles containing various personal information provided by the user; and blog pages for recent entries in each crawled blog. We created an initial seed set of blogs and profiles by randomly identifying a subset of blogs hosted by each site. This initial seed set was expanded by downloading blogs (and corresponding profiles) referenced from the initial set. Our final data set therefore consists of (a subset of) blogs from each of the three crawled sites, corresponding profiles, and extracted links between blogs, and to webpages. Each web node corresponds to a single domain name (so links to `http://www.cnn.com/WEATHER/` and `http://www.cnn.com/US/` are counted as `www.cnn.com`). This improves the connectivity of the induced graph. We did not extract links from webpages back to blogs, since these were very rare. The results for the number of user profiles collected and the number of links extracted are shown in Table 3(c).

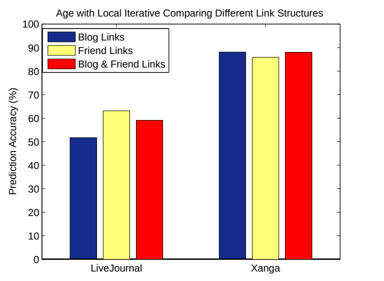
Demographics. We plotted the age, gender and location distribution of the users in our data set for each blog site. Ages claimed by blog users range from 1 to 120, with a small fraction of extreme values (on average, less than 0.6% with ages above 80). We did not filter implausible values, and this did not seem to impact re-



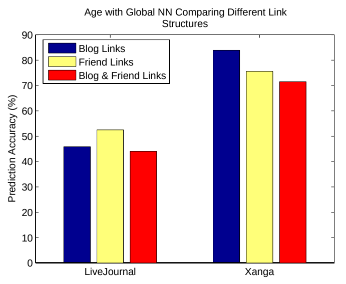
(a) Age with Local Iterative on Blog Links



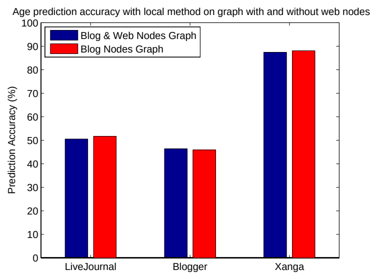
(b) Age with Global NN on Blog Links



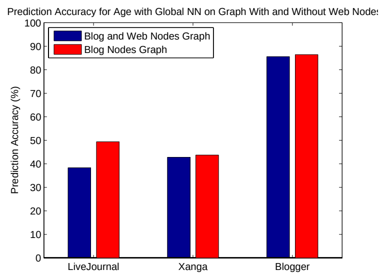
(c) Age with Local Iterative, Different Links



(d) Age with Global NN, Different Links



(e) Age with Local Iterative using Web Nodes



(f) Age with Global NN using Web Nodes

Figure 4: Experiments of Accuracy in Finding Age Labels

sults adversely. Not all users reveal their age, while some reveal only birthday, but not birth year. It can be observed from the distribution of ages in Figure 3(a) that Xanga has the youngest user population of the three networks, while Blogger has the most mature population among the three. Among the users that listed their gender, 63% of Xanga users are females while Blogger has only 47% of female users. LiveJournal does not make gender information available. Each of the three hosting sites we crawled offered a different format for specifying location. Blogger allows free text in location fields “City/Town” and “Region/State”, and LiveJournal allows free text in “City”; as a result many users provide nonsensical values. Xanga organizes its users into “metros” according to geographical hierarchy (continent, country, city). Figure 3(b) shows the distribution of locations by continent. Additional details on the data collection process and analysis of the data can be found in [3].

Label Distribution. We analyzed the average and the median number of labeled neighbors in our multigraph for each of the nodes per data source, taking into account only incoming edges. In Blogger, the average number of labeled neighbors is 0.45 per node (most have 0). In Livejournal and Xanga the average is 7.5 and 3.1 respectively; the median is 2 in Livejournal and 1 in Xanga. Including friend links did not make a significant difference.

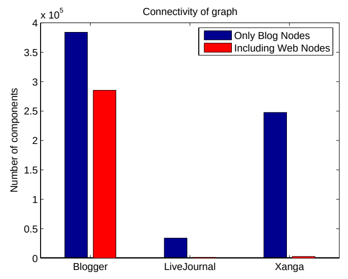
Implementation Issues. We implemented our algorithms in C++ and performed a detailed set of experiments to compare the performance of our methods on the blog data. In each task, the blog nodes are labeled with one of the three types of label: continuous (age), binary (gender), nominal (location). We also varied the multigraph by setting different weights for the link types, discussed in Section 3.3. For the Iterative local algorithm we set the number of iterations, s, to five, and the voting function to $\arg\max$ (plurality voting). For Global nearest neighbor, we used correlation coefficient as the similarity function, with weighting factor $\alpha = 0.5$ when including web nodes as features. In each experimental setting, we performed 10-fold cross validation, and report the average scores over the 10 runs: the set of labeled nodes is further divided into 10 subsets and evaluation is in turn performed on each subset using the remaining 9 for training. Across all experiments, the results were highly consistent: the standard deviation was less than 2% in each case. Although we run our experiments on the entire data set, we are able to evaluate only on the labeled subset (which is different for each label type).

4.2 Accuracy Evaluation

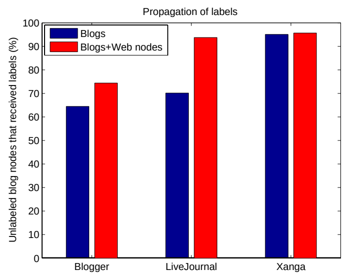
Age label. Figure 4 summarizes the various experiments performed while labeling the blog nodes with ages 1 to 120. We evaluate against the *stated* age in the blog profile. The features used by the two algorithm classes, Local Iterative and Global NN, are derived from the labels on the training set. We observe that with this information alone it is possible to attain an accurate labeling in blog data. Note that due to the graph structure, some nodes have empty feature vectors due to lack of links and, no matter how we propagate information, will never have any useful features with which to label. We therefore exclude such nodes from our evaluation.

Figure 4(a) shows the performance of the Local Iterative method for different accuracy levels from exact prediction to predicting within five years. The predictions for LiveJournal and Blogger show that with label data alone, it is possible to label with accuracy about 60% and 50% respectively within 3 years difference of the reported age. For the Xanga dataset, which is the most densely connected, we saw results that are appreciably much stronger: 88% prediction accuracy within 2 years off the reported age. Figure 4(b) shows a similar plot for Global NN algorithm. The prediction accuracy is not significantly different than the Local Iterative method. We observed that both methods tended to make accurate predictions for the same set of nodes.

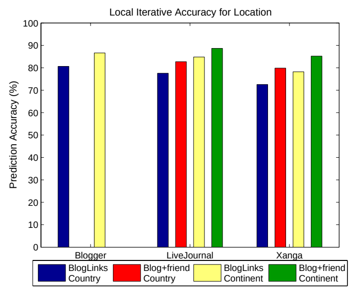
Multigraph Edges. The remainder of the plots in Figure 4 compare the accuracy with the inclusion of additional edge types at unit weight. For LiveJournal, both the local and global methods benefit from using just the friend links (Figures 4(c) and 4(d)), suggesting that these edges conform more strongly to the hypotheses.



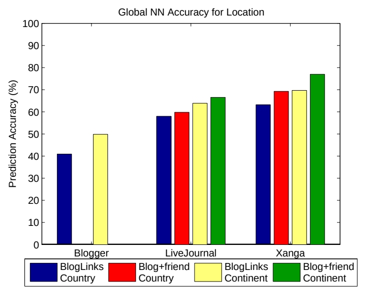
(a) Distribution of component size



(b) Fraction of nodes assigned labels



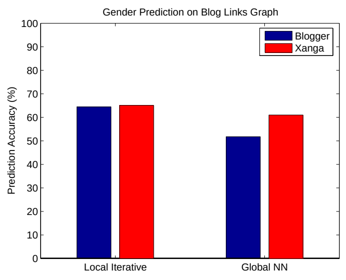
(c) Location with Local Iterative



(d) Location with Global NN

Predicted →	NA	SA	As	Eu	Af	Oc
N America	**92**	1	2	3	0	1
S America	4	**90**	0	6	0	0
Asia	13	0	**81**	4	0	1
Europe	12	3	1	**82**	1	1
Africa	19	4	4	17	**56**	0
Oceania	25	2	10	7	1	**55**
Precision	0.56	0.90	0.82	0.69	0.97	0.95
Recall	0.93	0.90	0.81	0.82	0.56	0.55

(e) Confusion Matrix with Precision and Recall



(f) Gender Accuracy

Figure 5: Gender and Location Labels, and propagation results

The number of explicit friend links in Xanga is quite small which explains the decrease in the prediction accuracy when considering only friends (Blogger does not define friend links explicitly, so is not included here). We observe that including both the friend and blog edges often *reduces* the accuracy in all cases, showing the importance of different edge weights in the multigraph case.

Multigraph Nodes. We next analyzed the multigraph composed of both blog and web nodes, linked by blog and web links. There was no significant difference (less than 2%) in the prediction accuracy for age of bloggers when web nodes were considered as shown in Figures 4(e) and Figure 4(f). However, including web nodes in the analysis greatly improves the connectivity of the graph: Figure 5(a) shows that including web links significantly reduces the number of connected components. Consequently a larger fraction of nodes are given labels (Figure 5(b)), with little change in overall accuracy.

A side-effect of the iterative labeling is to assign age labels to websites ("pseudo-ages"). For sites with highly targeted audiences, the age reflects the age of the audience: in all data sets Facebook, a social networking site frequented by college students, is assigned an age of 21, while its high-school only counterpart is assigned an age around 17. The assigned age also reflects the demographic of the blog network: the USA Today website is consistently assigned lower ages than the Washington Post, but these age pairs are higher in LiveJournal (28 vs. 25) than Xanga (23 vs. 17), which has younger users. The website for the band Slipknot is given age much lower (15) than that for Radiohead (28), reflecting the ages of their respective fans.

Location label. Figures 5(c) and 5(d) show the accuracy of predicting the country and continent of the blogger with local and global methods respectively while considering blog links and friend links. The local method predicts the country with about 80% accuracy for each dataset, significantly higher than the accuracy of predicting the correct age with no error. The reflects the fact that the hypothesis that connected blogs have similar locations (homophily) holds here. The global method performs less well, suggesting that the global hypothesis does not hold well for location.

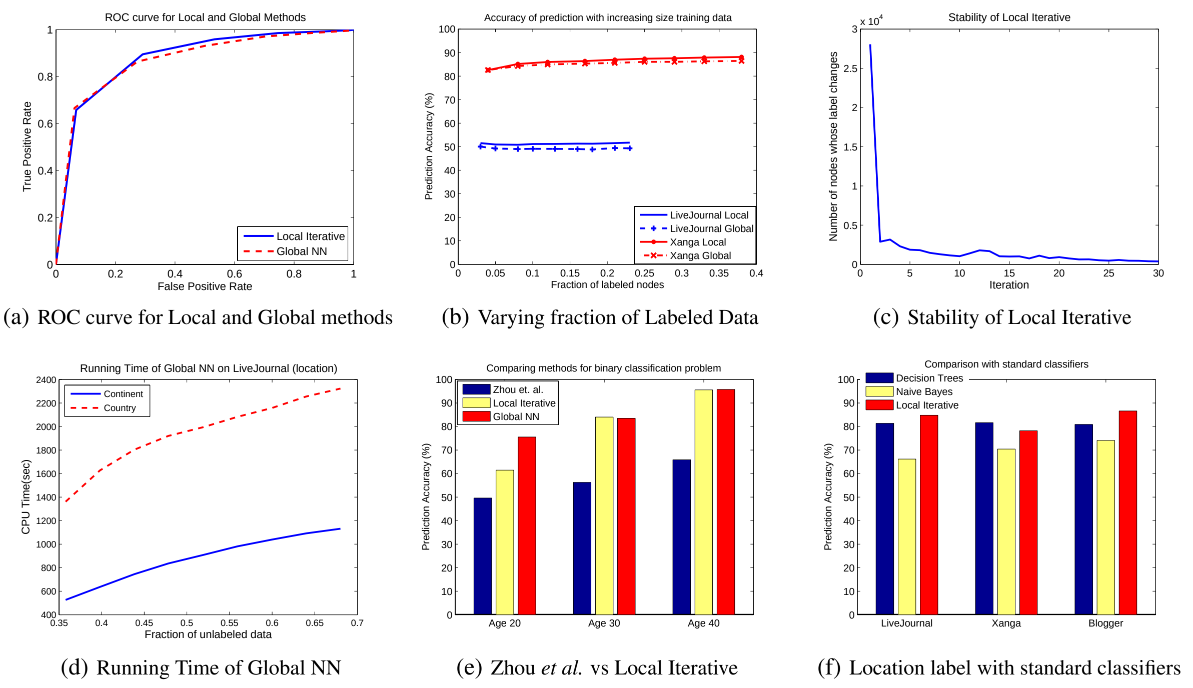
For inferring the continent of the blogger, the local algorithm's accuracy is 88% for LiveJournal and about 85% for Blogger and Xanga (Figure 5(c)). This task should be easier than predicting country, and indeed all methods improve by up to 10 percentage points. Drilling down, the confusion matrix (in percentage) presented in Table 5(e) helps evaluate the performance of the classifier on the Blogger dataset with the local method. Analysis of the precision and recall shows that the algorithm has a slight tendency to over represent the most common labels: North America (very common) has high recall but lower precision, while Africa (very rare) has high precision but lower recall.

Gender. As shown in Figure 5(f), our methods predict gender with up to 65% accuracy, with link information alone. This is better than random guessing, but not much, especially compared to methods that have looked at richer features (text content, word use, color schemes) [18]. Here, the local and global hypotheses do not hold well, and so more information is needed to improve the labeling.

4.3 Algorithm Performance Analysis

We compare the two methods by plotting the ROC curve for predicting the class of twenty-year old bloggers on the Xanga data set in Figure 6(a). The area under the curve (AUC) is 0.88 for Local Iterative and 0.87 for the Global NN method. Our methods give consistent accuracy as the fraction of labeled data for training is varied, even to below 1% of the total number of nodes (Figure 6(b)).

To study the impact of number of iterations for labeling, we studied the number of nodes whose age label *changes* during each iteration of the local method. The results for Blogger are plotted in Figure 6(c). There is a sharp decline in the first iteration, since many unlabeled nodes gain labels from their neighbors. The labeling quickly stabilizes, and although labels continue to shift, they do not impact the accuracy. We determined that just five iterations



(a) ROC curve for Local and Global methods

(b) Varying fraction of Labeled Data

(c) Stability of Local Iterative

(d) Running Time of Global NN

(e) Zhou *et al.* vs Local Iterative

(f) Location label with standard classifiers

Figure 6: Accuracy and Comparison with other methods

sufficed for a good quality labeling, and the accuracy was not very sensitive to this choice.

Recall that the running time of the global method depends on length of the feature vector, number of edges, labeled and unlabeled nodes from Theorem 2.2. Figure 6(d) shows the running time of the global method for varying feature vectors lengths and number of nodes and edges. The iterative algorithms (not shown) were significantly faster, typically taking around 10 seconds of CPU time instead of tens of minutes for the global case. The feature vector for countries is longer than that for continents, so continent labeling is faster than country labeling. Experiments on age (not shown) took longer, since the numbers of labeled and unlabeled nodes were approximately equal, the worst case for our algorithm. From the results of Figure 6(b), we could improve running time by using only a subset of the labeled nodes: that is, only compare to a (random) sample of the training data instead of every example.

4.4 Comparison with Previous Work

In prior work, Zhou *et al.* demonstrate a method for learning binary labels on graph data [24]. We implemented the method in Matlab using sparse matrix representation, with the same parameter values used in [24]. For comparison, we considered binary age labeling problems. The target label is whether the age is below a threshold of 20, 30, or 40 years. Figure 6(e) shows the result of ten-fold cross-validation on LiveJournal data: there is a benefit to using our local and global algorithms. The results were similar for the other datasets and labels.

We also compared with a large class of methods based on applying standard machine learning methods as a 'black box' over the feature vectors described above. This approach is at the core of prior work on relational learning and link classification [16, 10]. We tested methods drawn from the Weka [20] library, and show only the *best* of these on location classification in Figure 6(f). Since the data is skewed, the Naïve Bayes classifier trivially assigns all test samples to one or two popular classes. The performance of Decision Trees approaches that of the local method. Analysis of these resulting trees shows that they seem to encode rules similar to the ones embodied by the local iterative algorithm: the learned trees rule out locations with frequency count zero, then use a series of thresholds to (approximately) identify the most common adjacent label. This strengthens our confidence in the local hypothesis for this data. Note that Decision Trees do not scale well to a large number of classes, and were significantly slower to learn and label than the local iterative approach.

5. RELATED WORK

Our work is most closely related to problems of classification of relational data [7, 19, 16, 6]. The area of Relational Learning is concerned with classification of objects that can be represented as relational databases. Our problem of graph labeling fits into this framework, since a (multi)graph can be encoded as tables of nodes and edges (one table for each node type, and one for each link type). Getoor *et al.* [7] proposed Probabilistic Relational Models (PRMs) which induce a Bayesian network over the given relational data to encode dependencies between each entity type and its attributes. This generative model is also limited by the constraint that the induced Bayesian network must be acyclic, which is typically not the case in blog/web networks: our graphs have cycles. To overcome this constraint, Taskar *et al.* [19, 10] introduced Relational Markov Networks (RMN), a discriminative model based on Markov networks. Generalized Belief Propagation [21] is used for propagation of labels in the network. RMNs are described in the context of graphs with only one node type, instead of multigraphs, and rely on undirected edges—a limitation in settings where directionality provides information (as in web links). Neville *et al.* propose a decision tree variant Relational Probability Trees (RPT) [16], which incorporates aggregates over the neighborhood of a node (number of neighbors, average or mode of an attribute value, etc.).

Prior work often operates in the traditional supervised classification mode, with the assumption of a fully labeled graph to train on. This is not practical in our setting, especially where we observe a large graph of which only a subset of nodes are labeled. The induced fully labeled subgraph is typically so sparse that it is not representive, and may have few remaining links. Instead, one needs semi-supervised learning on graphs. Semi-supervised methods use the combination of labeled and unlabeled examples [25, 26], viewed as a graph. Zhou *et al.* use graph regularization to impose a smoothness condition on the graph for labeling [24]. The proposed method is defined for a binary classification problem, and it not easily extensible to the general multi-class case. Earlier work [23] addresses the multi-class labeling problem, however does not consider the underlying link-structure in the data and assumes a distance-based affinity matrix. More recently, Zhang *et al.* [22] studied graph regularization for web-page categorization.

Our problem of graph labeling lies at the intersection of relational learning and semi-supervised learning. Here, prior approaches involve preprocessing the data such that it can be used as input to a known machine learning methods like random forests [2], or logistic regression [10]. This achieved by transforming the graph features into object attributes, and summarizing a multiset of neighboring nodes and their attributes by aggregate functions such as mean, mode, and count. Neville and Jensen [15] introduced the notion of an 'iterative classifier' over simple graphs (graphs with only one node type), which allows the result of one iteration of classification to influence the features used by the next round, for a fixed number of rounds. Similar to our local method, Macskassy *et al.* [12] proposed a simple Relational Neighbor (RN) classifier. The authors report comparable empirical performance of RN classifier and more sophisticated methods like PRM and RPT. Our graph labeling problem hearkens back to work of Chakrabarti *et al.* [5]. They observe that topic classification of webpages based on text can be improved by including information about the class of neighbors. Our aim is to go further and perform the classification based *only* on the neighborhood information from the link structure.

Analysis of blogs and other social media have been the focus of much research in the recent years. For classification in particular, prior work has studied blogs with respect to political orientation [1], mood [14], and so on. Non-link aware techniques like Natural Language Processing have been used for this [17]. There has also been some initial work on predicting the age and gender [18] and [4] of blog authors using textual features. These papers showed the quality of certain textual features for producing similar labels to those we study here. Here, we show that links alone can be very powerful. More recently, the study by MacKinnon *et al.* [11] determined the probability distribution of friends in LiveJournal to infer location and age. The work of Hu *et al.* [8] uses a Bayesian framework to model web-click data for predicting age and gender; a side effect is to assign demographic attributes to the web pages themselves. Note that click data is used, not link data, which means it applies to very different settings to those we consider.

6. CONCLUDING REMARKS

We have formalized the graph labeling problem for classification on blogs, and studied two classes of algorithms. These algorithms scale to large graphs, with hundreds of thousands of nodes and edges in a matter or minutes or seconds. On a case study with blog data, we see accuracy of up to 80-90% for correctly assigning labels, based only on link information. These results hold with a training set of 1% or even less compared to the size of the whole graph, and training data extracted automatically from profiles. It remains to validate these result on other domains, and to understand better how incorporating additional features can improve the results of these methods.

7. REFERENCES

[1] L. A. Adamic and N. Glance. The political blogosphere and the 2004 U.S. election: divided they blog. In *Intl. Workshop on Link Discovery (LinkKDD)*, pages 36–43, 2005.

[2] A.Van Assche, C. Vens, H. Blockeel, and S. Džeroski. A random forest approach to relational learning. In *Workshop on Statistical Relational Learning*, 2004.

[3] S. Bhagat, G. Cormode, S. Muthukrishnan, I. Rozenbaum, and H. Xue. No blog is an island - analyzing connections across information networks. In *Intl. Conference on Weblogs and Social Media*, 2007.

[4] J. D. Burger and J. C. Henderson. Barely legal writers: An exploration of features for predicting blogger age. In *AAAI Spring Symposium on Computational Approaches to Analyzing Weblogs*, 2006.

[5] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *ACM SIGMOD*, 1998.

[6] P. Domingos and M. Richardson. Markov logic: A unifying framework for statistical relational learning. In *Workshop on Statistical Relational Learning*, 2004.

[7] L. Getoor, N. Friedman, D. Koller, and B. Taskar. Learning probabilistic models of link structure. *Journal of Machine Learning Research*, 3:679–707, 2002.

[8] J. Hu, H.-J. Zeng, H. Li, C. Niu, and Z. Chen. Demographic prediction based on user's browsing behavior. In *Intl. World Wide Web Conference*, 2007.

[9] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *STOC*, 1998.

[10] Q. Lu and L. Getoor. Link-based classification. In *Intl. Conference on Machine Learning*, 2003.

[11] I. MacKinnon and R. H. Warren. Age and geographic inferences of the LiveJournal social network. In *Statistical Network Analysis Workshop*, 2006.

[12] S. A. Macskassy and F. Provost. A simple relational classifier. In *Workshop on Multi-Relational Data Mining*, 2003.

[13] M. McPherson, L. Smith-Lovin, and J. M. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27:415–444, 2001.

[14] G. Mishne. Experiments with mood classification in blog posts. In *Workshop on Stylistic Analysis of Text for Information Access*, 2005.

[15] J. Neville and D. Jensen. Iterative Classification in Relational Data. In *Workshop on Learning Statistical Models from Relational Data*, 2000.

[16] J. Neville, D. Jensen, L. Friedland, and M. Hay. Learning relational probability trees. In *ACM Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2003.

[17] H. Qu, A. L. Pietra, and S. Poon. Classifying blogs using NLP: Challenges and pitfalls. In *AAAI Spring Symposium on Computational Approaches to Analyzing Weblogs*, 2006.

[18] J. Schler, M. Koppel, S. Argamon, and J. Pennebaker. Effects of age and gender on blogging. In *AAAI Spring Symposium on Computational Approaches to Analyzing Weblogs*, 2006.

[19] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Conference on Uncertainty in Artificial Intelligence*, 2002.

[20] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

[21] J. Yedidia, W. Freeman, and Y. Weiss. Generalized belief propagation. In *Neural Information Processing Systems*, 2000.

[22] T. Zhang, A. Popescul, and B. Dom. Linear prediction models with graph regularization for web-page categorization. In *ACM Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2006.

[23] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Neural Information Processing Systems*, 2004.

[24] D. Zhou, J. Huang, and B. Schölkopf. Learning from labeled and unlabeled data on a directed graph. In *Intl. Conference on Machine Learning*, pages 1041–1048, 2005.

[25] X. Zhu. Semi-supervised learning literature survey. Technical report, Computer Sciences, University of Wisconsin-Madison, 2006.

[26] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *Intl. Conference on Machine Learning*, 2003.