

Deduplication and Group Detection using Links

Indrajit Bhattacharya, Lise Getoor
Department of Computer Science
University of Maryland
College Park, MD 20742, USA
{indrajit,getoor}@cs.umd.edu

ABSTRACT

Clustering is a fundamental problem in data mining. Traditionally, clustering is done based on the similarity of the attribute values of the entities to be clustered. More recently, there has been greater interest in clustering relational and structured data. Often times this data is best described as a graph, in which there are both entities, described by a collection of attributes, and links between entities, representing the relations between them. Clustering in these scenarios becomes more complex, as we should also take into account the similarity of the entity links when we are clustering. We propose novel distance measures for clustering linked data, and show how they can be used to solve two important data mining tasks, entity deduplication and group discovery.

Categories and Subject Descriptors

H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval—*clustering*; H.2.8 [Database Management]: Database Applications—*Data Mining*

General Terms

algorithms, performance

Keywords

deduplication, group detection, distance measure, clustering

1. INTRODUCTION

Recently there has been a surge of interest in mining relational and semi-structured data [1, 27, 26, 2, 3, 17]. There has been a great deal of work on link-based classification [8, 39, 42, 34, 28], and less work on link-based clustering [47, 33, 29]. Here, we propose novel distance measures for link-based clustering algorithms. We show how they can be used to solve two important data mining tasks: entity deduplication and group detection.

In data cleaning, deduplication [23, 37] is important for both accurate analysis, for example determining the number

of customers, and for cost-effectiveness, for example removing duplicates from direct mailing list. In information integration, determining approximate joins [10] is important for consolidating information from multiple sources; most often there will not be a unique key that can be used to join tables in distributed databases, and we must infer when two records from different databases, possibly with different structures, refer to the same entity. Traditional approaches to duplicate detection are based on approximate string matching criteria, in some cases augmented with domain specific rules. More recently, there have been adaptive approaches which make use of multiple attributes and use labeled data [46, 6, 4, 9].

The problem of identifying groups of similar entities in the presence of linked environments has been studied in the areas of social sciences, bibliometry, criminal intelligence, hypertext connectivity and document information retrieval. But deduplication and group detection have mostly been dealt with separately and as unrelated problems. We argue that the two problems occur together for most real world applications and this calls for a unified framework for addressing them. For instance, Goldberg and Senator [22] discuss the importance of ‘consolidation’ and ‘linking’ in the context of large real databases, highlighting the related nature of the two problems.

Not only do the two problems co-occur, we observe that the issue of measuring distances between sets of links comes up for both tasks when working with link data. We propose different distance measures for sets of links tailored for the two problems.

The traditional approach to deduplication is to consider distance between the observed attributes of the entities. We augment this approach to consider distances between the entity relations or links. In an earlier workshop paper [5], we have proposed two different measures of link distance for deduplication. The first is the link detail distance that considers all the information in the relations involving that observation. This is useful but computationally expensive, so we propose as a more tractable alternative the link summary distance. Our experiments show that when the distance measure is augmented to consider either of the link distances, it leads to significant improvements in performance.

The second task is detecting groups of similar entities from link data. This a problem that is relevant in many different domains. As an example, we may be looking to group authors by their research interests from the author lists of academic papers in a database. Here the names of the authors of a particular paper would constitute a link. Considering

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

LinkKDD 2004, August 22, 2004, Seattle, Washington, USA
Copyright 2004 ACM 1-58113-888-1/04/0008 ...\$5.00.

real-world scenarios where such links are observed, we argue that the entities in each link are very likely to come from the same group of entities. When this assumption holds for the observed link data, we can cluster the links to discover the hidden groups. Once we have the links clustered into groups, the mapping of entities to groups follows naturally from it.

Since we focus on clustering the links, a distance measure between sets of links again becomes relevant. We propose a distance measure for this task based on a clearly defined generative probabilistic model. It measures the change in probability of the observed data with regard to the generative model that occurs on merging two clusters. This we call the probabilistic distance between two link sets. We also motivate the use of the link summary distance as an efficient approximation of the probabilistic distance. We evaluate and compare the two distance measures for the task of group detection over varying data characteristics.

We show that link-based clustering can be used to solve both the tasks of entity deduplication and group detection. The two tasks are quite related and any framework for group detection from links arising in real-world applications must also account for uncertainty in the identity of the entities themselves. We propose a single generative model for the two tasks and use it as motivation for our link-based clustering algorithm.

2. RELATED WORK

Deduplication: There has been a large body of work on deduplication, record linkage, and co-reference detection.¹ Here we review some of the main work, but the review is not exhaustive. For summary reports on deduplication and record linkage, see [50].

Within the statistics community, the earliest work was done by Newcombe [40], who describes methods for limiting the number of comparisons required. Fellegi and Sunter [19] define a statistical framework for record linkage, classifying pairs as a “match” or a “non-match”, based on certain computed features of the pairs. They describe how to estimate the parameters of their model and use it for classification. More recently, Winkler [49] builds upon the work by Fellegi and Sunter and uses a probabilistic approach with a latent match variable which is estimated using the EM algorithm.

There has been extensive work on defining approximate string matching algorithms [37, 38, 12] and adaptive algorithms that learn string similarity measures [45, 6, 13] and use active learning [46]. ‘Hardening soft databases’ is known to be a difficult problem [11] and an important focus is on efficient data cleaning; examples include Hernandez and Stolfo [23] and Monge and Elkan [37]. Another related area deals with identity uncertainty [44], object identification [48] and co-reference resolution in natural language processing. Co-reference resolution [36] is typically done with unstructured text; here our focus is on structured data.

One of the domains commonly used as a testbed is the bibliographic citation domain, which we use as motivation. However as we will see, we focus on deduplicating authors rather than papers. The work most closely related to our

¹The term deduplication is used more commonly within the database community, record linkage is the term used by statisticians and co-reference resolution is the term used more commonly by the AI and natural language processing community.

deduplication approach is that of Chaudhuri et al. [9, 4]. They also make use of join information to aid in deduplication; a key difference is that they assume that the secondary tables are themselves duplicate-free. Ananthakrishna et al. [4] use co-occurrence in dimensional hierarchies to identify duplicates. Specifically, to resolve whether two entities are duplicates, they check for co-occurrence in the children sets of the entities. We work in a more general setting where there is no hierarchy within the links/tuples and we use the link in its entirety to check for co-occurrences. As a result, instead of the easier problem of having to match sets, we are faced with the problem of matching sets of sets. Also, we use an iterative framework for our algorithm, motivated by our recursive definition of duplicates.

Group Detection: Work on detecting related groups of entities from link information also spreads over many different areas. Extensive work has been done in the area of document information retrieval, where the task is to cluster documents with regard to content. Each document is observed as a bag of words, and documents having similar distributions over words are considered similar. Spectral clustering approaches [41] aim to identify lower dimensional sub-spaces (eigen vectors) from the term document matrix and cluster documents with similar components along these sub-spaces. Another approach involves probabilistic modeling of the semantic space of documents with latent variables [24, 7]. Each document is modeled as a mixture of ‘topics’ or meaningful word distributions. It is not clear if complex ‘group mixture’ models are required for explaining co-authorship relations in bibliographic domains, but these are directions for future research.

Equally important and interesting is the domain of the internet and hypertext connectivity. Most of the research in this area is driven by the pioneering ranking algorithms of Page and Brin [43] and Kleinberg [30]. The basic idea is again to identify the principal eigen vectors for different forms of the connectivity matrix of all pages and this is done with an iterative algorithm. The idea of hubs and authorities has been extended to identify web communities [20]. The concept has also been augmented to have a probabilistic interpretation [14] and combined with document topic models to propose a unified model for documents and hypertexts [15]. The hypertext domain is inherently directional, which distinguishes it from our motivation. Also, our focus is to use the links for identifying similarities among entities and not ranking them with regard to importance or relevance.

Several researchers have made use of entity relationships for iteratively classifying or categorizing entities [39, 42, 34]. Taskar et al [47] have extended the framework of probabilistic relational models to propose a joint model for classification and clustering for relational data.

Recently, work has been done in probabilistic modeling of stochastic links given groups of entities [33, 31, 32]. Our generative model for links is most similar to that of Kubica et al [33]. Their model however does not consider uncertainty in the identity of entities. Also, while they look to learn the generative model directly using stochastic gradient ascent techniques, ours is a clustering approach that uses the probability of the observed links given the generative model to define a distance measure between clusters. This is similar in flavor to information theoretic distance measures that have previously been used for clustering [16].

3. DEDUPLICATION: PAPER AND AUTHOR RESOLUTION

Consider the problem of trying to construct a database of papers, authors and citations, from a collection of paper references, perhaps collected by crawling the web. A well-known example of such a system is CiteSeer [21], an autonomous citation indexing engine. CiteSeer is an important resource for CS researchers, and makes searching for electronic versions of papers easier. However, as anyone who has used CiteSeer can attest, there are often multiple references to the same paper; citations are not always resolved and authors are not always correctly identified [46, 44].

In the context of our motivating example, there are several potential references that must be resolved. The first is the paper resolution problem; this is the most commonly studied bibliographic deduplication task. Sometimes, the paper resolution can be done based simply on the title. We can use one of the many existing methods for string matching, perhaps even tuned to the task of title matching. There is additional relational information, in terms of the venue, the authors of the paper, and the citations made by the paper; this additional information may help add evidence to the fact that two references are the same. This type of entity resolution has been the focus of much of the work in citation matching [25, 35, 44, 46].

A more novel example of deduplication is the case of author resolution. Suppose that we have two different papers, and we are trying to determine if there are any authors in common between them. We can also do a string similarity match between the author names, but often references to the same person vary significantly. The most common difference is the variety of ways in which the first name and middle name are specified. For an author entity “Jeffrey David Ullman”, we may see references “J. D. Ullman”, “Jeff Ullman”, “Ullman, J. D.”, and so on. For the most part, these types of transformations can be handled by specialized code that checks for common name presentation transforms. However, we are still presented with the dilemma of determining whether a first name or middle name is the same as some initial; while the case of matching “J. D. Ullman” and “Jeffrey D. Ullman” seems quite obvious, for common names such as “J. Smith” and “X. Wang” the problem is more difficult. Existing systems take name frequency into account and will give unusual names higher matching scores. But this still leaves the problem of determining when two references to “J. Smith” refer to the same individual.

We propose to make use of additional context information in the form of coauthor relationships. If the coauthors of “J. Smith” for these two papers are the same, then we should take this into account, and give the two references a higher matching score. But in order to do this we must have already determined that the other two author references refer to the same individual; thus it becomes a chicken and egg problem.

Consider the example shown in Figure 1, where we have four paper references, each with a title and author references. Figure 2 shows the final result after all the author references have been correctly resolved. We begin by examining the author references to see which ones we consider to be the same. In the first step, we might decide that all of the Aho references refer to the same individual, because Aho is an unusual last name. This corresponds to identifying

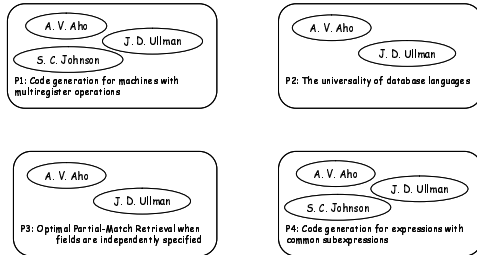


Figure 1: An example author/paper resolution problem. Each box represents a paper reference (in this case unique) and each oval represents an author reference.

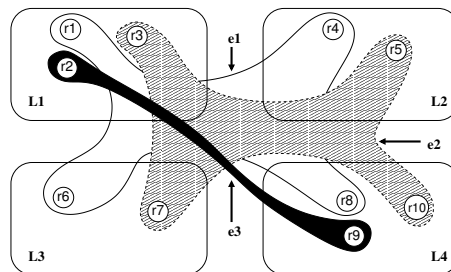


Figure 2: The deduplicated author entities corresponding to the example author/paper resolution problem.

r_1, r_4, r_6 and r_8 as duplicates. However, based on this name information alone, we are not quite sure whether the Ullman references (r_3, r_5, r_7 and r_{10}) are to the same individual, and we are certainly not sure about the Johnson references (r_2 and r_9). But, having decided that the Aho references are the same gives us additional information for the Ullman references. With high-confidence we consolidate the r_5 and r_7 Ullman references; we may also consolidate the other Ullman references, although we may not be as certain that this is correct. And, at this stage, based solely on the Aho entity consolidation, we may not have enough evidence to consolidate the Johnson references. However, after having made the Ullman consolidations, we may decide that having two common coauthors for the two references is enough evidence to tip the balance, and we decide that they refer to the same Johnson.

Thus the problem of author resolution is likely to be an iterative process; as we identify common authors, this will allow us to identify additional potential co-references. We can continue in this fashion until all of the entities have been resolved.

3.1 Deduplication of Entities: Formulation

In the deduplication problem, we have some collection of references to entities and from this set of references we would like to identify the (unique, minimal) collection of individ-

uals or entities to which they should be mapped. In other words, we would like to find a many-to-one mapping from references to entities. But apriori we do not know the set of entities.

Formally, we have a collection of references $R = \{r_1, r_2, \dots, r_n\}$. Each reference r corresponds to a unique entity, $E(r) \in \{e_1, e_2, \dots, e_k\}$ and conversely $R(e) = \{r_i | E(r_i) = e\}$. The references are members of links $L = \{l_1, l_2, \dots, l_m\}$. Each link is a collection of references and a reference appears in only one link. In our citation example, each observed author name is a reference, the list of author names for a paper form a link and the hidden entities are the true authors. Given R and L , our task is to correctly determine both the entities (including the number of entities k) and the mapping from references to entities.

Not surprisingly, we do this by clustering references that are similar to each other. The key to the success of this clustering algorithm is the similarity measure (or, equivalently, the distance measure). We define a distance measure that takes into account both the attributes of the entities and the links between them.

The attribute similarity of two references measures the similarity of the attributes of two references. For references r_i and r_j , it measures the similarity between two author names. If there are other attributes known, such as the author's institutions, then these can be factored in as well. There has been significant work on computing attribute similarity, several packages, such as [18], that implement this are available, so we assume that this measure is given.

In addition, to compute the similarity of the relationships that two entities participate in, we must compare their links. But if we simply compare whether the references are the same, then we will not find any overlap — references are derived from the authors and, as we defined them, each reference is distinct. We are looking for common authors in the links, but these are the entities that we do not know.

Instead, we compare two links by considering the references that are currently known/believed to be duplicates. So clearly, this notion of similarity is bound to the current set of duplicates and will change as more references are deduplicated. In the following section, we describe a distance measure that leverages this dynamic nature of the similarity.

3.2 Link Distances for Deduplication

Our aim is to construct the relation $dup(r_i, r_j)$ over the set of references given to us. This relation is symmetric and also reflexive, meaning that every reference is its own duplicate. Our definition of duplicates is based on a distance measure between references. We define

$$dup(r_i, r_j) = true \text{ if } d(r_i, r_j) < t$$

for a given threshold t and false otherwise. This distance measure is a weighted combination of the attribute distance of the references as well as distance between their links. We define the distance measure in the following subsection. The definition of duplicates is recursive in that the distance between references is tied to the current set of duplicates.

The distance between two references is defined as

$$d(r_i, r_j) = (1 - \alpha) \times d_{attr}(r_i, r_j) + \alpha \times d_{link}(L(r_i), L(r_j))$$

where $d_{attr}()$ is the distance between the attributes, $d_{link}()$

is the distance measure between *link sets* of references and α is a weighting of the two distances.

Now we define the link sets of references and the distance between them. For a reference r , $L(r)$ is all the links that r or its duplicates occur in. $L(r) = \{l | r' \in l \text{ and } dup(r, r')\}$. There may be many possible ways to define distance between sets of sets. We build the definition of distance between link sets on the basic notion of distance between two links of references. The similarity of two links is defined as the ratio of the number of duplicates they share and the length of the longer link. Formally, for two links l_1 and l_2 ,

$$sim(l_1, l_2) = \frac{|(r_1, r_2) | dup(r_1, r_2), r_1 \in l_1, r_2 \in l_2 |}{max(|l_1|, |l_2|)}$$

The distance is then $d(l_1, l_2) = 1 - sim(l_1, l_2)$. Now we take the distance of a link l from a link set L to be the shortest distance from l to some link l' in L . $d(l, L) = min_{l' \in L} d(l, l')$. Finally, for the distance between two link sets L_1 and L_2 , we find the mean distance of links in L_1 to L_2 and of links in L_2 to L_1 and take their average. This we choose to call the *link detail distance* between two references.

We can now see the recursion in the definition of duplicates and appreciate the need for an iterative algorithm. As new duplicates are discovered, the distances between link sets of references are going to change, potentially leading to the discovery of more duplicates. We represent the current sets of duplicates as clusters. We associate with each cluster the set of links that its references occur in. This is the link set of the cluster. Formally, for a cluster c_k , $L(c_k) = \{l(r_i) | r_i \in c_k\}$. Also, with each cluster, we maintain a representative attribute value of all its references.² Once we have these two features for a cluster, we can easily extend the definition of distance between references to the distance between clusters. At each step, the algorithm re-evaluates the distances between clusters and merges the 'nearest' cluster-pair to represent the same entity. The iterations continue until there are no more candidates worthy of merging.

We may imagine the link detail distance as comparing the co-authors of all papers that two observed authors have written to determine if they are the same author. It is computationally expensive since it involves pairwise comparison of members of the link sets in the two clusters. An approximate alternative that is computationally more tractable is to maintain and compare *link summaries*. The link summary l_{sum} of a reference is the set of all unique references in its link set. In terms of our example, it is the list of deduplicated collaborators for an observed author. Defining in terms of the cluster, it is the set of all cluster labels in the link set of the cluster. $l_{sum}(c_k) = \{c_i | c_i \in l_j, l_j \in L(c_k)\}$. Since the link summary is a set of cluster labels of references, our definition of distance between two links carries over to distance between link summaries. This we call the *link summary distance* $d_{l_{sum}}$ between two clusters. If the references in the summaries are kept sorted by their cluster labels, this distance is computable in time that is linear in the link summary lengths. Finally, we define the summary distance between two clusters c_i and c_j as a weighted combi-

²For numeric attribute values, we use the mean. For ordinals or string attributes, we can use domain specific knowledge to decide the representative value or use the mode or medoid.

nation of the distance d_{attr} of the representative attributes and the link summary distance d_{isum} .

$$d(c_i, c_j) = (1 - \alpha) \times d_{attr}(c_i, c_j) + \alpha \times d_{isum}(c_i, c_j)$$

4. GROUP DETECTION FROM LINKS

For most real world scenarios, the entities in a link are not random selections. It will be observed that there are distinct patterns in the observed relations and that a particular entity is more likely to co-occur in a link with a specific subset of all entities rather than a random one. This behavior or pattern can be better explained if we assume that the entities belong to groups or cliques and that the entities in any observed link come from the same group in most cases. We can allow each entity to belong to multiple groups at the same time. For the bibliographic domain, for example, we can imagine the entities in a group to be authors with a common research interest who collaborate. An author can have multiple research interests but all authors of any paper share a common interest. We would like to discover these groups given our link data.

4.1 Group Detection: Formulation

Any algorithm that directly clusters the entities must allow overlap among the clusters, so that an entity can belong to multiple groups. Most clustering algorithms do not have this feature. However, we can make use of the fact that the entities in a link come from the same group. Instead of clustering the entities, we can cluster the links, so that links that come from the same group are clustered together. Here we are looking for a partition of the links into clusters and any clustering algorithm with an appropriately defined distance measure should work. Each link cluster will now represent a group that we are interested in and an assignment of entities to groups follows naturally. Each cluster/group q_i has a set of links $L(q_i)$. The set of entities assigned to the group will be $E(q_i) = \{e | e \in L(q_i)\}$. Note that we are not constraining any entity to belong to a single group. An entity that occurs in two different links l_1 and l_2 that have different group labels q_1 and q_2 will be assigned to both groups q_1 and q_2 .

The key to the clustering will again be an appropriate distance measure. Each cluster or group is associated with its own set of links. Two clusters are similar if their link sets are similar. In the following subsection we propose link distances that are appropriate for the task of detecting groups of entities.

4.2 Link Distances for Group Detection

Our primary distance measure between clusters of links for the task of group detection is motivated by the unified generative model. It considers the change in probability of the observed data given the model that would occur if the two clusters were merged into one. The model structure M includes the set of groups Q and the set of entities E . The model parameters include the priors $P(q)$ over the groups q and the conditional probabilities $P(e|q)$ for an entity e belonging to a group q . The model generates each link by first selecting a group according to the priors and then sampling entities from the group according to the conditional probabilities. The joint log-probability of the observed link data

$L = \{l_i\}$ given a model M can then be written as

$$\begin{aligned} \log P(L|M) &= \sum_{l \in L} \log P(l|M) \\ &= \sum_l \sum_{q \in Q} \log P(q) P(l|q, M) \\ &= \sum_l \sum_q \log P(q) + \sum_l \sum_q \log P(l|q, M) \end{aligned}$$

A link of size k is formed by sampling k distinct entities from a group. If the group has n elements this can be done in $\binom{n}{k}$ ways, assuming that all entities in a group are equally likely to get chosen. The probability of a link coming from a group can now be written as $P(l|q) = 1/\binom{|q|}{|l|}$.

As we merge two different groups q_1 and q_2 to a single group $q_{1,2}$, the model structure is changed. The new group will now own all the links from q_1 and q_2 and its entities will be the union of the entity sets $E(q_1)$ and $E(q_2)$. The likelihood of the observed data is different given the new model M' . However the change is local in that only the generation probabilities of the links belonging to q_1 or q_2 will be affected. If L_a represents the set of affected links, then the change in log-probability can be expressed as

$$\begin{aligned} d_{LP}(q_1, q_2) &= \sum_{l \in L_a} (\log P(q_1) + \log P(q_2) - \log P(q_{1,2})) \\ &\quad + (\log P(l|q_1) + \log P(l|q_2) - \log P(l|q_{1,2})) \end{aligned}$$

For the way that we have defined $P(l|q)$, the probability is lower when the group contains more entities and in general the log-probability will always be adversely affected by the merge. We can imagine this as the cost of the merge operation. If we further assume uniform priors over the groups for both M and M' , then we can use

$$d_{LP}(q_i, q_j) \approx \sum_{l \in L_a} \log P(l|q_i) + \log P(l|q_j) - \log P(l|q_{i,j})$$

for comparing different merge options, since the part of d_{LP} involving the priors over groups will be the same for all pairs of groups (q_i, q_j) . A lower d_{LP} will be preferred since, of the possible models that have the same number of groups, it is desirable to select the one that has better likelihood for generating the observed links. This is the first distance measure that we explore for clustering links into groups and we will call it the *log-probability distance* or the *LP distance* between link clusters.

Since we have defined the probability of a link of size k being generated from a group of size n as the probability of choosing a k -set from n items, the greater the number of entities in a group, the smaller will be the probability for a given link. In that light, when choosing two groups to merge, we are more inclined to select a pair that has a large number of entities in common. The larger the overlap in the entity set, the lower will be number of entities in the merged group. Consequently, the generation probabilities of the links from the two old clusters will be less adversely affected when the overlap is large. So, as an alternative to log-probability distance, we are tempted to define the distances between groups directly in terms of the overlap in their entity sets. But the entity sets are nothing but the link summaries of the groups and the distance measure that we want to use is the *link summary distance* between the groups! This is the second distance measure that we will evaluate for

the group detection task. We must remember that the link summary distance will again be an approximation. But it will be computationally cheaper since we are not explicitly calculating the generation probability of each link in a group, but instead comparing just the link summaries or entity sets.

5. CLUSTERING ALGORITHMS

We have now defined the distance measures to be used and evaluated for tasks of clustering the references into entities and the entities/links into groups. In the next two subsections, we elaborate the steps of the two clustering algorithms.

5.1 Iterative Deduplication

Recall that we use a combined distance measure for references that takes into account the attribute distance as well as the link distance of references. The link distance considers the current duplicates shared between two links and evolves as new duplicates are detected. Accordingly, we need an iterative algorithm. If each reference is in a different cluster at the beginning, the distance between all link summaries will be 1, since there are no known duplicates. Thus, in order to jump-start our clustering algorithm, we start off by merging together references that ‘obviously’ correspond to the same entity, or, in other words, are separated by negligible attribute distance. Once the initial clusters have been so formed, we choose the candidate cluster-pairs that are likely to be the same entity. The candidate set is chosen using a *distance threshold*. At each iterative step, the algorithm re-evaluates distances for the candidates, selects the closest pair according to the distance measure, merges the clusters and updates the representative attributes and link summaries. This procedure continues until the candidate set is exhausted.

5.2 Clustering for Group Detection

For the task of clustering links into groups, we select the candidate pairs of clusters whose distances are less than a threshold and merge them. We use the link summary distance for choosing the candidate pairs since it is cheaper to compute. For link summary clustering, we merge all candidate pairs. However for *LP* clustering, since we are choosing the candidates using a cheaper measure, we select them with a high link summary threshold, to make sure that we do not miss any potential candidates. We observe that *LP* distances of the closest pairs of clusters increase monotonically. So we continue merging pairs of groups from the liberally chosen candidate set till the distance between the closest pair goes above a selected *LP distance threshold*.

6. EXPERIMENTAL EVALUATION

A difficulty with evaluating record linkage performance is the lack of gold standard for real-world data sets. Here we report a systematic evaluation of our algorithms on synthetic data, where we can model associations among authors, quantify the amount of noise and evaluate the recall-precision profiles for our algorithms.

6.1 Data Generator

Since one of our goals is to evaluate the importance of co-occurrence information for deduplication, we use a unified data generator for both tasks that incorporates structure in

the author domain by mimicking a real-life scenario where authors affiliated with a research group or a department in a university co-author papers. It creates groups of entities, where an entity belongs to a group with some probability. Since any author can be associated with multiple research groups or have multiple research interests, we allow an entity to belong to multiple groups. Each entity has fixed attribute values, corresponding to the true identity of the author. However, when it appears as an author name in any paper, it is likely to look different, which is why author identification is difficult. We mimic this phenomenon by probabilistically adding noise to the author identity when generating the author information for any particular paper.

The parameters for the generator include the number of links, groups and authors, the degree of overlap between the groups and the mean size of the groups and the links. The degree of overlap controls the extent to which entities belong to multiple groups. Two other parameters control the noise in the attribute values of the references in each link — the error probability p_{err} and standard deviation σ_{err} .

Each link l in our dataset is generated independently. First, a preference group q is selected according to the prior probabilities of the groups. Then the number of authors for the link is chosen from a normal distribution. Each reference r is chosen by selecting an author a from the group q according their probabilities of belonging to q . An author is not repeated in a link. Each reference also has a small probability of being selected from any random group q' different from q . This is the data that we use to evaluate our group detection algorithm. For the purposes of entity identification from references, we add noise to the entity attributes. The attribute values of the reference r , as they appears in the link, are generated by modifying the attributes of a with probability p_{err} , the magnitude of modification being determined by σ_{err} . While this procedure does not exactly capture the actual generative process for author-names in a paper, we believe ours is a simple and reasonable model that considers relationships among authors.

6.2 Evaluation Measure

We evaluate our algorithm by measuring the quality of the clusters generated. We use two measures of cluster quality — *dispersion* and *diversity*. For the task of deduplication, *entity dispersion* reflects the number of different clusters that references corresponding to the same entity are spread over. Lower dispersion is better; a perfect deduplication has dispersion 1. *Cluster diversity* quantifies the number of distinct entities that have been put in the same cluster. Lower diversity is also preferable; a perfect clustering has diversity 1. For the task of group detection using clustering, we similarly measure *group dispersion* over clusters and the *cluster diversity* in terms of group labels of links included in the same cluster. For both tasks, there is an inherent tradeoff between improving diversity and dispersion; an improvement in one will usually adversely affect the other. We consider the weighted average of the dispersion over entities or groups and of the diversity over clusters as a measure of performance.

6.3 Deduplication Results

We compare the entity dispersion and cluster diversity achieved using link summary clustering against those using attribute clustering for varying data characteristics and al-

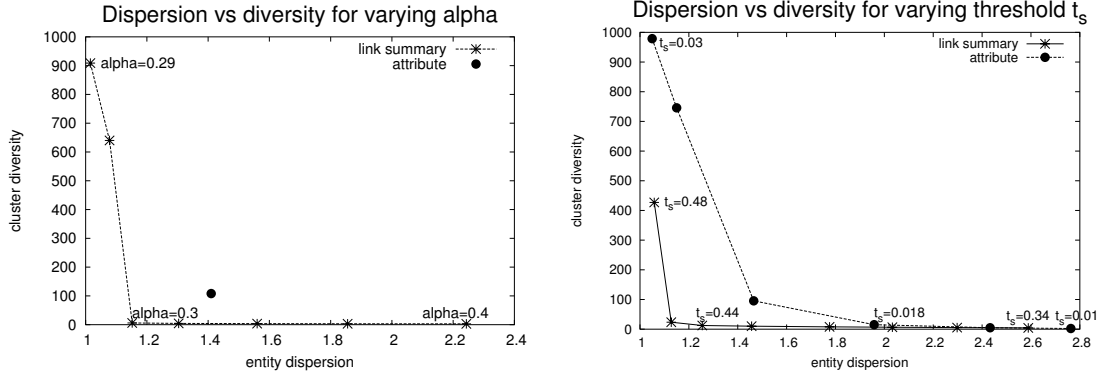


Figure 3: Deduplication: Dispersion-diversity plots for mixing weight and candidate selection threshold

gorithm parameters. The two algorithm parameters are the mixing weight α and the candidate selection threshold t_s . The data parameters that we experiment with are the error probability p_{err} and standard deviation σ_{err} , the mean group size g_{mean} and the mean link size l_{mean} . As default values, we choose 1000 entities, 5000 links with $l_{mean} = 4$, 100 groups with $g_{mean} = 10$ and minimal overlap between groups, error parameters $p_{err} = 0.1$ and $\sigma_{err} = 0.03$.

The dispersion-diversity plots in Figure 3 show how the performance of the two algorithms varies with α and t_s . Each point on the plots is for a different value of α or t_s . Naturally, a low threshold leads to more clusters and higher entity dispersion. As the threshold is increased, the number of clusters decreases leading to lower dispersion but higher cluster diversity. The α -plot for attribute clustering is a single point as the algorithm does not involve a mixing weight. We simply show the best result achievable by varying the threshold parameter. The plots show that the best (*dispersion, diversity*) combination achievable with link summary clustering is much better than attribute clustering.

We have extensively evaluated performance with varying data generator characteristics. We refer the reader to [5] for the detailed plots. The conclusions that we draw are that the gains with link clustering are more when the mean size of the links is larger, the groups of authors are more in number and smaller in size and the error probability and standard deviation are higher for the attribute values.

In Figure 4, we compare deduplication using link detail distance and link summary distance against that using attribute distance alone. We recall that link detail distance uses all the information in the links in a cluster for measuring distances while link summary distance is a computationally more tractable but approximate alternative that considers a summary representation of the links. While both of them do significantly better than attribute clustering, link detail clustering expectedly shows bigger improvements.

Figure 5 shows the execution times of link summary clustering and attribute clustering for increasing number of references. The plots show that link summary clustering scales gracefully with increasing data size. It is expectedly more costly than performing attribute similarity and for our datasets it takes roughly twice the time. However, the added cost

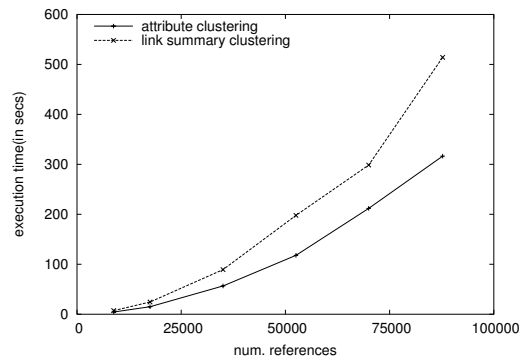


Figure 5: Deduplication: Execution times of attribute clustering and link summary clustering for varying data sizes

clearly reaps bigger benefits, as shown by the performance plots. Database cleaning is not an operation that is likely to be performed very frequently and the increased computation time is not expected to be too critical. There may of course be situations where this approach is not likely to prove advantageous, for example where distinctive groups do not exist for the entities or if references for each link appear randomly. There the user has the choice of falling back on traditional attribute similarity or choosing α to set a low weight for link distances.

6.4 Group Detection Results

Figure 6 shows the dispersion-diversity plots of our group detection task using the two distance measures — log-probability distance and link summary distance — as the data generator parameters are varied. Each curve shows the dispersion and diversity for different values of the threshold that determines termination of the algorithm — the candidate selection threshold for link summary distance and the merge distance threshold for log-probability distance. The parameters that we investigate are the size and number of groups for the same number of authors, the degree of overlap among

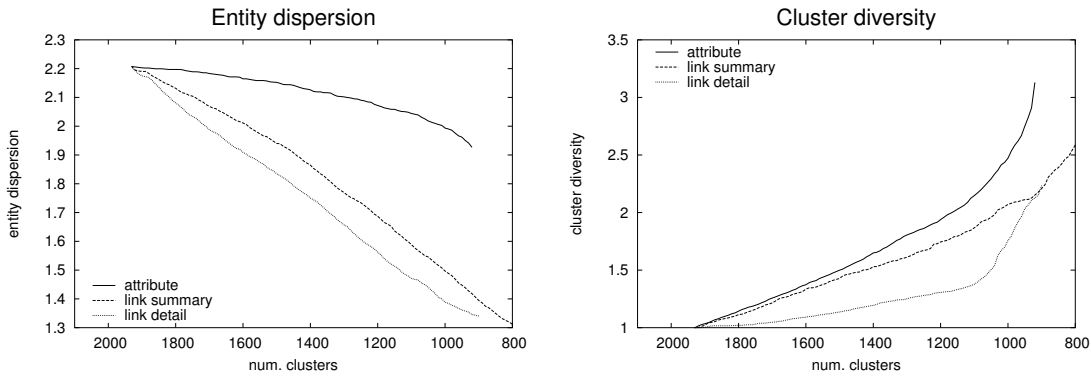


Figure 4: Deduplication: Performance comparison of attribute clustering, link detail clustering and link summary clustering

the groups in terms of the entities that they share and mean size of each link. We use as default 1000 entities, distributed over 20 groups with a mean size of 50 and reasonable overlap among groups, and a mean link size of 4. The plots show that log-probability distance generally performs better than link summary distance. However, the differences in performance shrink considerably as the size of the links increases and the entities are distributed over a larger number of groups thereby making each group smaller in size. The plots also expectedly show that groups are better identifiable when the links are larger, the groups are smaller and more in number and there is less overlap among the groups.

Figure 7 shows the execution times of link clustering using log-probability distance and summary distance for varying number of links in the dataset. For the comparisons, candidates are selected with the same threshold for both distance measures and the algorithms are terminated when the correct number of clusters (which is assumed to be known for this experiment) is reached. Note that the execution times shown are not averaged over multiple runs on datasets of the same size and are only intended to show the difference in running times for the two distance measures for the same clustering task. The conclusion is that while log-probability distance generally produces superior groups, it does so at the cost of significantly more execution time.

7. CONCLUSION

In this paper, we have proposed novel distance measures that take into account entity relationships for the purpose of clustering similar entities in linked environments. We show how these measures can be useful for the important data mining tasks of data deduplication and group detection. We argue that the two problems are quite related and use a unified generative model for link-data to evaluate our approaches. We present comparisons of the different distance measures for varying data characteristics that highlight the tradeoffs involved and results that show significant improvements over algorithms based just on entity attributes.

8. REFERENCES

[1] AAAI. *Workshop on Learning Statistical Models From Relational Data*, Menlo Park, CA, 2000.

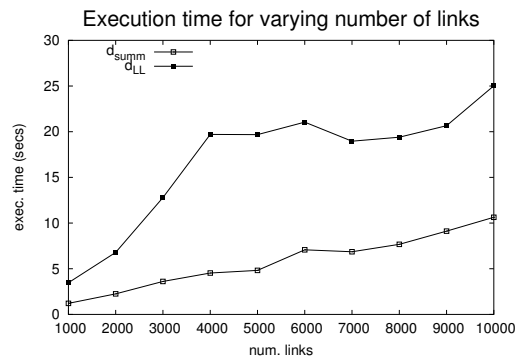


Figure 7: Group Detection: Comparison of execution times for link summary and log-probability distance measures

[2] ACM. *SIGKDD Workshop on Multi-Relational Data Mining*, Washington DC, August 2003.

[3] ACM. *SIGKDD Workshop on Multi-Relational Data Mining*, Seattle, USA, August 2004.

[4] R. Ananthakrishna, S. Chaudhuri, and V. Ganti. Eliminating fuzzy duplicates in data warehouses. In *Proceedings of the 28th International Conference on Very Large Databases (VLDB-2002)*, Hong Kong, China, 2002.

[5] I. Bhattacharya and L. Getoor. Iterative record linkage for cleaning and integration. In *Proceedings of the SIGMOD 2004 Workshop on Research Issues on Data Mining and Knowledge Discovery*, June 2004.

[6] M. Bilenko and R. J. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2003)*, Washington, DC, 2003.

[7] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:951–991, Jan 2003.

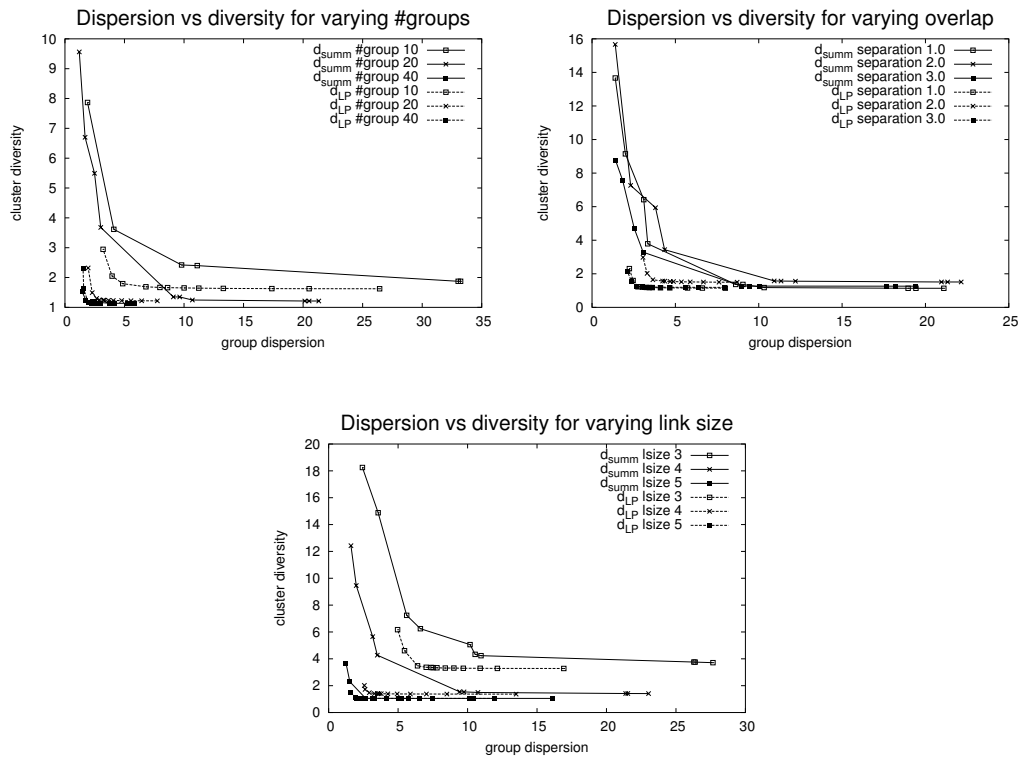


Figure 6: Group Detection: Dispersion diversity plots of link summary distance and log-probability distance for different data characteristics

[8] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 307–318. ACM Press, 1998.

[9] S. Chaudhuri, K. Ganjam, V. Ganti, and R. Motwani. Robust and efficient fuzzy match for online data cleaning. In *Proceedings of the 2003 ACM SIGMOD international conference on on Management of data*, pages 313–324, San Diego, CA, 2003.

[10] W. Cohen. Data integration using similarity joins and a word-based information representation language. *ACM Transactions on Information Systems*, 18:288–321, 2000.

[11] W. W. Cohen, H. Kautz, and D. McAllester. Hardening soft information sources. In *Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining (KDD-2000)*, pages 255–259, Boston, MA, August 2000.

[12] W. W. Cohen, P. Ravikumar, and S. E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *Proceedings of the IJCAI-2003 Workshop on Information Integration on the Web*, pages 73–78, Acapulco, Mexico, Aug. 2003.

[13] W. W. Cohen and J. Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2002)*, Edmonton, Alberta, 2002.

[14] D. Cohn and H. Chang. Learning to probabilistically identify authoritative documents. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 167–174. Morgan Kaufmann Publishers Inc., 2000.

[15] D. Cohn and T. Hofmann. The missing link - a probabilistic model of document content and hypertext connectivity. In *Neural Information Processing Systems 13*, 2001.

[16] I. S. Dhillon, S. Mallela, and R. Kumar. A divisive information theoretic feature clustering algorithm for text classification. *Journal of Machine Learning Research*, 3:1265–1287, 2003.

[17] S. Dzeroski and N. Lavrac. *Relational Data Mining*. Springer-Verlag New York, Inc., 2001.

[18] M. G. Elfeky, A. K. Elmagarmid, and V. S. Verykios. Tailor: A record linkage tool box. In *18th International Conference on Data Engineering (ICDE'02)*, 2002.

[19] I. P. Fellegi and A. B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64:1183–1210, 1969.

[20] D. Gibson, J. Kleinberg, and P. Raghavan. Inferring web communities from link topology. In *Proceedings of*

- the Ninth ACM Conference on Hypertext and Hypermedia : links, objects, time and space - structure in hypermedia systems, pages 225–234. ACM Press, 1998.
- [21] C. L. Giles, K. Bollacker, and S. Lawrence. CiteSeer: An automatic citation indexing system. In *Proceedings of the Third ACM Conference on Digital Libraries*, pages 89–98, Pittsburgh, PA, June 23–26 1998.
- [22] H. G. Goldberg and T. E. Senator. Restructuring databases for knowledge discovery by consolidation and link formation. In *Knowledge Discovery and Data Mining*, pages 136–141, 1995.
- [23] M. A. Hernández and S. J. Stolfo. The merge/purge problem for large databases. In *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data (SIGMOD-95)*, pages 127–138, San Jose, CA, May 1995.
- [24] T. Hofmann. Probabilistic latent semantic analysis. In *Proc. of Uncertainty in Artificial Intelligence, UAI'99*, Stockholm, 1999.
- [25] J. A. Hylton. Identifying and merging related bibliographic records. Master's thesis, Department of Electrical Engineering and Computer Science, MIT, 1996.
- [26] ICML. *Workshop on Learning Statistical Models from Relational Data*, Banff, Canada, July 2004.
- [27] IJCAI. *Workshop on Learning Statistical Models from Relational Data*, Acapulco, Mexico, August 2003.
- [28] D. Jensen. Statistical challenges to inductive inference in linked data. In *Seventh International Workshop on Artificial Intelligence and Statistics*, 1999.
- [29] I. Jonyer, D. J. Cook, and L. B. Holder. Graph-based hierarchical conceptual clustering. *Journal of Machine Learning Research*, 2:19–43, 2002.
- [30] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
- [31] J. Kubica, A. Moore, D. Cohn, and J. Schneider. Finding underlying connections: A fast graph-based method for link analysis and collaboration queries. In *Proceedings of the International Conference on Machine Learning*, August 2003.
- [32] J. Kubica, A. Moore, and J. Schneider. Tractable group detection on large link data sets. In *The Third IEEE International Conference on Data Mining*, pages 573–576. IEEE Computer Society, November 2003.
- [33] J. Kubica, A. Moore, J. Schneider, and Y. Yang. Stochastic link and group detection. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, pages 798–804. AAAI Press/MIT Press, July 2002.
- [34] Q. Lu and L. Getoor. Link-based classification. In *Proceedings of the International Conference On Machine Learning*, Washington DC, August 2003.
- [35] A. McCallum, K. Nigam, and L. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the Sixth International Conference On Knowledge Discovery and Data Mining (KDD-2000)*, pages 169–178, Boston, MA, Aug. 2000.
- [36] A. McCallum and B. Wellner. Toward conditional models of identity uncertainty with application to proper noun coreference. In *Proceedings of the IJCAI-2003 Workshop on Information Integration on the Web*, pages 79–86, Acapulco, Mexico, Aug. 2003.
- [37] A. E. Monge and C. P. Elkan. The field matching problem: Algorithms and applications. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 267–270, Portland, OR, August 1996.
- [38] G. Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, 2001.
- [39] J. Neville and D. Jensen. Iterative classification in relational data. In *Proc. AAAI-2000 Workshop on Learning Statistical Models from Relational Data*, pages 13–20. AAAI Press, 2000.
- [40] H. Newcombe, J. Kennedy, S. Axford, and A. James. Automatic linkage of vital records. *Science*, 130:954–959, 1959.
- [41] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, 2001.
- [42] H.-J. Oh, S. H. Myaeng, and M.-H. Lee. A practical hypertext categorization method using links and incrementally available class information. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 264–271. ACM Press, 2000.
- [43] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bring order to the web. Technical report, Stanford University, 1998.
- [44] H. Pasula, B. Marthi, B. Milch, S. Russell, and I. Shpitser. Identity uncertainty and citation matching. In *Advances in Neural Information Processing Systems 15*. MIT Press, 2003.
- [45] E. Ristad and P. Yianilos. Learning string edit distance. *IEEE Transactions on PAMI*, 20(5):522–532, 1998.
- [46] S. Sarawagi and A. Bhamidipaty. Interactive deduplication using active learning. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2002)*, Edmonton, Alberta, 2002.
- [47] B. Taskar, E. Segal, and D. Koller. Probabilistic classification and clustering in relational data. In *Proceeding of IJCAI-01, 17th International Joint Conference on Artificial Intelligence*, pages 870–878, Seattle, US, 2001.
- [48] S. Tejada, C. A. Knoblock, and S. Minton. Learning object identification rules for information integration. *Information Systems Journal*, 26(8):635–656, 2001.
- [49] W. E. Winkler. String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage. In *Proceedings of the Section on Survey Research Methods, American Statistical Association*, pages 354–359, 1990.
- [50] W. E. Winkler. The state of record linkage and current research problems. Technical report, Statistical Research Division, U.S. Census Bureau, Washington, DC, 1999.