Multi-Relational Record Linkage

Parag and Pedro Domingos

Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195, U.S.A.
{parag,pedrod}@cs.washington.edu
http://www.cs.washington.edu/homes/{parag,pedrod}

Abstract. Data cleaning and integration is typically the most expensive step in the KDD process. A key part, known as record linkage or de-duplication, is identifying which records in a database refer to the same entities. This problem is traditionally solved separately for each candidate record pair (followed by transitive closure). We propose to use instead a multi-relational approach, performing simultaneous inference for all candidate pairs, and allowing information to propagate from one candidate match to another via the attributes they have in common. Our formulation is based on conditional random fields, and allows an optimal solution to be found in polynomial time using a graph cut algorithm. Parameters are learned using a voted perceptron algorithm. Experiments on real and synthetic databases show that multi-relational record linkage outperforms the standard approach.

1 Introduction

Data cleaning and preparation is the first stage in the KDD process, and in most cases it is by far the most expensive. Data from relevant sources must be collected, integrated, scrubbed and pre-processed in a variety of ways before accurate models can be mined from it. When data from multiple databases is merged into a single relation, many duplicate records often result. These are records that, while not syntactically identical, represent the same real-world entity. Correctly merging these records and the information they represent is an essential step in producing data of sufficient quality for mining. This problem is known by the name of record linkage, de-duplication, merge/purge, object identification, identity uncertainty, hardening soft information sources, and others. In recent years it has received growing attention in the KDD community, with a related workshop at KDD-2003 and a related task as part of the 2003 KDD Cup.

Traditionally, the de-duplication problem has been solved by making an independent match decision for each candidate pair of records. A similarity score is calculated for each pair, and the pairs whose similarity score is above some

pre-determined threshold are merged. This is followed by taking a transitive closure over matching pairs. In this paper, we argue that there are several advantages to making the co-reference decisions together rather than considering each pair independently. In particular, we propose to introduce an explicit relation between each pair of records and each pair of attributes appearing in them, and use this to propagate information among co-reference decisions. To take an example, consider a bibliography database where each bibliography entry is represented by a title, a set of authors and a conference in which the paper appears. Now, determining that two bib-entries in which the conference strings are "KDD" and "Knowledge Discovery in Databases" refer to the same paper would lead to the inference that the two conference strings refer to the same underlying conference. This in turn might provide sufficient additional evidence to match two other bib-entries containing those strings. This new match would entail that the respective authors are the same, which in turn might trigger some other matches, and so on. Note that none of this would have been possible if we had considered the pair-wise decisions independently.

Our formulation of the problem is based on conditional random fields, which are undirected graphical models [9]. Conditional random fields are discriminative models, freeing us from the need to model dependencies in the evidence data. Our formulation of the problem allows us to perform optimal inference in polynomial time. This is done by converting the original graph into a network flow graph, such that the min-cut of the network flow graph corresponds to the optimal configuration of node labels in the original graph. The parameters of the model are learned using a voted perceptron algorithm [5]. Experiments on real and semi-artificial data sets show that our approach performs better than the standard approach of making pairwise decisions independently.

The organization of this paper is as follows. In Section 2, we describe the standard approach to record linkage. In Section 3, we describe in detail our proposed solution to the problem based on conditional random fields, which we call the *collective model*. Section 4 describes our experiments on real and semi-artificial data sets. Section 5 discusses related work. Finally, we conclude and give directions for future research in Section 6.

2 Standard Model

In this section, we describe the standard approach to record linkage [6]. Consider a database of records which we want to de-duplicate. Let each record be represented by a set of attributes. Consider a candidate pair decision, denoted by y, where y can take values from the set $\{1, -1\}$. A value of 1 means that the records in the pair refer to the same entity and a value of -1 means that the records in the pair refer to different entities. Let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ denote a vector of similarity scores between the attributes corresponding to the records in the candidate pair. Then, in the standard approach, the probability distribution of y given $\mathbf{x}$ is defined using a naive Bayes or logistic regression model:

$$f(\mathbf{x}) = \log \frac{P(y=1|\mathbf{x})}{P(y=-1|\mathbf{x})} = \lambda_0 + \sum_{i=1}^{n} \lambda_i x_i \quad (1)$$

$f(\mathbf{x})$ is known as the discriminant function. λ_i, for $0 \leq i \leq n$, are the parameters of the model. Given these parameters and the attribute similarity vector $\mathbf{x}$, a candidate pair decision y is predicted to be positive (a match) if $f(\mathbf{x}) > 0$ and predicted to be negative (non-match) otherwise. The parameters are usually set by maximum likelihood or maximum conditional likelihood. Gradient descent is used to find the parameters which maximize the conditional likelihood of y given $\mathbf{x}$, i.e., $P_\lambda(y|\mathbf{x})$ [1].

3 Collective Model

The basic difference between the standard model and the collective model is that the collective model does not make pairwise decisions independently. Rather, it makes a collective decision for all the candidate pairs, propagating information through shared attribute values, thereby making a more informed decision about the potential matches. Our model is based on conditional random fields as described in Lafferty et al. [9]. Before we describe the model, we will give a brief overview of conditional random fields.

3.1 Conditional Random Fields

Conditional random fields are undirected graphical models which define the conditional probability of a set of output variables Y given a set of input or evidence variables X. Formally,

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_\mathbf{x}} \prod_{c \in C} \phi_c(\mathbf{y_c}, \mathbf{x_c}), \quad (2)$$

where C is the set of cliques in the graph, and $\mathbf{y_c}$ and $\mathbf{x_c}$ denote the subset of variables participating in the clique c. ϕ_c, known as a clique potential, is a function of the variables involved in the clique c. Z_x is the normalization constant. Typically, ϕ_c is defined as a log-linear combination of features over c, i.e., $\phi_c(\mathbf{y_c}, \mathbf{x_c}) = \exp \sum_l \lambda_{lc} f_{lc}(\mathbf{y_c}, \mathbf{x_c})$, where f_{lc}, known as a feature function, is a function of variables involved in the clique c, and λ_{lc} are the feature weights.

In many domains, rather than having different parameters (feature weights) for each clique in the graph, the parameters of a conditional random field are tied across repeating clique patterns in the graph. Following the terminology of Taskar et al. [17], we call each such pattern a *relational clique template*. Each clique c matching a clique template t is called an instance of the template. The probability distribution can then be specified as

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_{\mathbf{x}}} \sum_{t \in T} \sum_{c \in C_t} \exp \sum_{l} \lambda_{lt} f_{lt}(\mathbf{y_c}, \mathbf{x_c}) \tag{3}$$

where T is the set of all the templates, C_t is the set of cliques which satisfy the template t, and f_{lt}, λ_{lt} are respectively the feature functions and feature weights pertaining to template t. Because of the parameter tying, the feature functions and the parameters vary over the clique templates and not the individual cliques. A conditional random field with parameter tying as defined above closely matches a relational Markov network as defined by Taskar et al. [17].

3.2 Notation

Before we delve into the model, let us introduce some notation. Consider a database relation $R = \{r_1, r_2, \ldots, r_n\}$, where r_i is the ith record in the relation. Let $A = \{A^1, A^2, \ldots, A^m\}$ denote the set of attributes. For each attribute A^k, we have a set AS^k of corresponding attribute values appearing in the relation, $AS^k = \{a_1^k, a_2^k, \ldots, a_{l_k}^k\}$. Now, the task at hand is to, given a pair of records (r_i, r_j) (and corresponding attribute values), find out if they refer to the same underlying entity. We will denote the kth attribute value of record r_i by $r_i.A^k$.

Our formulation of the problem is in terms of undirected graphical models. For the rest of the paper, we will use the following notation to denote node types, a specific instance of a node and the node values. A capital letter subscripted by a "$*$" will denote a node type, e.g. R_*. A capital letter with two subscripted letters will denote a specific instance of a node type, e.g., R_{ij}. A lower-case letter with two subscripts will denote a binary or continuous node value, e.g., r_{ij}.

3.3 Constructing the Graph

Given a database relation which we want to de-duplicate, we construct an undirected graph as follows. For each pairwise question of the form, "Is r_i the same as r_j?", we have a binary node R_{ij} in the graph. Because of the symmetric nature of the question, R_{ij} and R_{ji} represent the same node. We call these nodes record nodes. The record node type is denoted by R_*. For each record node, we have a corresponding set of continuous-valued nodes, called attribute nodes. The kth attribute node for record node R_{ij} is denoted by $R_{ij}.A^k$. The type of these nodes is denoted by A_*^k, for each attribute A^k. The value of the node $R_{ij}.A^k$ is the similarity score between the corresponding attribute values $r_i.A^k$ and $r_j.A^k$. For example, for textual attributes this could be the TF/IDF similarity score [15]. For numeric attributes, this could be the normalized difference between the two numerical values. Since the value of these nodes is known beforehand, we also call them evidence nodes. We interchangeably use the term evidence node and attribute node to refer to these nodes. We now introduce an edge between each R_* node and each of the the corresponding A_*^k nodes, i.e., an edge between each record node and the corresponding evidence nodes for each attribute. An edge

in the graph essentially means that values of the two nodes are dependent on each other. To take an example, consider a relation which contains bibliography entries for various papers. Let the attributes of the relation be author, title and venue. Figure 1(a) represents the graph corresponding to candidate pairs b_{12} and b_{23} for this relation, where b_{12} corresponds to asking the question "Is bib-entry b_1 same as bib-entry b_2?". b_{23} is similarly defined. $Sim(b_i.A, b_j.A)$ denotes the similarity score for the authors of the bibliography entries b_i and b_j for the various values of i and j. Similarly, $Sim(b_i.T, b_j.T)$ and $Sim(b_i.V, b_j.V)$ denote the similarity scores for title and venue attributes, respectively.

The graph corresponding to the full relation would have many such disconnected components, each component representing a candidate pair decision. The above construction essentially corresponds to the way candidate pair decisions are made in the standard approach, with no information sharing among the various decisions. Next, we describe how we change the representation to allow for the exchange of information between candidate pair decisions.

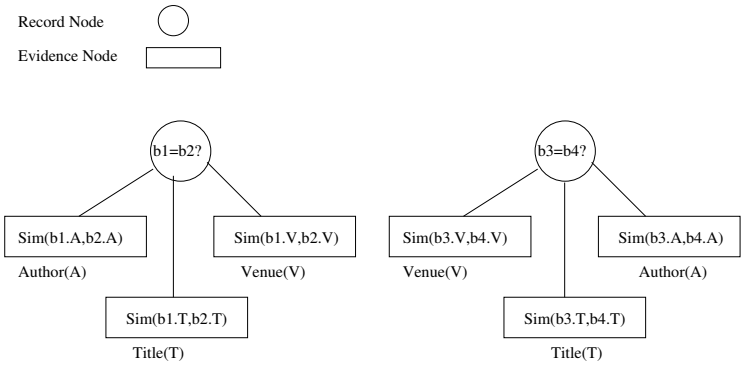
3.4 Merging the Evidence Nodes

We note the fact that the graph construction as described in the previous section, would in general have many duplicates among the evidence nodes. In other words, many record pairs would have the same attribute value pair. Using our notation, we say that nodes $R_{xy}.A^k$ and $R_{uv}.A^k$ are duplicates of each other if $(r_x.a^k = r_u.a^k \wedge r_y.a^k = r_v.a^k) \vee (r_x.a^k = r_v.a^k \wedge r_y.a^k = r_u.a^k)$. Our idea is to merge each such set of duplicates into a single node. Consider the bibliography example introduced in Section 3.3. Let us suppose that $(b_{12}.V, b_{34}.V)$ are the duplicate evidence pairs. Then, after merging the duplicate pairs, the graph would be as shown in Figure 1(b). Since we merge the duplicate pairs, instead of having a separate attribute node for each r_{ij} we now have an attribute node for each pair of values $a^k_{i'}, a^k_{j'} \in AS^k$, for each attribute A^k.

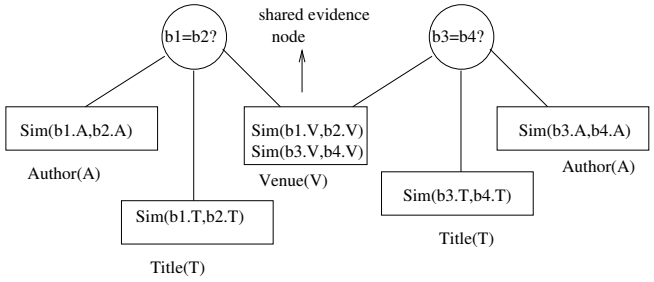
Although the formulation above helps to identify the places where information is shared between various candidate pairs, it does not facilitate any propagation of information. This is because the shared nodes are the evidence nodes and hence their values are fixed. The model as described above is thus no better than the decoupled model (where there is no sharing of evidence nodes) for the purpose of learning and inference. This sets the stage for the introduction of auxiliary nodes, which we also call information nodes. As the name suggests, these are the nodes which facilitate the exchange of information between candidate pairs.

3.5 Propagation of Information through Auxiliary Nodes

For each attribute pair node $A^k_{i'j'}$, we introduce a binary node $I^k_{i'j'}$. The node type is denoted by I^k_* and we call these information nodes. Semantically, an information node $I^k_{i'j'}$ corresponds to asking the question "Is $a^k_{i'}$ the same as $a^k_{j'}$?". The binary value of the information node $I^k_{i'j'}$ is denoted by $i^k_{i'j'}$, and



(a) Each pairwise decision considered independently



(b) Evidence nodes merged

Fig. 1. Merging the evidence nodes

is 1 *iff* the answer to the above question is "Yes" and -1 otherwise. While the attribute node $A^k_{i'j'}$ corresponds to the similarity score between the two attribute values as present in the database, the information node $I^k_{i'j'}$ corresponds to the Boolean-valued answer to the question of whether the two attribute values refer to the same underlying attribute. Each information node $I^k_{i'j'}$ is connected to the corresponding attribute node $A^k_{i'j'}$ and the corresponding record nodes R_{ij}. For instance, information node $I^k_{i'j'}$ would be connected to the record node R_{ij} *iff* $r_i.A^k = a^k_{i'}$ and $r_j.A^k = a^k_{j'}$. Note that the same information node $I^k_{i'j'}$ would in general be shared by several R_* nodes. This sharing lies at the heart of our model. Figure 2(a) shows how our hypothetical bibliography example is represented using the collective model.

Table 1. An example bibliography relation

Record	Title	Author	Venue
b_1	"Record Linkage using CRFs"	"Linda Stewart"	"KDD-2003"
b_2	"Record Linkage using CRFs"	"Linda Stewart"	"9th SIGKDD"
b_3	"Learning Boolean Formulas"	"Bill Johnson"	"KDD-2003"
b_4	"Learning of Boolean Expressions"	"William Johnson"	"9th SIGKDD"

3.6 An Example

Consider the subset of a bibliography relation shown in Table 1. Each bibliography entry is represented by three string attributes: title (T), author (A) and venue (V). Consider the corresponding undirected graph constructed as described in Section 3.5. We would have R_* nodes for pairwise binary decisions of the form "Does bib-entry b_i refer to the same paper as bib-entry b_j?", for each pair (i, j). Correspondingly, we would have evidence nodes for each pair of attribute values for each of the three attributes. We would also have I^k_* nodes for each attribute. For example, I^k_* nodes for the author attribute would correspond to the pairwise decisions of the form "Does the string a_i refer to same author as the string a_j?", where a_i and a_j are some author strings appearing in the database. Similarly, we would have I^k_* nodes for venue and title attributes. Each record node R_{ij} would have edges linking it to the corresponding author, title and venue information nodes, denoted by $I^k_{i'j'}$, where k varies over author, title and venue. In addition, each information node $I^k_{i'j'}$ would be connected to corresponding evidence node $A^k_{i'j'}$.

The corresponding graphical representation as described by the collective model is given by Figure 2(b). The figure shows only a part of the complete graph which is relevant to the following discussion. Note how dependencies flow through information nodes. To take an example, consider the bib-entry pair consisting of b_1 and b_2. The titles and authors for the two bib-entries are essentially the same string, giving sufficient evidence to infer that the two bib-entries refer to the same underlying paper. This in turn leads to the inference that the corresponding venue strings, "KDD-2003" and "9th SIGKDD", refer to the same venue. Now, since this venue pair is shared by the bib-entry pair (b_3, b_4), the additional piece of information that "KDD-2003" and "9th SIGKDD" refer to the same venue might give sufficient evidence to merge b_3 and b_4, when added to the fact that the corresponding title and author pairs have high similarity scores. This in turn would lead to the inference that the strings "William Johnson" and "Bill Johnson" refer to the same underlying author, which might start another chain of inferences somewhere else in the database.

Although the example above focused on a case when positive influence is propagated through attribute values, i.e., a match somewhere in the graph results in more matches, we can easily think of an example where negative influences are propagated through the attribute values, i.e., a non-match somewhere in the graph results in a chain of non-matches. In fact, our model is able to capture

complex interactions of positive and negative influences, resulting in an overall most likely configuration.

3.7 The Model and its Parameters

We have a singleton clique template for R_* nodes and another for I_* nodes. Also, we have a two-way clique template for an edge linking an R_* node to an I_*^k node. Additionally, we have a clique template for edges linking I_*^k and A_*^k nodes. Hence, the probability of a particular assignment $\mathbf{r}$ to the R_* and I_* nodes, given that the attribute (evidence) node values are $\mathbf{a}$, can be specified as

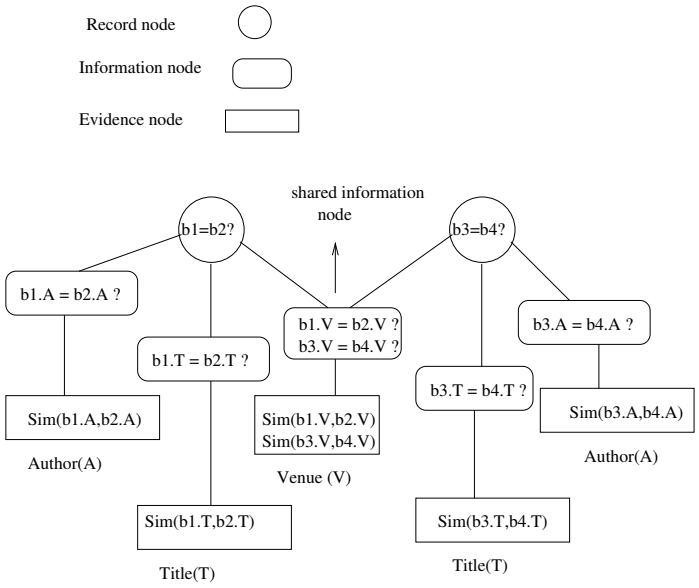
$$P(\mathbf{r}|\mathbf{a}) = \frac{1}{Z_{\mathbf{a}}} \exp \sum_{i,j} \left\{ \sum_l \lambda_l f_l(r_{ij}) + \sum_k \left[\sum_l \phi_{kl} f_l(r_{ij}.I^k) + \sum_l \gamma_{kl} g_l(r_{ij}, r_{ij}.I^k) + \sum_l \delta_{kl} h_l(r_{ij}.I^k, r_{ij}.A^k) \right] \right\} \quad (4)$$

where: (i, j) varies over all the candidate pairs; $r_{ij}.I^k$ denotes the binary value of the pairwise information node for the kth attribute pair corresponding to the node R_{ij}, and $r_{ij}.A^k$ denotes the corresponding evidence value; λ_l and ϕ_{kl} denote the feature weights for singleton cliques; γ_{kl} denotes the feature weights for two way cliques involving binary variables; and δ_{kl} denotes the feature weights for two way cliques involving evidence variables. For the singleton cliques and two-way cliques involving binary variables, we have a feature function for each possible configuration of the arguments, i.e., $f_l(x)$ is non-zero for $x = l$, $0 \leq l \leq 1$. Similarly, $g_l(x, y) = g_{ab}(x, y)$ is non-zero for $x = a, y = b$, $0 \leq a, b \leq 1$. For two-way cliques involving a binary variable r and a continuous variable e, we use two features: h_0 is non-zero for $r = 0$ and is defined as $h_0(r, e) = 1 - e$; similarly, h_1 is non-zero for $r = 1$ and is defined as $h_1(r, e) = e$.

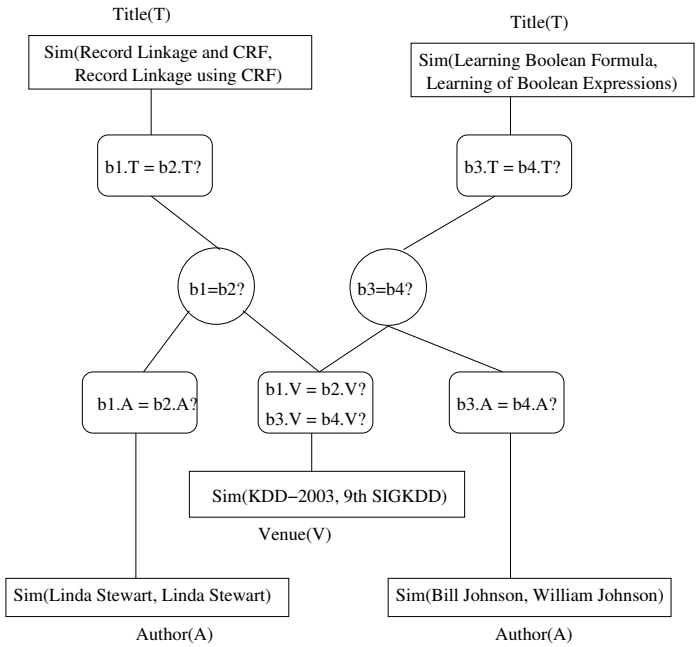
The way the collective model is constructed, a single information node in the graph would in general correspond to many record pairs. But semantically this single information node represents an aggregate of a number of nodes which have been merged together because they would always have the same value in our model. Therefore, for Equation 4 to be a correct model of the underlying graph, each information node (and the corresponding cliques with the evidence nodes) should be treated not as a single clique, but as an aggregate of cliques whose nodes always have the same values. Equation 4 takes this fact into account by summing the weighted features of the cliques for each candidate pair separately.

3.8 The Standard Model Revisited

If the information nodes are removed, and the corresponding edges are merged into direct edges between the R_* and A_*^k nodes, the probability distribution given by Equation 4 reduces to



(a) Complete representation



(b) A bibliography database example

Fig. 2. Collective model

$$P(\mathbf{r}|\mathbf{a}) = \frac{1}{Z_{\mathbf{a}}} \exp \sum_{i,j} \left[\sum_{l} \lambda_l f_l(r_{ij}) + \sum_{k} \sum_{l} \omega_{kl} h_l(r_{ij}, r_{ij}.A^k) \right] \tag{5}$$

where ω_{kl} denotes the feature weights for two-way variables. The remaining symbols are as described before. This formulation in terms of a conditional random field is very closely related to the standard model. Since in the absence of information nodes each pairwise decision is made independently of all others, we have $P(\mathbf{r}|\mathbf{a}) = \prod_{i,j} P(r_{ij}|\mathbf{a})$. When $\forall k$, $\omega_{k0} = \omega_{k1} = \omega_k$, for some ω_k, we have

$$\log \frac{P(r_{ij} = 1|\mathbf{a})}{P(r_{ij} = 0|\mathbf{a})} = \lambda' + \sum_{k} 2\omega_k r_{ij}.A^k \tag{6}$$

where $\lambda' = \lambda_1 - \lambda_0 - \omega_k$. This equation is in fact the standard model for making candidate pair decisions.

3.9 Inference

Inference corresponds to finding the configuration $\mathbf{r}^*$ such that $P(\mathbf{r}^*|\mathbf{a})$ given the learned parameters is maximized. For the case of conditional random fields where all non-evidence nodes and features are binary-valued and all cliques are singleton or two-way (as is our case), this problem can be reduced to a graph min-cut problem, provided certain constraints on the parameters are satisfied [7]. The idea is to map each node in the conditional random field to a corresponding node in a network-flow graph.

Consider a conditional random field with binary-valued nodes and having only one-way and two-way cliques. For the moment, we assume that there are no evidence variables. Further, we assume binary-valued feature functions $f(x)$ and $g(x, y)$ for singleton and two-way cliques respectively, as specified in the collective model. Then the expression for the log-likelihood of the probability distribution for assignment $\mathbf{y}$ to the nodes is given by

$$\begin{aligned} L(\mathbf{y}) = \sum_{i=1}^{n} [\lambda_{i_0}(1 - y_i) + \lambda_{i_1} y_i] + \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} [\gamma_{ij_{00}}(1 - y_i)(1 - y_j) \\ + \gamma_{ij_{01}}(1 - y_i) y_j + \gamma_{ij_{10}} y_i (1 - y_j) + \gamma_{ij_{11}} y_i y_j] + C \end{aligned} \tag{7}$$

where the first term varies over all the nodes in the graph taking the singleton cliques into account, and the second term varies over all the pairs of the nodes in the graph taking the two-way cliques into account. We assume the parameters for non-existent cliques to be zero. Now, ignoring the constant term and rearranging the terms, we obtain

$$-L(\mathbf{y}) = \sum_{i=1}^{n} -(\lambda_i y_i) + \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (\alpha_{ij} y_i + \beta_{ij} y_j - 2\gamma_{ij} y_i y_j) \tag{8}$$

where $\lambda_i = \lambda_{i_1} - \lambda_{i_0}$, $\gamma_{ij} = \frac{1}{2}(\gamma_{ij_{00}} + \gamma_{ij_{11}} - \gamma_{ij_{01}} - \gamma_{ij_{10}})$, $\alpha_{ij} = \gamma_{ij_{00}} - \gamma_{ij_{10}}$ and $\beta_{ij} = \gamma_{ij_{00}} - \gamma_{ij_{01}}$. Now, if $\gamma_{ij} \geq 0$ then the above equation can be rewritten as

$$-L(\mathbf{y}) = \sum_{i=1}^{n} -(\lambda_i^{'} y_i) + \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \gamma_{ij}(y_i - y_j)^2 \tag{9}$$

for some $\lambda_i^{'}, 1 \leq i \leq n$, given the fact that $y_i^2 = y_i$, since the y_i's are binary-valued.

Now, consider a capacitated network with $n+2$ nodes. For each node i in the original graph, we have a corresponding node in the network graph. Additionally, we have a source node (denoted by s) and a sink node (denoted by t). For each node i, there is a directed edge (s, i) of capacity $c_{si} = \lambda_i^{'}$ if $\lambda_i^{'} \geq 0$, else there a directed edge (i, t) of capacity $c_{it} = -\lambda_i^{'}$. Also, for each ordered pair (i, j), there is a directed edge of capacity $c_{ij} = \frac{1}{2}\gamma_{ij}$. For any partition of the network into sets B and W, $B = \{s\} \cup \{i : y_i = 1\}$ and $W = \{t\} \cup \{i : y_i = 0\}$, the capacity of the cut $C(\mathbf{y}) = \sum_{k \in B} \sum_{l \in W} c_{kl}$ is precisely the negative of the probability of the induced configuration on the original graph, offset by a constant. Hence, the partition induced by the min-cut corresponds to the most likely configuration in the original graph. The details can be found in Greig et al. [7]. We know that an exact solution to min-cut can be found in polynomial time. Hence, the exact inference in our model takes time polynomial in the size of the conditional random field.

It remains to see how to handle evidence nodes. This is straightforward. Notice that a clique involving an evidence node would account for an additional term of the form ωe in the log likelihood, where e is the value of the evidence node. Let y_i be the binary node in the clique. Since e is known beforehand, the above term can simply be taken into account by adding ωe to the singleton parameter $\lambda^{'}$ in Equation 9 corresponding to y_i.

3.10 Learning

Learning involves finding the maximum likelihood parameters (i.e., the parameters that maximize the probability of observing the training data). Instead of maximizing $P(\mathbf{r}|\mathbf{a})$, we maximize its logarithm (the log likelihood), using the standard approach of gradient descent. The partial derivative of the log-likelihood L given by Equation 4 with respect to the parameter λ_l is

$$\frac{\partial L}{\partial \lambda_l} = \sum_{i,j} f_l(r_{ij}) - \sum_{\mathbf{r}'} P_\Lambda(\mathbf{r}'|\mathbf{a}) \sum_{i,j} f_l(r'_{ij}) \tag{10}$$

where $\mathbf{r}'$ varies over all possible configurations of the nodes in the graph and $P_\Lambda(\mathbf{r}'|\mathbf{a})$ denotes the probability distribution with respect to current set of parameters. This expression has an intuitive meaning: it is the difference between the observed feature counts and the expected ones. The derivative with respect

to other parameters can be found in the same way. Notice that, for our inference to work, a constraint on the parameters of the two-way binary-valued cliques must be satisfied: $\gamma_{00} + \gamma_{11} - \gamma_{01} - \gamma_{10} \geq 0$. To ensure this, instead of learning the original parameters, we perform the following substitution on the parameters and learn the new parameters: $\gamma_{00} = g(\delta_1) + \delta_2$, $\gamma_{11} = g(\delta_1) - \delta_2$, $\gamma_{01} = -g(\delta_3) + \delta_4$, $\gamma_{10} = -g(\delta_3) - \delta_4$ where $g(x) = log(1 + e^x)$. It can be easily seen that, for any values of the parameters δ_i, the required constraint is satisfied on the original parameters. The derivative expression is modified appropriately for the substituted parameters.

The second term in the derivative expression involves the expected value over an exponential number of configurations. Hence finding this term would be intractable for any practical problem. Like McCallum and Wellner [11], we use a voted perceptron algorithm as proposed by Collins [5]. The expected value in the second term is approximated by the feature counts of the most likely configuration. The most likely configuration based on the current set of parameters can be found using our polynomial-time inference algorithm. At each iteration, the algorithm updates the parameters by the current gradient and then finds the gradient for the updated parameters. The final parameters are the average of the parameters learned during each iteration.

We initialize each λ parameter to the log odds of the corresponding feature being true in the data, which is the parameter value that would be obtained if all features were independent of each other. Notice that the value of the information nodes is not available in the training data. We initialize them as follows. An information node is initialized to 1 if there is at least one record node linked to the information node whose value is 1, otherwise we initialize it to 0. This reflects the notion that, if two records are the same, all of their corresponding fields should also be the same.

3.11 Canopies

If we consider each possible pair of records for a match, the potential number of matches becomes $O(n^2)$, which is a very large number even for databases of moderate size. Therefore, we use the technique of first clustering the database into possibly-overlapping *canopies* as described by [10], and then applying our learning/inference algorithms only to record pairs which fall in the same canopy. This reduces the potential number of matches by a large factor. For example, for a 650-record database we obtained on the order of 15000 potential matches after forming the canopies. In our experiments we used this technique with both our model and the standard one. The basic intuition behind the use of canopies and related techniques in de-duplication is that most record pairs are very clearly non-matches, and the plausible candidate matches can be found very efficiently using a simple distance measure based on an inverted index.

Table 2. Performance of the two models on the Cora database

Model	F-measure(%)	Recall(%)	Precision(%)
Standard	84.4	81.5	88.5
Collective	87.0	89.0	85.8

Table 3. Performance comparison after taking the transitive closure

Model	F-measure(%)	Recall(%)	Precision(%)
Standard	80.7	92.0	73.7
Collective	87.0	90.9	84.2

4 Experiments

To evaluate our model, we performed experiments on real and semi-artificial databases. This section describes the databases, methodology and results. The results that we report are inclusive of the canopy process, i.e., they are over all the possible $O(n^2)$ candidate match pairs. The evidence node values were computed using cosine similarity with TF/IDF [15].

4.1 Real-World Data

Our primary source of data was the hand-labeled subset of the Cora database provided by Andrew McCallum and previously used by Bilenko and Mooney [2] and others.[1] This dataset is a collection of 1295 different citations to 112 computer science research papers from the Cora Computer Science Research Paper Engine. The original data set contains only unsegmented citation strings. Bilenko and Mooney [2] used a segmented version of the data for their experiments, with each bibliographic reference split into its constituent fields (author, venue, title, publisher, year, etc.) using an information extraction system. We used this processed version of the Cora dataset for our experiments. We used only the three most informative attributes: author, title and venue (with venue encompassing different types of publication venue, such as conferences, journals,workshops, etc.).

We divided the data into equal-sized training and test sets, ensuring that no true set of matching records was split among the two, to avoid contamination of the test data by the training set. We performed two-fold cross-validation, and report the average F-measure, recall and precision [15] over twenty different random splits. We trained the models using a number of iterations that was first determined using a validation subset of the data. The "optimal" number of iterations was 125 for the collective model and 17 for the standard one. The results are shown in Table 2. The collective model gives an F-measure gain of about 2.5% over the standard model, which is the result of a large gain in recall

[1] http://www.cs.umass.edu/∼mccallum/data/cora-refs.tar.gz

that outweighs a smaller loss in precision. Next, we took the transitive closure over the matches produced by each model as a post-processing step to remove any inconsistent decisions. Table 3 compares the performance of the standard and the collective model after this step. The recall of the standard model is greatly improved, but the precision is reduced even more drastically, resulting in a substantial deterioration in F-measure. This points to the fact that the standard model makes a lot of decisions which are inconsistent with each other. On the other hand, the collective model is relatively stable with respect to the transitive closure step, with its F-measure remaining the same as a result of a small increase in recall and a small loss in precision. The net F-measure gain of the collective model over the standard model after transitive closure is about 6.2%. This relative stability of the collective model leads us to infer that the flow of information it facilitates not only improves predictive performance but also helps to produce overall consistent decisions.

We hypothesize that as we move to larger databases (in number of records and number of attributes) the advantage of our model will become more pronounced, because there will be many more interactions between sets of candidate pairs which our model can potentially benefit from.

4.2 Semi-Artificial Data

To further observe the behavior of the algorithms, we generated variants of the Cora database by taking distinct field values from the original database and randomly combining them to generate distinct papers. The semi-artificial data has the advantage that we can control various factors like the number of clusters, level of distortion, etc., and observe how these factors affect the performance of our algorithm. To generate the semi-artificial database, we first made a list of author, title and venue field values. In particular, we had 80 distinct titles, 40 different venues and 20 different authors. Then, for each field value, we created a fixed number of distorted duplicates of the string value (in our current experiments, we created 8 different distorted duplicates for each field value). The number of distortions within each duplicate was chosen according to a binomial distribution whose Bernoulli parameter (success probability) we varied in our experiments. A single Bernoulli trial corresponds to the distortion of a single word in the original string. For each word that we decided to perturb, we randomly chose between one of the following: introduce a spelling mistake, replace by a word from another field value, or delete the word. To generate the records in the database, we first decided the total number of clusters the database would have. We varied this number in our experiments. The total number of documents was kept constant at 1000 across all the experiments we carried out with semi-artificial data. For each cluster to be generated, we randomly chose a combination of original field values. This uniquely determines a cluster. To create the duplicate records within each cluster, we randomly chose, for each field value assigned to the cluster, one of the corresponding distorted field duplicates.

In the first set of experiments on the semi-artificial databases, our aim was to analyze the relative performances of the standard model and the collective model

as we vary the number of clusters. We used 50, 100, 200, 300 and 400 clusters. The average number of records per cluster was varied inversely, to keep the total number of records in the database constant (at 1000). The distortion parameter was kept at 0.4. Figures 3(a), 3(c) and 3(e) show the results. Each data point was obtained by performing two-fold cross validation over five random splits of the data. All the results reported are before taking the transitive closure over the matching pairs. The F-measure (Figure 3(a)) drops as the number of clusters is increased, but the collective model always outperforms the standard model. The recall curve (Figure 3(c)) shows similar behavior. Precision (Figure 3(e)) seems to drop with increasing number of clusters, with neither of the models emerging as the clear winner.

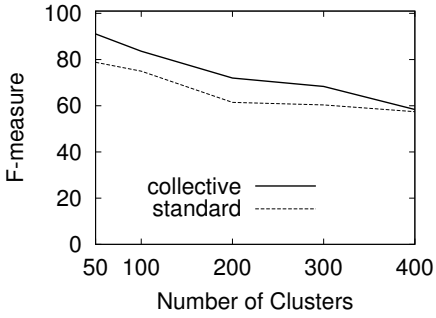
In the second set of experiments on the semi-artificial databases, our aim was to analyze the relative performances of the standard model and the collective model as we vary the level of distortion in the data. We varied the distortion parameter from 0 to 1, at intervals of 0.2. 0 means no distortion and 1 means that every word in the string is distorted. The number of clusters in the database was kept constant at 100, the total number of documents in the database being 1000. Figures 3(b), 3(d) and 3(f) show the results. Each data point was obtained by performing two-fold cross validation over five random splits of the data. All the results reported are before taking the transitive closure over the matching pairs. As expected, the F-measure (Figure 3(b)) drops as the level of distortion in the data is increased. The collective model outperforms the standard model at all levels of distortion. The recall curve (Figure 3(d)) shows similar behavior. Precision (Figure 3(f)) initially drops with increasing distortion, but then partly recovers. The collective model performs as well as or better than the standard model until the distortion level reaches 0.4, after which the standard model takes over.

In summary, these experiments support the hypothesis that the collective model yields improved predictive performance relative to the standard model. It improves F-measure as a result of a substantial gain in recall while reducing precision by a smaller amount. Investigating these effects and trading off precision and recall in our framework are significant items for future work.

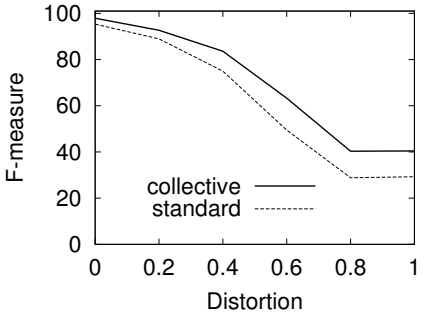
5 Related Work

Most work on the record linkage problem to date has been based on computing pairwise distances and collapsing two records if their distance falls below a certain threshold. This is typically followed by taking a transitive closure over the matching pairs. The problem of record linkage was originally proposed by Newcombe [13], and placed into a rigorous statistical framework by Fellegi and Sunter [6]. Winkler [19] provides an overview of systems for record linkage. There is a substantial literature on record linkage within the KDD community ([8], [3], [12], [4],[16], [18], [2], etc.).

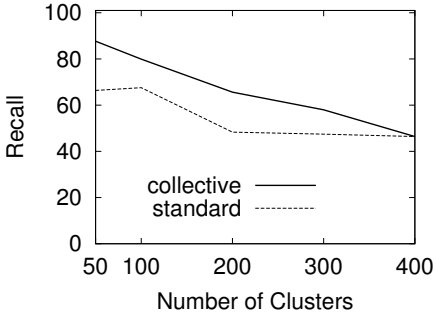
Recently, Pasula et al. proposed a multi-relational approach to the related problem of reference matching [14]. This approach is based on directed graphi-



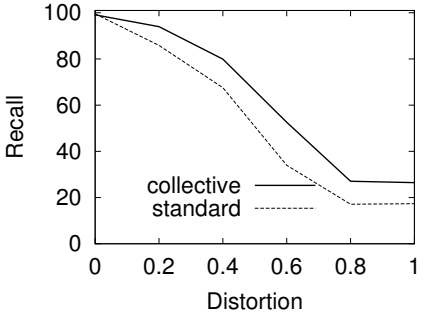
(a) F-measure as a function of the number of clusters



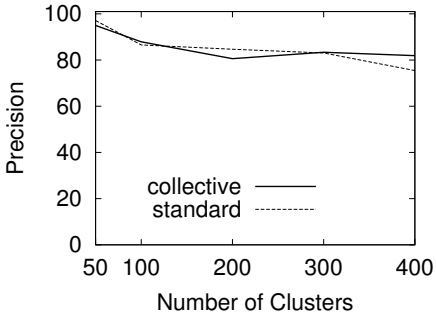
(b) F-measure as a function of the level of distortion



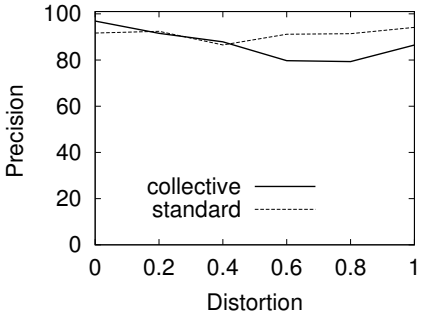
(c) Recall as a function of the number of clusters



(d) Recall as a function of the level of distortion



(e) Precision as a function of the number of clusters



(f) Precision as a function of the level of distortion

Fig. 3. Performance of the two models on semi-artificial datasets

cal models and a different representation of the matching problem, also includes parsing of the references into fields, and is quite complex. In particular, it is a generative rather than discriminative approach, requiring modeling of all dependencies among all variables, and the learning task is correspondingly more difficult. A multi-relational discriminative approach has been proposed by McCallum and Wellner [11]. The only inference performed across candidate pairs, however, is the transitive closure that is traditionally done as a post-processing step. While our approach borrows much of the conditional machinery developed by McCallum et al., its representation of the problem and propagation of information through shared attribute values are new.

Taskar et al. [17] introduced relational Markov networks, which are conditional random fields with templates for cliques as described in Section 3.1, and applied them to a Web mining task. Each template constructs a set of similar cliques via a conjunctive query over the database of interest. Our model is very similar to a relational Markov network, except that it cannot be directly constructed by such queries; rather, the cliques are over nodes for the relevant record and attribute pairs that must first be created.

6 Conclusion and Future Work

Record linkage or de-duplication is a key problem in KDD. With few exceptions, current approaches solve the problem for each candidate pair independently. In this paper, we argued that a potentially more accurate approach to the problem is to set up a network with a node for each record pair and each attribute pair, and use it to infer matches for all the pairs simultaneously. We designed a framework for collective inference where information is propagated through shared attribute values of record pairs. Our experiments confirm that our approach outperforms the standard approach.

We plan to apply our approach to a variety of domains other than the bibliography domain. So far, we have experimented with relations involving only a few attributes. We envisage that as the number of attributes increases, there will be potentially more sharing among attribute values, and our approach should be able to take advantage of it.

In the current model, we use only cliques of size two. Although this has the advantage of allowing for polynomial-time exact inference, it is a strong restriction on the types of dependencies that can be modeled. In the future we would like to experiment with introducing larger cliques in our model, which will entail moving to approximate inference.

Acknowledgements

This research was partly supported by ONR grant N00014-02-1-0408, by a gift from the Ford Motor Co., and by a Sloan Fellowship to the second author.

References

1. A. Agresti. *Categorical Data Analysis*. Wiley, New York, NY, 1990.
2. M. Bilenko and R. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *Proc. 9th SIGKDD*, pages 7–12, 2003.
3. W. Cohen, H. Kautz, and D. McAllester. Hardening soft information sources. In *Proc. 6th SIGKDD*, pages 255–259, 2000.
4. W. Cohen and J. Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *Proc. 8th SIGKDD*, pages 475–480, 2002.
5. M. Collins. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proc. 2002 EMNLP*, 2002.
6. I. Fellegi and A. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64:1183–1210, 1969.
7. D. M. Greig, B. T. Porteous, and A. H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society, Series B*, 51:271–279, 1989.
8. M. Hernandez and S. Stolfo. The merge/purge problem for large databases. In *Proc. 1995 SIGMOD*, pages 127–138, 1995.
9. J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th ICML*, pages 282–289, 2001.
10. A. McCallum, K. Nigam, and L. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proc. 6th SIGKDD*, pages 169–178, 2000.
11. A. McCallum and B. Wellner. Object consolidation by graph partitioning with a conditionally trained distance metric. In *Proc. SIGKDD-2003 Workshop on Data Cleaning, Record Linkage, and Object Consolidation*, pages 19–24, 2003.
12. A. Monge and C. Elkan. An efficient domain-independent algorithm for detecting approximately duplicate database records. In *Proc. SIGMOD-1997 Workshop on Research Issues in Data Mining and Knowledge Discovery*, 1997.
13. H. Newcombe, J. Kennedy, S. Axford, and A. James. Automatic linkage of vital records. *Science*, 130:954–959, 1959.
14. H. Pasula, B. Marthi, B. Milch, S. Russell, and I. Shpitser. Identity uncertainty and citation matching. In *Adv. NIPS 15*, 2003.
15. G. Salton and M. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, NY, 1983.
16. S. Sarawagi and A. Bhamidipaty. Interactive deduplication using active learning. In *Proc. 8th SIGKDD*, pages 269–278, 2002.
17. B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Proc. 18th UAI*, pages 485–492, 2002.
18. S. Tejada, C. Knoblock, and S. Minton. Learning domain-independent string transformation weights for high accuracy object identification. In *Proc. 8th SIGKDD*, pages 350–359, 2002.
19. W. Winkler. The state of record linkage and current research problems. Technical report, Statistical Research Division, U.S. Census Bureau, 1999.