A Unified Approach for Schema Matching, Coreference and Canonicalization

Michael Wick
Computer Science
University of Massachusetts
Amherst, MA 01003
mwick@cs.umass.edu

Khashayar Rohanimanesh
Computer Science
University of Massachusetts
Amherst, MA 01003
khash@cs.umass.edu

Karl Schultz
Computer Science
University of Massachusetts
Amherst, MA 01003
kschultz@cs.umass.edu

Andrew McCallum
Computer Science
University of Massachusetts
Amherst, MA 01003
mccallum@cs.umass.edu

ABSTRACT

The automatic consolidation of database records from many heterogeneous sources into a single repository requires solving several information integration tasks. Although tasks such as coreference, schema matching, and canonicalization are closely related, they are most commonly studied in isolation. Systems that do tackle multiple integration problems traditionally solve each independently, allowing errors to propagate from one task to another. In this paper, we describe a discriminatively-trained model that reasons about schema matching, coreference, and canonicalization jointly. We evaluate our model on a real-world data set of people and demonstrate that simultaneously solving these tasks reduces errors over a cascaded or isolated approach. Our experiments show that a joint model is able to improve substantially over systems that either solve each task in isolation or with the conventional cascade. We demonstrate nearly a 50% error reduction for coreference and a 40% error reduction for schema matching.

Categories and Subject Descriptors

H.2 [**Information Systems**]: Database Management; H.2.8

General Terms

Algorithms,

Keywords

Data Integration, Coreference, Schema Matching, Canonicalization, Conditional Random Field, Weighted Logic

KDD'08, August 24–27, 2008, Las Vegas, Nevada, USA.

1. INTRODUCTION

As the amount of electronically available information continues to grow, automatic knowledge discovery is becoming increasingly important. Unfortunately, electronic information is typically spread across multiple heterogeneous resources (databases with different schemas, or web documents with different structures) making it necessary to consolidate the data into a single repository or representation before data mining can be successfully applied. However, data integration is a challenging problem. Even the task of merging two databases with similar schemas about the same real-world entities is non-trivial. An automatic system must be able to perform coreference (to identify duplicate records), canonicalization (to pick the best string representation of the duplicate record), and schema matching (to align the fields across schemas).

Coreference and other integration tasks have been studied almost exclusively in isolation, yet the individual problems are highly correlated. As an example, consider the two different data records of a person named John Smith in Table 1. Each data record is represented using a different schema. In this example, knowing that the $Contact$ attribute from schema A maps to the $Phone$ attribute from schema B provides strong evidence that the two mentions are coreferent, indicating that schema matching is a valuable precursor to coreference. However, knowing that the two John Smith mentions are coreferent provides strong evidence about which fields should be matched across the schemas (for example, the $FirstName$ and $LastName$ attributes of schema A should be mapped to the $Name$ attribute of schema B). The high correlation of these two tasks indicate that a cascaded approach, where one task must be solved before the other, is likely to lead to gratuitous error propagation.

To motivate the idea further, consider the task of canonicalization, the process of creating a single standardized representation of a record from several different alternatives. The result of canonicalization on a set of records is a single record containing a high density of information about a real-world entity. Intuitively, these canonical representations of entities contain valuable evidence for coreference. We would

Schema A		Schema B	
FirstName	John	**Name**	John Smith
MiddleName	R.	**Phone**	1 -(222)-222-2222
LastName	Smith		
Contact	222-222-2222		

Table 1: Two records from different schemas representing the same John Smith

like exploit this entity-level information, yet canonicalization assumes coreference has already been performed.

In this paper we investigate a unified approach to data integration that jointly models several tasks and their dependencies. More precisely, we propose a conditional random field for simultaneously solving coreference resolution, record canonicalization, and schema matching. As described in Section 3.1, one particular feature of our model is that it automatically discovers the top level canonical schema. We use first order logic clauses for parameter tying, effectively combining logic and probability in a manner similar to [24, 7, 19]. Exact inference and learning in these models are intractable, thus we present approximate solutions to both these problems. Our approximations prove to be effective allowing us to achieve almost a 50% reduction in error for coreference and a 40% error reduction in schema matching over non-joint baselines.

2. RELATED WORK

2.1 Coreference Resolution

Coreference is a pervasive problem in data integration that has been studied in several different domains. The ACE and MUC corpora have helped initiate a line of research on newswire coreference, beginning with approaches that examine mention pairs [25, 18, 12] to more complicated models that reason over entire sets of mentions [7]. Person disambiguation, another form of coreference, has also been studied in detail in [21, 11, 10, 26, 14, 4, 1]. However, these works only resolve coreference between objects of the same representation (e.g., database schema). The coreference problem we tackle involves objects that contain different representations, making direct comparisons between these objects difficult (for example, we may not know in advance that *Phone* from schema A maps to *Contact* in schema B). The coreference portion of our model factorizes over sets of mentions and incorporates first order logic features making it most similar to Culotta et al. [7].

2.2 Canonicalization

Since records can refer to the same underlying entity in multiple ways (common aliases, acronyms and abbreviations), it is often necessary to choose a single and standardized representation when displaying the result to a user, or storing it compactly in a database. Additionally, because the canonical record is constructed from multiple records, it contains a high density of information about the entity, making it a convenient source of evidence for coreference resolution.

Canonicalization has played an important role in systems that perform coreference and database merging. Traditionally, it is performed *post hoc* and often relies on metrics for evaluating distances between strings. An example of canonicalization in a database merging task is Zhu and Unger [28], who obtain positive results by learning string edit parameters with a genetic algorithm. McCallum et al. [17] extend usual edit distance models with a conditional random field, demonstrating more accurate distance evaluations on several corpora; however, they do not apply their string distance model to the problem of canonicalization. Other approaches include Ristad and Yianilos [23], who use expectation maximization to learn the parameters of a generative model that defines a string in terms of the string edit operations required to create it. This work is extended and applied successfully to record deduplication by Bilenko and Mooney [2]. Recently, Culotta et al. [6] describe several methods for canonicalization of database records that are robust to noisy data and customizable to user preferences (e.g., a preference for acronyms versus full words).

2.3 Schema Matching

Schema and ontology mapping are fundamental problems in many database application domains such as data integration, E-business, data warehousing, and semantic query processing. In general we can identify two major challenges with the schema matching (and ontology mapping) problem: (1) structural heterogeneity and, (2) semantic heterogeneity. The former concerns the different representations of information where the same information can be represented in different ways. This is a common problem with heterogeneous databases. The latter deals with the intended meaning of the described information. More specifically we can identify the following differences between schemas [27]: (1) structural conflicts concerned with different semantic structures; (2) naming conflicts where different attribute names may be used for the same type of information, or the same name for slightly different types of information; (3) conflicts where different formats maybe used to represent the values of attributes (for example, different units, different precision, or different abbreviation styles). This problem has been extensively studied primarily by the database and machine learning communities [20, 13, 15, 8, 9, 27] (for a survey of the traditional approaches refer to [22]).

Our model is able to reason about all three different kinds of conflicts mentioned above, based on a set of first order logic features employed by the CRF modeling the task. Our approach also differs from previous systems in that schema matching is performed jointly along with coreference and canonicalization, resulting in a significant error reduction as we will see in Section 5. One important aspect of our model is that it will automatically discover the top level canonical schema for the integrated data as will be demonstrated in Section 3.1.

3. PROBLEM DEFINITION

We seek a general representation that allows joint reasoning over a set of tasks defined on a set of possibly hetero-

geneous objects in the world. In our unified data integration approach we aim for a representation that enables us to perform inference and learning over two different types of objects: (1) data records (in relation to the coreference resolution and canonicalization tasks); (2) schema attributes (in relation to the schema matching task). In abstract terms, our model finds solutions to this problem in terms of a set of partitions (clusters) of data records where all the records within a particular partition are coreferent and canonicalized; and also a set of partitions (clusters) of the schema attributes across different databases, where all the attributes within a schema partition are mapped together. As we will see shortly, although data record clusters and schema clusters are kept disjoint in terms of the type of objects they contain, they are tied together through a set of factors that compute different features associated with each task. Thus, inference is performed jointly over all three tasks. In the next section we describe a graphical model representation of this approach in more detail.

3.1 Model

We use a conditional random field (CRF) [16] for jointly representing schema matching, coreference resolution, and canonicalization tasks as follows:

• Let $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_k\}$ denote a set of k databases. Each database $\mathcal{D}_i$ is represented as a 2-tuple $\mathcal{D}_i \equiv \langle \mathcal{S}_i, \mathcal{R}_i \rangle$, where $\mathcal{S}_i$ is a database schema, and $\mathcal{R}^i = \{r_j^i\}_{j=1}^{n_{\mathcal{D}_i}}$ is a set of data records generated using the schema $\mathcal{S}_i$. Each schema $\mathcal{S}_i$ is represented by a set of attributes $\{a_j^i\}_{j=1}^{n_{\mathcal{S}_i}}$ where a_j^i is the j^{th} attribute of the schema $\mathcal{S}_i$. We use $\mathcal{S} = \{\mathcal{S}_1, \ldots, \mathcal{S}_k\}$ to represent the complete set of schemas, and $\mathcal{A} = \{a_j^i\}$ to denote the complete set of schema attributes across all schemas.

• Let $X = \{x_1 \ldots x_n\} \cup \mathcal{A}$ denote a set of observed variables, where x_i is a data record drawn from some database $\mathcal{D}_l$, and $\mathcal{A}$ is the set of all schema attributes $\{a_j^i\}$.

• Let $\mathcal{X}_i$ denote a particular grouping of the observed variables X constrained by the fact that all the variables in a cluster $\mathcal{X}_i$ denote the same type of the objects (all the observed variables in a cluster $\mathcal{X}_i$ are either data record objects, or schema attribute objects). For clarity we use $\mathcal{X}_i^r$ to denote a cluster of data records, and $\mathcal{X}_j^s$ to denote a cluster of schema attributes.

• Let $Y = \{y_1 \ldots y_m\}$ denote a set of unobserved variables that we wish to predict. In this paper we focus only on a particular class of clustering models[1] where the variables y_i indicate some compatibility among clusters of X variables (i.e., clusters $\mathcal{X}_i^r$ and $\mathcal{X}_j^s$). We employ three types of Y variables: (1) variables that indicate the compatibility among instances within a cluster of X variables; (2) variables that indicate the compatibility between a *pair* of clusters of the same type (compatibility between two clusters of data records, or two clusters of schema attributes); (3) variables that indicate the compatibility between a *pair* of clusters of different types (compatibility between a data record cluster and a schema attribute cluster). Note that this representation renders an exponential complexity when instantiating the Y, $\mathcal{X}_i^r$, and $\mathcal{X}_j^s$ variables (e.g., $|Y| = O(2^{|X|})$, and $|\mathcal{X}_i^r| = O(2^n)$, and $|\mathcal{X}_i^r| = O(2^{|A|})$).

Let $\mathcal{X}_i$ denote a particular grouping of the observed variables X constrained by the fact that all variables in a cluster $\mathcal{X}_i$ denote the same type of the objects (all the observed variables in a cluster $\mathcal{X}_i$ are either data record objects, or schema attribute objects). For clarity we use $\mathcal{X}_i^r$ to denote a cluster of data records, and $\mathcal{X}_j^s$ to denote a cluster of schema attributes. Our goal is to learn a clustering $\mathcal{X} = \{\mathcal{X}_i^r\} \cup \{\mathcal{X}_j^s\}$ such that all the data records in $\mathcal{X}_i^r$ are coreferent, and all the schema attributes in $\mathcal{X}_j^s$ bear the correct schema matching.

Next, the conditional probability distribution $P(Y|X)$ is computed as:

$$P(Y|X) = \frac{1}{Z_X} \prod_{y_i \in Y} f_w(y_i, \mathcal{X}_i) \prod_{y_i, y_j \in Y} f_b(y_{ij}, \mathcal{X}_{ij}) \qquad (1)$$

where Z_X is the input-dependent normalizer, factor f_w parameterizes the *within-cluster* compatibility, and factor f_b parameterizes the *between-cluster* compatibility[2]. Note that in our model there are two types of within cluster factors f_w: those measuring the compatibility within a data record cluster (e.g., $\mathcal{X}_i^r$), and those measuring the compatibility within a schema attribute cluster (e.g., $\mathcal{X}_i^s$). Similarly, there are two types of between cluster factors f_b: those measuring the compatibility between a pair of homogeneous clusters (two data record clusters, or two schema attribute clusters), and those measuring the compatibility between a pair of heterogeneous clusters (a data record cluster and a schema attribute cluster). We also employ a log-linear model of potential functions (i.e., f_w and f_b):

$$f(y_i, x_i) = \exp\left(\sum_k \lambda_k g_k(y_i, x_i)\right)$$

This model can be intuitively described as follows: every possible clustering of the data induces a different set of instantiations of Y variables and possibly gives different assignments to them. The conditional distribution $P(Y|X)$ gives the probability of a configuration Y measured in terms of a normalized score of how likely that configuration is. We parameterize this score with a set of potential functions that evaluate the compatibility of both within-cluster attributes and between-cluster attributes.

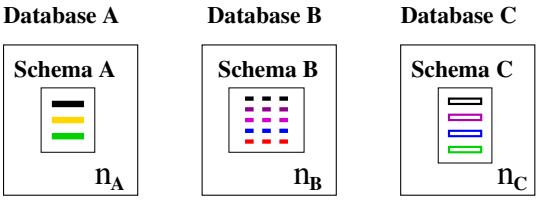

Figure 1: Three example databases: each database uses a different schema to generate a set of data records. Each schema is visually represented as a collection of color-coded objects of the same shape (solid rectangles in schema A, dotted lines in schema B, and hollow rectangles in schema C).

A desirable facet of our model is that it factorizes into clusters of data rather than pairs (Equation 1). This enables us to define features of entire clusters using *first-order*

[1] In general the inference and learning methods could be applied to a variety of model structures [5].

[2] In the above equation we use the notation $\mathcal{X}_{ij}$ to denote a pair of clusters $\mathcal{X}_i$ and $\mathcal{X}_j$.

logic features: features that can universally and existentially aggregate properties of a set of objects [5].

To further illustrate the model consider the simple example task demonstrated in Figure 1. There are three databases, where each database uses a different schema to generate a set of data records. Each schema is visually represented as a collection of color-coded objects of the same shape (solid rectangles in schema **A**, dotted lines in schema **B**, and hollow rectangles in schema **C**). Within each schema, different attributes are color-coded, and similar color across different schemas may refer to the same attribute concept. Each database consists of a number of data records generated using its own schema (for example, database **A** contains n_A data records generated using schema **A**). The goal is to perform joint inference among coreference resolution, canonicalization, and schema matching.

Figure 2 displays a factor graph of the conditional random field modeling the above joint task. There are two levels of clustering processes:

• **Schema attribute clusters (top level):** Each cluster in this level consists of a subset of the complete set of schema attributes. Note that two or more attributes of the same schema may be placed within the same cluster together with the attributes of other schemas. For example, one database may use a schema that has an attribute *Name* to represent the full name of a person, while a second database may use a schema with two different attributes, *First Name* and *Last Name*, for representing the same concept. Intuitively, we would like to place all three in the same cluster. Some clusters such as $\mathcal{X}_7$ may contain a single schema attribute, meaning that it does not match to any other schema attributes in the other databases. Note that the set of schema attribute clusters establishes the top level canonical schema for the integrated data (lightly shaded clusters).

• **Data record clusters (bottom level):** Each cluster represents a set of coreferent data records. Note that every data record is visually represented as an encapsulation of the schema from which it was generated. For example cluster $\mathcal{X}_1$ consists of a single data record from database **A**, a single data record of database **B**, and two data records from database **C**. There may also exists clusters that contain a set of data records from the same database (for example the cluster $\mathcal{X}_3$) due to duplication errors in that database.

• **Factors:** Although the clusterings at different levels are defined over the same type of objects (data records or schema attributes), they are tied using a set of factors. We can identify three types of factors in general: (1) factors that measure compatibility among instances within a cluster (e.g., f_1, or f_4); (2) factors that measure compatibility between pairs of clusters of the same type (compatibility between two clusters of data records such as f_{12}, or two clusters of schema attributes such as f_{67}); (3) factors that measure compatibility between pairs of clusters of different types (compatibility between a data record cluster and a schema attribute cluster such as f_{34}).

Although omitted from Figure 2 for clarity, there are additional canonicalization variables for each attribute in each coreference cluster. Even though we lack labeled data for canonicalization, we set these variables using a centroid-based approach with default settings for string edit parameters (insert, delete and substitute incur a penalty of one, and no penalty is given for copy). This method is shown in recent work by Culotta et al [6] to perform reasonably well and to capture many of the desirable properties of a canonical string.

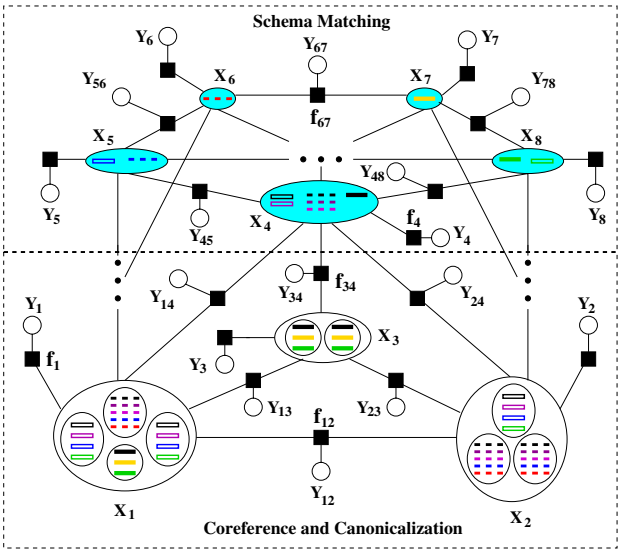

Figure 2: Factor graph representation of the model. There are two clustering processes, one at the level of schema attributes (top level), and one at the level of data records (bottom level). Different factors tie these two processes which allows for joint inference among different data integration tasks. Note that for clarity of the figure not all the factor names are represented. The top level clustering also automatically discovers the top-level canonical schema in the integrated data (lightly shaded clusters).

Even though we are able to achieve greater expressiveness in our model with cluster-wise first order features and high connectivity, we sacrifice the ability to apply exact inference and learning methods, since we cannot instantiate all of the Y variables. In Culotta and McCallum [5], approximate inference and parameter estimation methods operate with partial instantiations, where only the *difference* between two configurations are sufficient to perform learning. Building on these techniques, we briefly demonstrate how learning is performed in this model in the next section.

3.2 Inference and Parameter Estimation

Both the joint model and the individual conditional random fields for each subtask are too large to be fully instantiated, making exact training and inference intractable. In this section we describe in detail our approximate and inference methods.

3.2.1 Training

To learn the parameters for coreference resolution we fix the schema matching to ground-truth and fix the canonicalization to a reasonable default. Next, we sample pairs of clusters C_i and C_j and define the binary random variable $y_{ij} = 1$ if and only if all the mentions in c_i and c_j are coreferent. Given the fixed schema matching and canonicalization, and a whole set of cluster pairs with their corresponding labels, we set the coreference parameters to maximize the likelihood of the training set by performing gradient descent (and regularizing with the usual Gaussian prior). A similar

Algorithm 1 Joint Inference

1: **Input:**
   coreference clustering $\mathcal{C}$
   schema clustering $\mathcal{S}$
2: **while** Not Converged **do**
3: $\quad \mathcal{C} \Leftarrow$ *GreedyAgglomerative*(make-singletons($\mathcal{C}$), $\mathcal{S}$)
4: $\quad \mathcal{S} \Leftarrow$ *GreedyAgglomerative*(make-singletons($\mathcal{S}$), $\mathcal{C}$)
5: **end while**

procedure is used to set the parameters for schema matching. However, in this case the coreference ground-truth is held fixed and the label indicates whether or not all the instances in two schema clusters all match to each other. Canonicalization is not used for the schema-matching task. This training method can be viewed as a piecewise psuedo-likelihood approximation.

3.2.2 Testing

For inference we use a standard greedy agglomerative approximation to each subtask. The algorithm begins with a singleton clustering (each instance is in its own cluster) and greedily merges clusters until no merge scores are above a stopping threshold τ. Joint inference works in rounds, performing greedy agglomerative first on coreference, and then on schema-matching. The coreference prediction in $round_i$ is used to help schema matching in $round_i$, whereas a schema matching prediction from $round_i$ is used to help coreference in $round_{i+1}$. This process can be repeated for a fixed number of iterations.

4. EXPERIMENTS

4.1 Data

Synthetic data: The synthetic data is generated from a small number (10) of user records collected manually from the web. These records use a canonical schema containing attributes such as first name, phone number, email address, job title, institution, etc. Next, we created three new schemas derived from the canonical schema by randomly splitting, merging, or noisifying the attributes of the canonical schema. For example, one schema would contain a *Name* field whereas another would contain two fields, *FirstName* and *LastName*, for the same block of information (perhaps dropping the middle name if it existed).

In the training data we used the first two schemas and in the testing data we used one of the schemas from training, and also the third schema. This way we train a model on one schema but test it on another schema. For training we used a small number of user records that were similar in ways such that random permutations could make coreference, schema matching, and canonicalization decisions difficult. We first conformed the records to both schemas, and then made 25-30 copies of each data record for each schema while randomly permuting some of the fields to introduce noise.

The testing data was created similarly, but for a different set of data records. The random permutations included abbreviating fields, deleting an entire field, or removing a random number of tokens from the front and/or the end of a field. The result of this was a large number of coreferent records, but with the possibility that the disambiguating fields between different records have been altered or removed.

Real World Data: For our real-world data we manually extracted faculty and alumni listings from 7 university websites, of which we selected 8 lists that had different schemas. The information in each schema contains basic user data such as first name, phone number, email address, job title, institution, etc., as well as fields unique to the academic domain such as advisor, thesis title, and alma mater. For each name that had an e-mail address we used our DEX [3] system to search the Internet for that person's homepage, and if found we would extract another record. The DEX schema is similar to the university listings data, bringing the total number of different schemas to 9. Of the nearly 1400 mentions extracted we found 294 coreferent entities. Table 2 shows the DEX schema and two of the faculty listing schemas. There are several schema matching problems evident in table 2, for example *Job Department* from the UPenn schema is a superset of both of the *Job Title* fields. Another example, which occurs numerous times between other schemas, is where the pair of attributes $\langle$*First Name*, *Last Name*$\rangle$ from one schema is mapped to the singleton attribute *Name* that denotes the full name.

For each of the experiments we took all of the 294 coreferent clusters, and randomly selected 200 additional mentions that had no coreferent entities. This data was split into training and testing sets, where the only schema shared between training and testing was the DEX schema. This ensures that the possible schema matchings in the training set are disjoint from the possible schema matchings in the test set. The data provides us with a variety of schemas that map to each other in different ways, thus making an appropriate testbed for performing joint schema matching, coreference, and canonicalization.

4.2 Features

The features are first-order logic clauses that aggregate pairwise feature extractions. The types of aggregation depend on whether the extractor is real-valued or boolean. For real-valued features we compute the minimum, the maximum, and the average value over all pairwise combinations of records. For boolean-valued features we compute the following over all pairwise combinations of records: feature does not exist; feature exists; feature exists for the majority; feature exists for the minority; feature exists for all.

Table 3 lists the set of features used in our experiments. In cases where we compute features between records in different schemas, sometimes we only compute the feature between fields that are aligned in the current schema matching. This is indicated by the *Matched-fields only* column in Table 3. All of these features are used for coreference decisions, but only substring matching is used for schema matching.

4.3 Systems

We evaluate the following three systems with and without canonicalization for a total of six systems. Canonicalization is integrated with coreference inference by interleaving canonicalization with greedy agglomerative merges (every time a new cluster is created, a new canonical record is computed for it). Canonicalization has no affect on schema matching in isolation and is only relevant when coreference is involved.

Whenever we use greedy agglomerative inference, we set the stopping threshold to $\tau = 0.5$. This is a natural choice as it corresponds to the decision boundary for a binary max-

DEX	Northwestern Faculty	UPenn Faculty
First Name	Name	Name
Middle Name	Job Title	First Name
Last Name	PhD Alma Mater	Last Name
Title	Research Interests	Job Department
Job Title	E-mail	Office Address Line
Department		Work Phone
Company Name		E-mail
Home Phone Number		
Office Phone Number		
Fax Number		
etc ...		

Table 2: Three example schemas used in the real-world data.

Description	Matched-fields only	real/boolean
Unweighted Cosine distance between mentions	no	real
TFIDF Cosine distance between mentions	no	real
TFIDF Cosine distance between mentions	yes	real
Substring match	yes	boolean

Table 3: pairwise feature extractors

imum entropy classifier. Additionally, the joint inference method described earlier is run for four rounds for the joint system.

• **Isolation (ISO):** this system performs schema matching and coreference in complete isolation from each other. Greedy agglomerative search is used as an inference method in each task.

• **Cascaded (CASC):** the cascaded approach executes schema matching and coreference in a pipelined fashion, either performing schema matching first and then coreference, or the other way around. The predicted result of the first task has the potential to influence the second task.

• **Joint (JOINT):** the joint approach integrates coreference and schema matching allowing predictions in both tasks to influence future predictions in the other. The system iteratively runs the cascaded approach, allowing predictions to propagate in both directions between schema matching and coreference resolution.

5. RESULTS

We compare our joint model of coreference, canonicalization, and schema matching to the baselines of the cascaded and isolated inference (both with and without the benefits of joint canonicalization). Although we report precision, recall, and F1 for both Pairwise and MUC evaluation metrics, the error reductions discussed in the body of text below are in terms of Pairwise F1 only. Table 4 displays the impact on coreference performance and Table 5 shows schema matching. As expected, joint inference between coreference and schema matching improves performance in both tasks over the baselines of performing each task in isolation or as a cascade. Overall, by performing all three tasks jointly, an error reduction of 49% is realized in coreference over the cascaded approach and 32.1% over coreference in isolation. For schema matching, the joint model reduces error by 40% over both the isolated and cascaded approaches.

5.1 Cascades and Error Propagation

Although the cascaded approach of performing coreference followed by schema matching performs slightly better (with canonicalization) or the same (without canonicalization) as performing schema matching in isolation, the reverse is not true. First performing schema matching and then coreference actually has a negative impact on coreference performance when compared to just performing coreference in isolation (regardless of whether or or not canonicalization is performed). This is not surprising as the schema matching in isolation achieves just 50.9% f1, thus passing a large amount of error to the subsequent coreference task. This result helps highlights the potential danger of error propagation in cascade systems.

5.2 Impact of Joint Canonicalization

Canonicalization has a positive impact on performance on nearly all the systems. The only system it does not affect is schema matching in isolation since it is not relevant for that task. Both cascaded systems and both joint systems improve substantially when introducing factors over canonical coreference records. Even the model that jointly does coreference and schema matching is improved by canonicalization: coreference errors are reduced by 21% and schema matching errors by 6.5%.

The cascaded approach of first performing schema matching and then coreference actually does worse than having no schema matching information at all, highlighting the detrimental effect of error propagation in a pipeline. The other cascade of performing coreference first has no effect on the schema matching

5.3 Synthetic Data Experiments

In addition to the real-world data experiments, we also applied both joint approach (joint schema matching and coreference resolution with no canonicalization), and the cascaded approach to our synthetically generated dataset. The models used in these experiments are slightly simpler

		Pair			MUC		
		F1	**Precision**	**Recall**	**F1**	**Precision**	**Recall**
No Canon	ISO	72.7	88.9	61.5	75	88.9	64.9
	CASC	64.0	66.7	61.5	65.7	66.7	64.9
	JOINT	**76.5**	**89.7**	**66.7**	**78.8**	**89.7**	**70.3**
With Canon	ISO	78.3	90.0	69.2	80.6	90.0	73.0
	CASC	65.8	67.6	64.1	67.6	67.6	67.6
	JOINT	**81.7**	**90.6**	**74.4**	**84.1**	**90.6**	**74.4**

Table 4: Coreference results: a comparison of coreference in isolation (ISO), in cascade with schema matching (CASC), and jointly inferred with schema matching (JOINT). The effect of joint canonicalization is also shown (with canon)

		Pair			MUC		
		F1	**Precision**	**Recall**	**F1**	**Precision**	**Recall**
No Canon	ISO	50.9	40.9	67.5	69.2	81.8	60.0
	CASC	50.9	40.9	67.5	69.2	81.8	60.0
	JOINT	**68.9**	**100**	**52.5**	**69.6**	**100**	**53.3**
With Canon	ISO	50.9	40.9	67.5	69.2	81.8	60.0
	CASC	52.3	41.8	70.0	74.1	83.3	66.7
	JOINT	**71.0**	**100**	**55.0**	**75.0**	**100**	**60.0**

Table 5: A comparison of schema matching in isolation (ISO), in cascade with coreference (CASC), and jointly inferred with coreference (JOINT)

since they do not include the canonicalization task, but inference is performed in the exact same manner as described in Section 3.2.

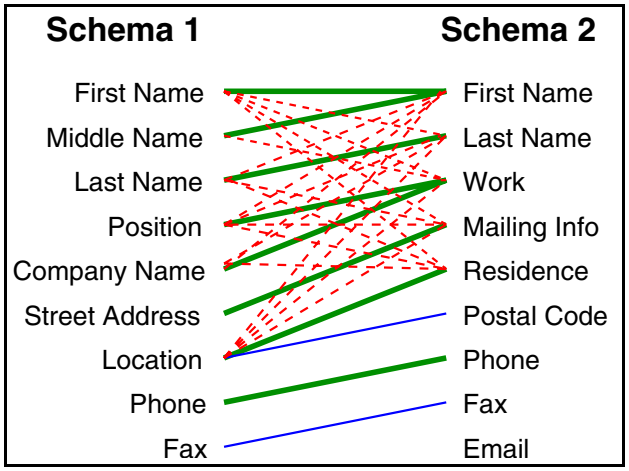

Figure 3: An example schema matching produced by our model using the cascaded approach on synthetic data. The dotted lines represents false positives, while the thick lines represents the correct matches, and the thin lines represents the matchings that were missed. The cascaded approach produces a lot of wrong matches.

As expected we find that joint model on synthetic data reduces coreference resolution errors by 42% compared to the cascaded model. The improvements in schema-matching can be seen graphically: Figures 3 and 4 show examples of schema matching produced by the cascaded and joint approaches respectively. The dotted lines represents false positives, while the thick lines represents the correct matches (hits), and the thin lines represents the matchings that were missed (misses). It can be observed that the cascaded approach produces a lot of false positive matches. The joint approach does not produce any false positive compared to the cascaded approach, while having a relatively low recall.

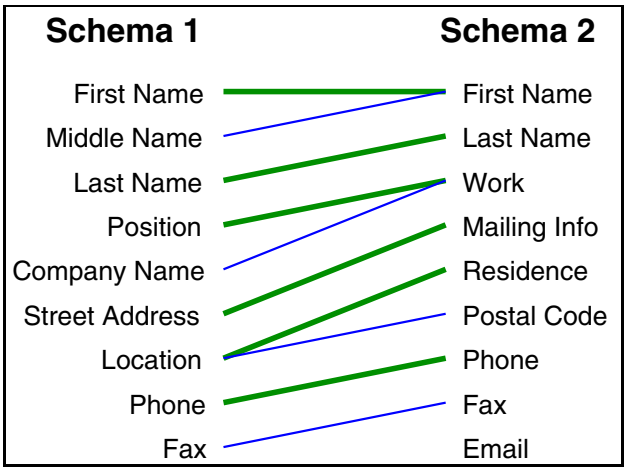

Figure 4: An example schema matching produced by our model using the joint approach on synthetic data. The thick lines represents the correct matches, and the thin lines represents the matchings that were missed. The joint approach does not produce any false positive compared to the cascaded approach.

6. CONCLUSIONS

In this paper we have demonstrated a successful method for performing coreference resolution, record canonicalization, and schema matching simultaneously with a conditional random field. Joint inference in this model outperformed cascaded and isolated approaches on both synthetic

and real-world data despite having to use approximate inference and parameter estimation.

We believe a ripe area of future work would be to explore less greedy inference methods such as Metropolis-Hastings, simulated-annealing, or similar stochastic search methods. Additionally, rank-based learning may be combined with Metropolis-Hastings to train an even larger model that includes first-order logic clauses over entire clustering configurations. Finally, extending the model to include other tasks such as named entity recognition (NER) and record assembly is a natural next step. This could be particularly interesting because exact learning and inference for linear-chain CRFs (like those used in NER) is tractable, yet the other components of the model would have to be approximated.

7. ACKNOWLEDGMENTS

This work was supported in part by the Center for Intelligent Information Retrieval, in part by Lockheed Martin through prime contract #FA8650-06-C-7605 from the Air Force Office of Scientific Research, in part by The Central Intelligence Agency, the National Security Agency and National Science Foundation under NSF grant #IIS-0326249, in part by The Central Intelligence Agency, the National Security Agency and National Science Foundation under NSF grant #IIS-0427594, and in part by the Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under contract number NBCHD030010, and AFRL #FA8750-07-D-0185. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

8. REFERENCES

[1] J. Artiles, J. Gonzalo, and S. Sekine. The semeval-2007 weps evaluation: Establishing a benchmark for the web people search task. In *4th Workshop on Semantic Evaluations (SemEval-2007)*, pages 64–69, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[2] M. Bilenko and R. J. Mooney. Learning to combine trained distance metrics for duplicate detection in databases. Technical Report AI-02-296, University of Texas at Austin, 2002.

[3] A. Culotta, R. Bekkerman, and A. McCallum. Extracting social networks and contact information from email and the web. In *First Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, 2004.

[4] A. Culotta, P. Kanani, R. Hall, M. Wick, and A. McCallum. Author disambiguation using error-driven machine learning with a ranking loss function. In *Sixth International Workshop on Information Integration on the Web (IIWeb-07)*, Vancouver, Canada, 2007.

[5] A. Culotta and A. McCallum. Tractable learning and inference with high-order representations. In *ICML Workshop on Open Problems in Statistical Relational Learning*, Pittsburgh, PA, 2006.

[6] A. Culotta, M. Wick, R. Hall, M. Marzilli, and A. McCallum. Canonicalization of database records using adaptive similarity measures. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, San Jose, CA, 2007. (19% accepted).

[7] A. Culotta, M. Wick, R. Hall, and A. McCallum. First-order probabilistic models for coreference resolution. In *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT/NAACL)*, pages 81–88, 2007. (24% accepted).

[8] A. Doan, P. Domingos, and A. Y. Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In *SIGMOD Conference*, 2001.

[9] A. Doan, J. Madhavan, P. Domingos, and A. Y. Halevy. Learning to map between ontologies on the semantic web. In *WWW*, page 662, 2002.

[10] X. Dong, A. Y. Halevy, E. Nemes, S. B. Sigurdsson, and P. Domingos. Semex: Toward on-the-fly personal information integration. In *IIWEB*, 2004.

[11] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A. Popescu, T. Shaked, S. Soderland, D. Weld, and A. Yates. Web-scale information extraction in KnowItAll. In *WWW*. ACM, May 2004.

[12] A. Haghighi and D. Klein. Unsupervised coreference resolution in a nonparametric bayesian model. In *ACL*, 2007.

[13] R. Ichise, H. Takeda, and S. Honiden. Rule induction for concept hierarchy alignment. In *Workshop on Ontology Learning*, 2001.

[14] P. Kanani, A. McCallum, and C. Pal. Improving author coreference by resource-bounded information gathering from the web. In *Proceedings of IJCAI*, 2007.

[15] M. Lacher and G. Groh. Facilitating the exchange of explicit knowledge through ontology mappings, 2001.

[16] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA, 2001.

[17] A. McCallum, K. Bellare, and F. Pereira. A conditional random field for discriminatively-trained finite-state string edit distance. In *Conference on Uncertainty in AI*, 2005.

[18] A. McCallum and B. Wellner. Conditional models of identity uncertainty with application to noun coreference. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA, 2005.

[19] B. Milch, B. Marthi, S. Russell, D. Sontag, D. L. Ong, and A. Kolobov. BLOG: Probabilistic models with unknown objects. In *IJCAI*, 2005.

[20] N. F. Noy and M. Musen. Anchor-prompt: Using non-local context for semantic matching, 2003. 1 Ontology Merging and.

[21] B.-W. On, D. Lee, J. Kang, and P. Mitra. Comparative study of name disambiguation problem using a scalable blocking-based framework. In *JCDL*, pages 344–353, New York, NY, USA, 2005. ACM Press.

[22] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *The VLDB Journal*, 10(4):334–350, 2001.

[23] E. S. Ristad and P. N. Yianilos. Learning string edit

distance. Technical Report CS-TR-532-96, Princeton University, 1997.

[24] P. Singla and P. Domingos. Entity resolution with markov logic. In *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, pages 572–582, Washington, DC, USA, 2006. IEEE Computer Society.

[25] W. M. Soon, H. T. Ng, and D. C. Y. Lim. A machine learning approach to coreference resolution of noun phrases. *Comput. Linguist.*, 27(4):521–544, 2001.

[26] V. I. Torvik, M. Weeber, D. R. Swanson, and N. R. Smalheiser. A probabilistic similarity metric for medline records: A model for author name disambiguation. *Journal of the American Society for Information Science and Technology*, 56(2):140–158, 2005.

[27] F. Wiesman and N. Roos. Domain independent learning of ontology mappings. In *AAMAS*, page 846, 2004.

[28] J. J. Zhu and L. H. Unger. String edit analysis for merging databases. In *KDD*, 2000.