AI Magazine: Special Issue on Semantic Integration (2005)

Semantic Integration in Text: From Ambiguous Names to Identifiable Entities

Xin Li Paul Morie Dan Roth
Department of Computer Science
University of Illinois, Urbana, IL 61801
{xli1,morie,danr}@uiuc.edu

July 28, 2005

Abstract

Intelligent access to information requires semantic integration of structured databases with unstructured textual resources. While the semantic integration problem has been widely studied in the database domain on structured data, it has not been fully recognized nor studied on unstructured or semi-structured textual resources. This paper presents a first step towards this goal by studying semantic integration in natural language texts — identifying whether different mentions of real world entities, within and across documents, actually represent the same concept.

We present a machine learning study of this problem. The first approach is a discriminative approach — a pairwise local classifier is trained in a supervised way to determine whether two given mentions represent the same real world entity. This is followed, potentially, by a global clustering algorithm that uses the classifier as its similarity metric. Our second approach is a global generative model, at the heart of which is a view on how documents are generated and how names (of different entity types) are "sprinkled" into them. In its most general form, our model assumes: (1) a joint distribution over entities (e.g., a document that mentions "President Kennedy" is more likely to mention "Oswald" or " White House" than "Roger Clemens"), (2) an "author" model, that assumes that at least one mention of an entity in a document is easily identifiable, and then generates other mentions via (3) an appearance model, governing how mentions are transformed from the "representative" mention.

We show that both approaches perform very accurately, in the range of $90\% - 95\%$ F_1 measure for different entity types, much better than previous approaches to (some aspects of) this problem.

1 Introduction

Integration or sharing data across disparate information sources provides a foundation for intelligent and efficient access to heterogenous information. This problem has been

coined the *semantic integration* problem and has been widely studied in the database domain on structured data [RB01, OS99]. In the past few years, a number of rule-based and learning-based techniques have been developed for the *schema matching* problem [MZ98, MBR01, DDH01, MMGR02, DLD+04, NHT+02] — matching related schemas from different structured information sources, and for the *tuple matching* problem [HS95a, TKM02, AMT04, DLLH03] — matching individual, duplicated, data items in databases. Although the same problem has not been fully recognized nor studied on unstructured or semi-structured textual resources, more attention has been diverted to some aspects of it, due to its promising applications [PMM+02, CRF03, BM03, MY03, MW03, GA04].

One of the significant applications is the integration of structured information with unstructured information. Relational databases are usually well structured and thus are able to provide users efficient and accurate access to a large collection of information. However, most available information resources are in textual form – including news articles, books, and online web pages. Lack of well-defined structure makes these resources much harder to access and exploit at the level of databases, both in terms of efficiency and accuracy. There is a need to access textual information efficiently and intelligently and a need to automatically build or augment databases from textual information. As more and more textual information becomes online, this need becomes more and more urgent. One possible solution to integration of structured and unstructured information is concept-based linkage and indexing. Most current databases are relational databases whose organization is centered around entities and relations between them. In this sense, a database has a natural and well-defined mapping from its tuples to real-world entities, and this may lay the foundation for concept-based linkage between databases and text.

At an abstract level, the problem of semantic integration for structured information sources, as defined in the database domain, can be phrased as follows: well defined concepts (in the case of schema matching) and entities (in the case of tuple matching) are manifested when put into databases in multiple, ambiguous occurrences. The matching approaches attempt to identify concepts or entities from these occurrences and allow for the integration of information based on this semantic connection. This abstract definition also captures the problem of semantic integration in texts, which we call here the problem of "robust reading".

Our goal in this article is to describe a first step in the direction of semantic integration of semi-structured and unstructured information, a mechanism that can automatically identify concepts in text and link textual segments representing concepts (*"mentions"* of concepts) to real world objects. Some details of the specific problem studied here are as follows:

Most names of people, locations, organizations, events and others entities, have multiple writings that are being used freely within and across documents. Consider, for example, a user that attempts to acquire a concise answer "on May 29, 1917." to the question"When was President Kennedy born?" by accessing a large collection of textual articles: The sentence, and even the document that contains the answer, may not contain the name "President Kennedy"; it may refer to this entity as "Kennedy", "JFK" or "John Fitzgerald Kennedy". Other documents may state that "John F. Kennedy, Jr. was born on November 25, 1960", but this fact refers to our target entity's son. Other

mentions, such as "Senator Kennedy" or "Mrs. Kennedy" are even "closer" to the writing of the target entity, but clearly refer to different entities. Even the statement "John Kennedy, born 5-29-1941" turns out to refer to a different entity, as one can tell observing that the document discusses Kennedy's batting statistics. A similar problem exists for other entity types, such as locations and organizations.

As discussed above, this problem is related to tuple matching and schema matching as studied in the database domain, but is different from them in at least two important aspects: (1) The information that can be used to discriminate between two names in text is not well-defined. Even when one can identify (in principle) the information, it is still hard to extract it accurately from a document and place it into a well-structured tuple. (2) Textual documents contain, in principle, more information that might influence the decision of whether to merge two mentions. For example, the notion of individual documents might be very significant here. While similar mentions that occur within and across different documents may refer to different entities, and very different (surface level) mentions could refer to the same entity, these variations are typically more restricted within one document. And, learning those variations within a document may contribute to better decisions across documents. Moreover, there is more contextual information for a given mention than in a typical database tuple; this information might include the syntactic structure of the sentence as well as entities that co-occur in the same document.

This problem is related to the general coreference resolution problem in natural language process [SNL01, Keh02], which attempts to determine whether two forms of reference in a text, typically a name (more generally, a noun phrase) and a pronoun, actually refer to the same thing. This is typically done among references that appear within a relatively short distance in one document. The problem we address here has a different goal and a much wider scope. We aim at the identification and disambiguation of real-world concepts from its multiple and ambiguous mentions both within and across documents. Our problem has broader relevance to the problem of intelligent information access and semantic integration across resources.

This paper presents the first attempt to apply a unified approach to all major aspects of the problem of *Robust Reading*. We present two conceptually different machine learning approaches [LMR04a], compare them to existing approaches and show that an unsupervised learning approach can be applied very successfully to this problem, provided that it is used along with strong but realistic assumptions on the usage of names in documents.

Our first model is a discriminative approach that models the problem as that of deciding whether any two names mentioned in a collection of documents represent the same entity. This straightforward modelling of the problem results in a classification problem – as has been done by several other authors [CRF03, BM03] – allowing us to compare our results with these. This is a standard pairwise classification task, and a classifier for it can be trained in a supervised manner; our main contribution in this part is to show how relational (string and token-level) features and structural features, representing transformations between names, can improve the performance of this classifier. Several attempts have been made in the literature to improve the results of a pairwise classifier of this sort by performing some global clustering, with the pairwise classifier

as a similarity metric. The results of these attempts were not conclusive and we provide some explanation for it. First, we show that, in general, a clustering algorithm used in this situation may in fact hurt the results achieved by the pairwise classifier. Then, we argue that attempting to use a locally trained pairwise classifier as a similarity metric might be the wrong choice for this problem. Our experiments concur with this. However, as we show, splitting data in some coherent way – e.g., to groups of documents originated at about the same time period – prevents some of these problems and aids clustering significantly.

This observation motivates our second model, which better exploits structural and global assumptions. The global probabilistic model for *Robust Reading* is detailed in [LMR04b]. Here we briefly illustrate one of its instantiations and concentrate on its basic assumptions, the experimental study and a comparison to the discriminative model.

At the heart of our unsupervised approach is a view on how documents are generated and how names (of different types) are "sprinkled" into them. In its most general form, our model assumes: (1) a joint distribution over entities, so that a document that mentions "President Kennedy" is more likely to mention "Oswald" or "White House" than "Roger Clemens"; (2) an "author" model, which makes sure that at least one mention of a name in a document is easily identifiable (after all, that is the author's goal), and then generates other mentions via (3) an appearance model, governing how mentions are transformed from the "representative" mention. Under these assumptions, the goal is to learn a model from a large corpus and use it to support *Robust Reading*. Given a collection of documents, learning proceeds in an *unsupervised* way; that is, the system is not told during training whether two mentions represent the same entity.

Both learning models assume the ability to recognize names and their type, using a named entity recognizer as a preprocessor.

Our experimental results are somewhat surprising; we show that the unsupervised approach can solve the problem accurately, giving accuracies (F_1) above 90%, and better than our discriminative classifier (obviously, with a lot more data).

After discussing some related work, the rest of this article first presents the experimental methodology in our evaluation, in order to present a more concrete instantiation of the problem at hand. It then describes the design of our pairwise name classifier, a comparison to other classifiers in the literature, and a discussion of clustering on top of a pairwise classifier. Finally, we present the generative model and compare the discriminative and generative approaches along several dimensions.

2 Previous Work

There is little previous work we know of that directly addresses the problem of Robust Reading in a principled way, but some related problems have been studied.

Robust Reading is, in principle, similar to tuple matching in databases — the problem of deciding whether multiple relational tuples from heterogenous sources refer to the same real-world entity [HS95a, TKM02, AMT04, DLLH03]. Most works in this area focus on well-structured database records and the problem of identifying the format variations of corresponding fields in different tuples. However, at the same time,

several works in this domain [CR02, HS95b, BM03, DLLH03] have started to look at this problem in semi-structured data sources. [PMM+02] considers the problem of identity uncertainty in the context of citation matching and suggests a relational probabilistic model, that is related to our relational appearance model. This work exhibits a need to perform tuple matching in a semi-structured data base with various textual fields. Other machine learning techniques [CR02, BM03, DLLH03] to this problem usually consider a pair of records and extract from the pair features that capture their similarity. The classifier is thus a parameterized similarity function that is trained given a set of annotated examples. That is, the pairs are labelled as matching or non-matching tags, and training serves to choose the parameters that optimize some loss function. Learning-based similarity metrics vary in their selection of features, hypotheses and learning algorithms. These approaches can be viewed as addressing some aspects of "the appearance model" – a lower level processing step in our approach.

From the natural language perspective, there has been a lot of work on the related problem of co-reference resolution (e.g., [SNL01, NC02, Keh02]). The goal is to link occurrences of noun phrases and pronouns, typically occurring in a close proximity, within a few sentences or a paragraph, based on their appearance and local context. Machine learning approaches to this problem first convert the local information into a set of features and then make use of a supervised learning approach to determine whether a given pronoun corresponds to a given noun phrase. Approaches differ in the algorithm used and features extracted.

A few works address some aspects of the Robust Reading problem with text data and study it in an across-document setting [MY03, BB98, MW03, GA04]. [MY03] considers one aspect of the problem – that of distinguishing occurrences of *identical* names in different documents, and only for one type of entity – *people*. That is, they consider the question of whether occurrences of "Jim Clark" in different documents refer to the same person. Their method makes use of "people-specific" information and may not be applied easily to other types of entities and other aspects of the robust reading problem. [BB98] builds a cross-document system based on an existing co-reference resolution tool, Camp. It extracts all the sentences containing an entity as a representation of the entity, and then applies a vector space model to compute the similarity between two such representations. Clustering is used subsequently to group entities in different documents into global co-reference chains. [MW03] uses a conditional model to address the problem of co-reference across documents. This work takes a more global view in that it defines a conditional probability distribution over partitions of mentions, give all observed mentions. The derived pairwise classification function that decides whether two names match is learned in a supervised manner, based on a maximum entropy model. However, this model does not incorporate contextual information and cannot resolve the ambiguity at the level we expect to.

3 Experimental Methodology

In our experimental study we evaluated different models on the problem of robust reading for three entity types – People (Peop), Locations (Loc) and Organizations (Org). The document segments shown in Figure 3 exemplify the preprocessed data given as

input to the evaluation. The learning approaches were evaluated on their ability to determine whether a pair of entities (within or across documents) actually correspond to the same real-world entity.

We collected $8,000$ names from randomly sampled 1998-2000 New York Times articles in the TREC corpus [Voo02]. These include about $4,000$ personal names[1], $2,000$ locations and $2,000$ organizations. The documents were annotated by a named entity tagger [2]. The annotation was verified and manually corrected if needed and each name mention was labelled with its corresponding entity by two annotators. Tests were done by averaging over five pairs of sets, each containing 600 names, that were randomly chosen from the $8,000$ names. For the discriminative approach, given a

Document 1: ***The Justice Department*** *has officially ended its inquiry into the assassinations of* ***President John F. Kennedy*** *and* ***Martin Luther King Jr.****, finding "no persuasive evidence" to support conspiracy theories, according to department documents.* ***The House Assassinations Committee*** *concluded in 1978 that* ***Kennedy*** *was "probably" assassinated as the result of a conspiracy involving a second gunman, a finding that broke from* ***the Warren Commission****'s belief that* ***Lee Harvey Oswald*** *acted alone in* ***Dallas*** *on Nov. 22, 1963.*

Document 2: ***David Kennedy*** *was born in* ***Leicester****,* ***England*** *in 1959.? ⋯* ***Kennedy*** *co-edited The New Poetry (Bloodaxe Books 1993), and is the author of New Relations: The Refashioning Of* ***British*** *Poetry 1980-1994 (Seren 1996).?*

Figure 1: Segments from two documents preprocessed by our named entity tagger as input to the Robust Reading Classifiers. Different types of entities are annotated with different colors. As shown, similar mentions within and across documents may sometimes correspond to the same entities and sometimes to different entities.

training set of 600 names (each of the 5 test sets corresponds to a different training set), we generated positive training examples using all co-referring pairs of names, and negative examples by randomly selecting pairs of names that do not refer to the same entity. Since most pairs of names do not co-refer, to avoid excessive negative examples in training sets, we use a ratio of $10:1$ between negative examples and positive examples. The probabilistic model is trained using a larger corpus.

The results in all the experiments in this paper are evaluated using the same test sets, except when comparing the clustering schemes. For a comparative evaluation, the outcomes of each approach on a test set of names are converted to a classification over all possible pairs of names (including non-matching pairs). Since most pairs are trivial negative examples, and the classification accuracy can always reach nearly 100%, the evaluation is done as follows. Only examples in the set M_p, those that are predicated to belong to the same entity (positive predictions) are used in the evaluation, and are compared with the set M_a of examples annotated as positive. The performance of an

[1] Honorifics and suffixes like "Jr." are considered part of a personal name.

[2] The named entity tagger was developed by the Cognitive Computation Group at UIUC. A demo of this tool is available at http://L2R.cs.uiuc.edu/∼cogcomp/eoh/ne.html.

approach is then evaluated by Precision and Recall, defined respectively as:

$$P = \frac{|M_p \bigcap M_a|}{|M_p|} \qquad R = \frac{|M_p \bigcap M_a|}{|M_a|},$$

and summarized by

$$F_1 = \frac{2P \cdot R}{P + R}.$$

Only F_1 values are shown and compared in this paper.

4 A Discriminative Model

A natural way to formalize the *Robust Reading* is as a pairwise classification problem: Find a function $f : N \times N \rightarrow \{0, 1\}$ which classifies two strings (representing entity mentions) in the name space N, as to whether they represent the same entity (1) or not (0). Most prior work in this line adopted fixed string similarity metrics and created the desired function by simply thresholding the similarity between two names. [CRF03] compared experimentally a variety of string similarity metrics on the task of matching entity names and found that, overall, the best-performing method is a hybrid scheme (SoftTFIDF) combining a TFIDF weighting scheme of tokens with the Jaro-Winkler string-distance scheme. Although this is a fixed scheme, the threshold used by the SoftTFIDF classifier is trained. [BM03] proposed a learning framework (Marlin) for improving entity matching using trainable measures of textual similarity. They compared a learned edit distance measure and a learned vector space based measure that employs a classifier (SVM) against fixed distance measures (but not the one mentioned above) and showed some improvements in performance.

We propose a learning approach, *LMR* [3], that focuses on representing a given pair of names using a collection of relational (string and token-level) and structural features. Over these we learn a linear classifier for each entity type using the SNoW (Sparse Network of Winnows [CCRR99]) learning architecture. A feature extractor[4] automatically extracts features in a data-driven way for each pair of names. Our decision is thus of the form:

$$f(n_1, n_2) = \arg \max_{c \in \{0,1\}} f^c(n_1, n_2) = \arg \max_{c \in \{0,1\}} \sum_{i=1}^{k} w_{i,c} f_i \qquad (1)$$

where $w_{i,c}$ is the weight of feature $f_i (1 \leq i \leq k)$ in the function $f^c(n_1, n_2)$.

4.1 Feature Extraction

An example is generated by extracting features from a pair of names. Two types of features are used: relational features representing mappings between tokens in the two names, and structural features, representing the structural transformations of tokens in one name into tokens of the other.

[3] Named after the initials of the authors' last names.

[4] We use FEX, a feature extractor tool available from http://L2R.cs.uiuc.edu/∼cogcomp/cc-software.html.

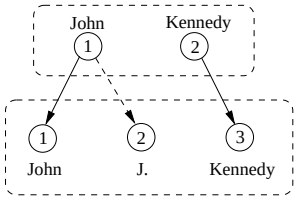

Figure 2: There are two possible features (Equality and Initial) for token 1 in the smaller partition but only the higher priority Equality feature is activated.

Each name is modelled as a partition in a bipartite graph, with each token in that name as a vertex (see Figure 2) and there is a solid directed edge between two tokens (from the vertex in the smaller partition to the vertex in the larger one) which activates a token-based feature for the two names. At most one token-based relational feature is extracted for each edge in the graph, by traversing a prioritized list of feature types until a feature is activated; if no active features are found, it goes to the next pair of tokens. This scheme guarantees that only the most important (expressive) feature is activated for each pair of tokens. An additional constraint is that each token in the smaller partition can only activate one feature. We define thirteen types of token-based features, shown in the priority order as described above:

- Honorific Equal: active if both tokens are honorifics and identical. An honorific is a title such as “Mr.”, “Mrs.”, “President”, “Senator” or “Professor”.
- Honorific Equivalence: active if both tokens are honorifics, not identical, but equivalent (“Professor”, “Prof.”).
- Honorific Mismatch : active for different honorifics.
- Equality: active if both tokens are identical.
- Case-Insensitive Equal: active if the tokens are case-insensitive equal.
- Nickname: active if tokens have a “nickname” relation (“Thomas” and “Tom”).
- Prefix Equality: active if the prefixes of both tokens are equal. The prefix is defined as substring starting at the beginning of a token and going until the first vowel (or second if the first letter is a vowel).
- Substring: active if one of the tokens is a substring of the other.
- Abbreviation: active if one of the tokens is an abbreviation of the other; ie “Corporation” and “Corp.”.
- Prefix Edit Distance: active if the prefixes of both tokens have an edit-distance of 1.
- Edit Distance: active if the tokens have an edit-distance of 1.

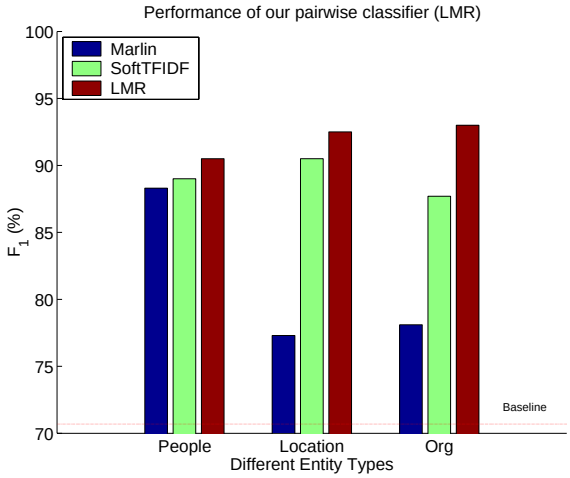

Figure 3: **Performance of Different Pairwise Classifiers.** Results are evaluated using the F_1 value and are averaged over five test sets of 600 names each, for each entity type. Our LMR classifier is compared with Marlin [BM03] and SoftTFIDF [CRF03]. The learned classifiers are trained using corresponding training sets with 600 names. The baseline performance in the experiment is 70.7% given by a classifier that predicts only identical names as positive examples, and it is averaged over the three entity types.

- Initial: active if one of the tokens is an initial of another; ie, "Paul" and "P.".
- Symbol Map: active if one token is a symbolic representative of the other ("and" and "&").

Relational features are not sufficient, since a non-matching pair of names could activate exactly the same set of features as a matching pair. Consider, for example, two names that are all the same except that one has an additional token. Our *structural* features were designed to distinguish between these cases. These features encode information on the relative order of tokens between the two names, by recording the location of the participating tokens in the partition. E.g., for the pairs ("John Kennedy","John Kennedy") and ("John Kennedy", "John Kennedy Davis"), the active relational features are identical; but, the first pair activates the structural features "$(1, 2)$" and "$(1, 2)$", while the second pair activates "$(1, 3)$" and "$(1, 2, \emptyset)$".

4.2 Experimental Comparison

Figure 3 presents the average F_1 for three different pairwise classifiers on the five test sets described in Sec.3. The LMR classifier outperforms the SoftTFIDF classifier and the Marlin classifier when trained and tested on the same data sets.

Figure 4 shows the contribution of different feature types to the performance of the LMR classifier. The *Baseline* classifier in this experiment only makes use of string-edit-distance features and "Equality" features. The *Token-Based* classifier uses all relational

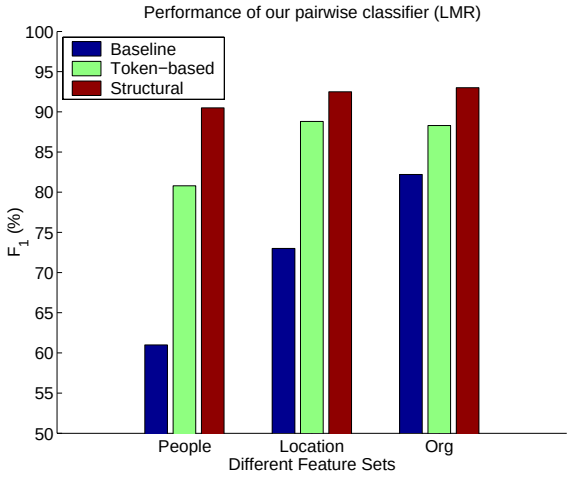

Figure 4: **The Contribution of Different Feature Sets.** The LMR classifier is trained with different feature sets using the five training sets. Results are evaluated using the F_1 value and are averaged over the five test sets for each entity type with 600 names in each of them. The *Baseline* classifier only uses string-edit-distance features and "Equality" features. The *Token-Based* classifier uses all relational token-based features while the *Structural* classifier uses, in addition, structural features.

token-based features while the *Structural* classifier uses, in addition, the structural features. Adding relational and structural features types is very significant, and more so to *People* due to a larger amount of overlapping tokens between entities.

4.3 Does Clustering Help ?

There is a long-held intuition that the performance of a pairwise classifier can be improved if it is used as a similarity metric and a global clustering is performed on top of it. Several works [CRF03, CR02, MW03] have thus applied clustering in similar tasks, using their pairwise classifiers as the metric. However, we show here that this may not be the case; we provide theoretical arguments as well as experimental evidence that show that global clustering applied on the pairwise classifier might in fact degrade its performance. Specifically, we show that while optimal clustering always helps to reduce the error of a pairwise classifier when there are two clusters (corresponding to two entities), in general, for $k > 2$ classes, this is not the case. We sketch these arguments below.

A typical clustering algorithm views data points $n \in N$ to be clustered as feature vectors $n = (f_1, f_2, \ldots, f_d)$ in a d-dimensional feature space. A data point n is in one of k classes $C = \{C_1, C_2, \ldots, C_k\}$. From a generative perspective, the observed data points $D = \{n_1, n_2, \ldots, n_t\}$ are sampled i.i.d. from a joint probability distribution P defined over tuples $(n, c) \in N \times C$ (P is a mixture of k models). A distance metric $dist(n_1, n_2)$ is used to quantify the distance between two points. Clustering algorithms

differ in how they approximate the hidden generative models given D.

In the following definitions we assume that $P = P_{N \times C}$ is a distribution over $N \times C$ and that $(n_1, c_1), (n_2, c_2) \in N \times C$ are sampled i.i.d according to it, with n_1, n_2 observed and c_1, c_2 hidden.

Definition 1 *The problem of Robust Reading is that of finding a function $f : N \times N \rightarrow \{0, 1\}$ which satisfies:*

$$f(n_1, n_2) = 1 \quad \textit{iff} \quad c_1 = c_2 \tag{2}$$

Definition 2 *Let $dist : N \times N \rightarrow \Re^+$ be a distance metric, and $T \geq 0$ is a constant threshold. The* pairwise classification *function f_p in this setting is defined by:*

$$f_p(n_1, n_2) = 1 \quad \textit{iff} \quad dist(n_1, n_2) \leq T. \tag{3}$$

The clustering based decision f_c is defined by:

$$f_c(n_1, n_2) = 1 \quad \textit{iff} \quad argmax_i Pr\{n_1|C_i\} = argmax_i Pr\{n_2|C_i\}. \tag{4}$$

Definition 3 *Define $I(n_1, n_2)$ to be 1 when $c_1 = c_2$ and 0 otherwise. The* error rate *of the function $f : N \times N \rightarrow \{0, 1\}$ is defined as:*

$$err(f) = E(Pr\{f(n_1, n_2) \neq I(n_1, n_2)\}) \tag{5}$$

where the expectation is taken over independent samples, according to $P_{N \times C}$, of pairs of points $\{(n_1, c_1), (n_2, c_2)\}$.

Theorem 1 ***(Proof Omitted)*** *Assume data is generated according to a uniform mixture of k Gaussians $G = \{g_1, g_2, \ldots, g_k\}$ with the same covariance matrix. Namely, a data point is generated by first choosing one of k models with probability $p_k = 1/k$, and then sampling according to the i-th Gaussian chosen. Suppose further that the clustering algorithms yields the correct k Gaussian distributions; then, $\forall$ threshold $T > 0$, if $k = 2$ then*

$$err(f_c) < err(f_p). \tag{6}$$

However, this doesn't hold in general for $k > 2$.

For $k > 2$, we can show cases where $err(f_p) < err(f_c)$. Specifically, this holds when the centers of the Gaussians are close, and $T \rightarrow 0$.

To study this issue experimentally, we designed and compared several clustering schemes for the *Robust Reading* task. These clustering approaches are designed based on the learned pairwise classifier LMR. Given the activation values of the classifier — the values output by the linear functions for the classes, we define a similarity metric (instead of a distance metric) as follows: Let p, n be the activation values for class 1 and class 0, respectively, for two names n_1 and n_2; then, $sim(n_1, n_2) = \frac{e^p}{e^p + e^n}$.

In our direct clustering approach, we cluster names from a collection of documents with regard to the entities they refer to. That is, entities are viewed as the hidden classes that generate the observed named entities in text. We have experimented with several clustering algorithms and show here the best performing one, a standard agglomerative

clustering algorithm based on complete-link. The basic idea of this algorithm is as follows: it first constructs a cluster for each name in the initial step. In the following iterations, these small clusters are merged together step by step until some condition is satisfied (for example, if there are only k clusters left). The two clusters with the maximum average similarity between their elements are merged in each step. The evaluation, presented in Figure 5, shows a degradation in the results relative to pairwise classification.

Although, as we show, clustering does not help when applied directly, we attempted to see if clustering can be helped by exploiting some structural properties of the domain. We split the set of documents into three groups, each containing documents from the same time period. We then cluster first names belonging to each group, then choose a representative for the names in each cluster and, hierarchically, cluster these representatives across groups into final clusters. The complete-link algorithm is applied again in each of the clustering stages. In this case (Hier (Date) – Hierarchically clustering according to Dates), the results are better than in direct clustering. We also performed a control experiment (Hier (Random)), in which we split the document set randomly into three sets of the same size; the deterioration in the results in this case indicates that the gain was due to exploiting the structure. The data set used here was slightly different from the one used in other experiments. It was created by randomly selecting names from documents of the years $1998 - 2000$, 600 names from each year and for each entity type. The $1,800$ names for each entity type were randomly split into equal training and test set. We trained the LMR pairwise classifier for each entity type using the corresponding labeled training set and clustered the test set with LMR as a similarity metric.

One reason for the lack of gain from clustering is the fact that the pairwise classification function learned here is local – without using any information except for the names themselves – and thus suffers from noise. This is because, in training, each pair of names is annotated with regard to the entities they refer to rather than their similarity in writing. Specifically, identical names might be labeled as negative examples, since they correspond to different entities, and vice versa. Our conclusion, reinforced by the slight improvement we got when we started to exploit structure in the hierarchical clustering experiment, is that the Robust Reading problem necessitates better exploitation of global and structural aspects of data. Our next model was design to address this issue.

5 A Generative Model for Robust Reading

Motivated by the above observation, we describe next a generative model for *Robust Reading*, designed to exploit documents' structure and assumptions on how they are generated. The details of this model are described in [LMR04b]. Here we briefly describe one instantiation of the model (Model II there), focusing on discussing the key assumptions and their advantages, and on its experimental study and a comparison to the discriminative model.

We define a probability distribution over documents $d = \{E^d, R^d, M^d\}$, by describing how documents are being generated. In its most general form the model has

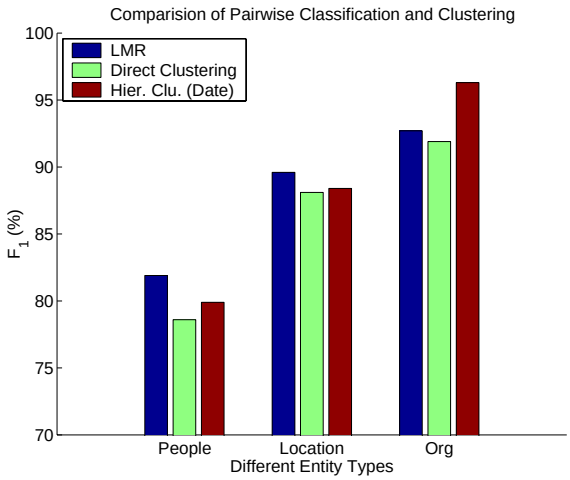

Figure 5: **Best Performance of Different Clustering Approaches** (Various parameter settings, including different numbers of clusters were experimented with in direct clustering and the hierarchical clustering.) 'LMR' represents our pairwise classifier. It is compared with different clustering schemes, based on it as a similairty metric. Results are evaluated using F_1 values. The test set has 900 names for each entity type.

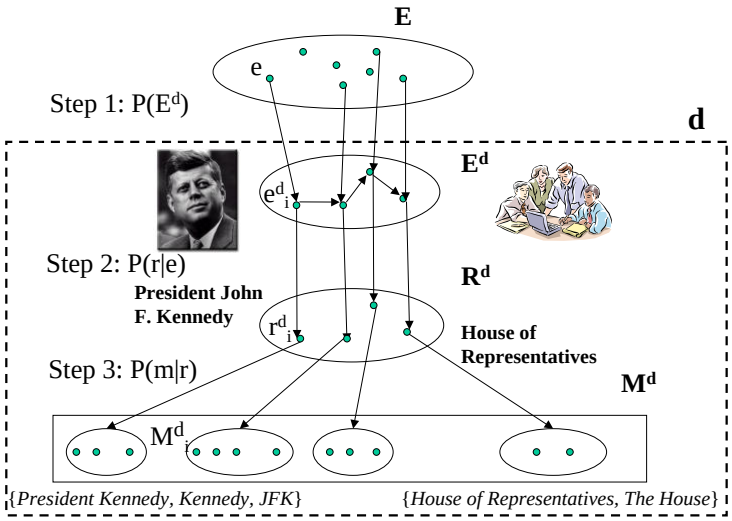

Figure 6: Generating a document. Real-word entities are represented by pictures. Mentions in a document d are generated in three steps according to underlying probabilities.

the following three components: (1) A joint probability distribution $P(E^d)$ that governs how entities (of different types) are distributed into a document and reflects their co-occurrence dependencies. (When initializing the model described here, we assume that entities are chosen independently of each other.) (2) An "author" model, that makes sure that at least one mention of an entity in a document — a representative r^d, is easily identifiable and other mentions are generated from it. The number of entities in a document, $size(E^d)$, and the number of mentions of each entity, $size(M_i^d)$, are assumed in the current evaluation to be distributed uniformly over a small plausible range so that they can be ignored in later inference. (3) The *appearance probability* of a name generated (transformed) from its representative is modelled as a product distribution over relational transformations of attribute values. This model captures the similarity between appearances of two names. In the current evaluation the same appearance model is used to calculate both the probability $P(r|e)$ that generates a representative r given an entity e and the probability $P(m|r)$ that generates a mention m given a representative r. Attribute transformations are relational, in the sense that the distribution is over transformation types and independent of the specific names.

Figure 6 depicts the generation process of a document d. Thus, in order to generate a document d, after picking $size(E^d)$ and $\{size(M_1^d), size(M_2^d), \ldots\}$, each entity e_i^d is selected into d independently of others, according to $P(e_i^d)$. Next, the representative r_i^d for each entity e_i^d is selected according to $P(r_i^d|e_i^d)$ and for each representative the actual mentions are selected independently according to $P(m_j^d|r_j^d)$. Assuming independence between M^d and E^d given R^d, the probability distribution over documents is therefore

$$\begin{aligned} P(d) &= P(E^d, R^d, M^d) = P(E^d)P(R^d|E^d)P(M^d|R^d) \\ &\sim \prod_{i=1}^{|E^d|}[P(e_i^d)P(r_i^d|e_i^d)] \prod_{(r_j^d, m_j^d)} P(m_j^d|r_j^d) \end{aligned}$$

after we ignore the size components.

Given a mention m in a document d (M^d is the set of observed mentions in d), the key inference problem is to determine the most likely entity e_m^* that corresponds to it. This is done by computing:

$$E^d = argmax_{E' \subseteq E} P(E', R^d, M^d|\theta), \tag{7}$$

where θ is the learned model's parameters. This gives the assignment of the most likely entity e_m^* for m. And the probability of the document collection D is: $P(D) = \prod_{d \in D} P(d)$.

5.1 Learning the Models

Confined by the labor of annotating data, we learn the probabilistic models in an unsupervised way given a collection of documents; that is, the system is not told during training whether two mentions represent the same entity. A greedy search algorithm modified from the standard EM algorithm (we call it Truncated EM algorithm) is adopted here to avoid complex computation.

Given a set of documents D to be studied and the observed mentions M^d in each document, this algorithm iteratively updates the model parameter θ (several underlying probabilistic distributions described before) and the structure (that is, E^d and R^d) of each document d. Different from the standard EM algorithm, in the E-step, it seeks the most likely E^d and R^d for each document rather than the expected assignment.

Truncated EM Algorithm

The basic framework of the Truncated EM algorithm is as follows:

1. In the initial (I-) step, an initial (E_0^d, R_0^d) is assigned to each document d by an initialization algorithm. After this step, we can assume that the documents are annotated with $D_0 = \{(E_0^d, R_0^d, M^d)\}$.
2. In the M-step, we seek the model parameter θ_{t+1} that maximizes $P(D_t|\theta)$. Given the "labels" supplied in the previous I- or E-step, this amounts to the maximum likelihood estimation (to be described later in this section).
3. In the E-step, we seek (E_{t+1}^d, R_{t+1}^d) for each document d that maximizes $P(D_{t+1}|\theta_{t+1})$ where $D_{t+1} = \{(E_{t+1}^d, R_{t+1}^d, M^d)\}$. It's the same inference problem as in Equation 7.
4. Stoping Criterion: If no increase is achieved over $P(D_t|\theta_t)$, the algorithm exits. Otherwise the algorithm will iterate over the M-step and E-step.

It usually takes 3-10 iterations before the algorithms stop in our experiments.

Initialization

The purpose of the initial step is to acquire an initial guess of document structures and the set of entities E in a closed collection of documents D. The hope is to find all entities without loss so duplicate entities are allowed. For all the models, we use the same algorithm. A local clustering is performed to group mentions inside each document: simple heuristics are applied to calculating the initial similarity between mentions[5], and pairs of mentions with similarity above a threshold are then clustered together. The first mention in each group is chosen as the representative and an entity having the same writing with the representative is created for each cluster. To goal we try to achieve with this simple initialization is high precision, even if the recall is low [6]. For all the models, the set of entities created in different documents become the global entity set E in the following M- and E-steps.

[5]One example of a heuristics is: *If mention n_1 is a substring of mention n_2, then their similarity is* 0.8.
[6]Note that the performance of the initialization algorithm is 97.3% precision and 10.1% recall.

Estimating the Model Parameters

In the learning process, assuming documents have already been annotated $D = \{(e, r, m)\}_1^n$ from previous I- or E-step, several underlying probability distributions of the relaxed models are estimated by maximum likelihood estimation in each M-step. The model parameters include a set of prior probabilities for entities P_E and the appearance probabilities $P_{W|W}$ of each name in the name space W being transformed from another.

- The prior distribution P_E is modelled as a multi-nomial distribution. Given a set of labelled entity-mention pairs $\{(e_i, m_i)\}_1^n$,

$$P(e) = \frac{freq(e)}{n}$$

where $freq(e)$ denotes the number of pairs containing entity e.

- Appearance probability, the probability of one name being transformed from another, denoted as $P(n_2|n_1)$ $(n_1, n_2 \in W)$, is modelled as a product of the transformation probabilities over attribute values. The transformation probability for each attribute is further modelled as a multi-nomial distribution over a set of predetermined transformation types: $TT = \{copy, missing, typical, non-typical\}$[7].

Suppose $n_1 = (a_1 = v_1, a_2 = v_2, ..., a_p = v_p)$ and $n_2 = (a_1 = v'_1, a_2 = v'_2, ..., a_p = v'_p)$ are two names belonging to the same entity type, the transformation probabilities $P_{M|R}$, $P_{R|E}$ and $P_{M|E}$, are all modelled as a product distribution (naive Bayes) over attributes:

$$P(n_2|n_1) = \Pi_{k=1}^p P(v'_k|v_k).$$

We manually collected typical and non-typical transformations for attributes such as *titles*, *first names*, *last names*, *organizations* and *locations* from multiple sources such as U.S. government census and online dictionaries. For other attributes like *gender*, only **copy** transformation is allowed. The maximum likelihood estimation of the transformation probability $P(t, k)$ $(t \in TT, a_k \in A)$ from annotated representative-mention pairs $\{(r, m)\}_1^n$ is:

$$P(t, k) = \frac{freq(r, m) : v_k^r \rightarrow_t v_k^m}{n} \quad (8)$$

$v_k^r \rightarrow_t v_k^m$ denotes the transformation from attribute a_k of r to that of m is of type t. Simple smoothing is performed here for unseen transformations.

5.2 A Comparison between the Models

We trained the generative model in an unsupervised way with all $8,000$ names. The somewhat surprising results are shown in Figure 7. The generative model outperformed the supervised classifier for People and Organizations. That is, by incorporating a lot of unannotated data, the unsupervised learning could do better. To understand this better,

[7] **copy** denotes v'_k is exactly the same as v_k; **missing** denotes "missing value" for v'_k; **typical** denotes v'_k is a typical variation of v_k, for example, "Prof." for "Professor", "Andy" for "Andrew"; **non-typical** denotes a non-typical transformation.

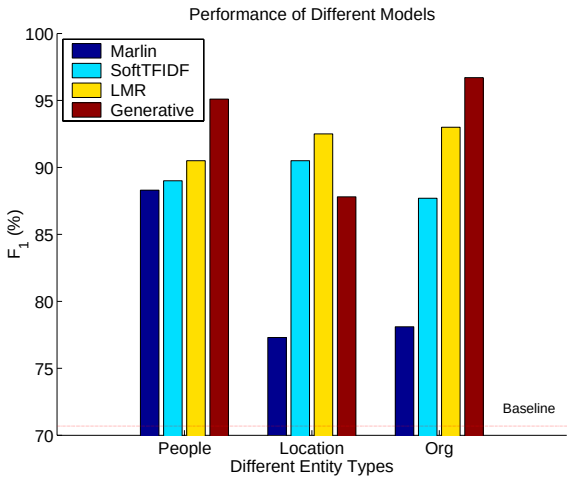

Figure 7: **Discriminative and Generative Models** Results are evaluated by the average F_1 values over the five test sets for each entity type. "Marlin", "SoftTFIDF" and "LMR" are the three pairwise classifiers; "Generative" is the generative model. The baseline performance is 70.7% given by a classifier that predicts only identical names as positive examples and it is averaged over the three entity types.

and to compare our discriminative and generative approaches further, we addressed the following two issues:

Learning protocol: A supervised learning approach is trained on a training corpus, and tested on a different one resulting, necessarily, in some degradation in performance. An unsupervised method learns directly on the target corpus. This, as we show, can be significant. In a second experiment, in which we do not train the generative model on names it will see in the test set, results clearly degrade (Table 1).

Structural assumptions: Our generative model benefits from the structural assumptions made in designing the model. For instance, the document structural information prevents the model from disambiguating mentions from different documents directly. This is done via the representatives – yielding a more robust comparison, since representatives are typically the full names of the entities. We exhibit this by evaluating a fairly weak initialization of the model, and showing that, nevertheless, this results in a *Robust Reading* model with respectable results. Table 2 shows that after initializing the model parameters with the heuristics used in the EM-like algorithm, and without further training (but with the inference of the probabilistic model), the generative model can perform reasonably well. Note, however, that the structural assumptions in the generative model did not help locations very much as shown in Figure 7. The reason is that entities of this type are relatively easy to disambiguate even without structural information, since those that correspond to the same entity typically have similar writings.

$F_1(\%)$	LMR	Generative (all)	Generative (unseen data)
Peop	90.5	95.1	88.8
Loc	92.5	87.8	78.2
Org	93.0	96.7	84.2

Table 1: **Results of Different Learning Protocols for the Generative Model**. The table shows the results of our supervised classifier (LMR) trained with 600 names, the generative model trained with all the $8,000$ names and the generative model trained with the part of $8,000$ names not used in the corresponding test set. Results are evaluated and averaged over five test sets for each entity type.

$F_1(\%)$	Generative	Initial
Peop	95.1	84.1
Loc	87.8	85.0
Org	96.7	92.1

Table 2: **Performance of Simple Initialization**. "Generative" – the generative model learned in a normal way. "Initial" – the parameters of the generative model initialized using some simple heuristics and used to cluster names. Results are evaluated by the average F_1 values over the five test sets for each entity type.

6 Conclusion

While semantic integration of structured information has been widely studied, little attention has been paid to a similar problem in unstructured and semi-structured data. This paper describes one of the first efforts towards sematic integration in unstructured textual data, providing a promising perspective on the integration of structured databases with unstructured or semi-structured information.

This paper presents two learning approaches to the "robust reading" problem — the problem of cross-document identification and tracing of names of different types, overcoming their ambiguous appearance in the text. In addition to a standard modelling of the problem as a classification task, we developed a model that aims at better exploiting the natural generation process of documents and the process of how names are "sprinkled" into them, taking into account dependencies among entities across types and an "author" model. We have shown that both models gain significantly from incorporating structural and relational information – features in the classification model; coherent data splits for clustering and the natural generation process in the the probabilistic model.

The *robust reading* problem is a very significant barrier in any effort towards supporting intelligent information access to texts and, specifically, on our way toward supporting semantic integration of structured information with unstructured information. The reliable and accurate results we show are thus very encouraging. Also important is the fact that our unsupervised model performs so well, since the availability of annotated data is, in many cases, a significant obstacle to good performance in semantic integration.

In addition to further studies of the discriminative model, including going beyond the current noisy supervision (given at a global annotation level, although learning is done locally), exploring how much data is needed for a supervised model to perform as well as the unsupervised model, and whether the initialization of the unsupervised model can gain from supervision, there are several other critical issues we would like to address from the robust reading perspective. These include (1) incorporating more contextual information (like time and place) related to the target entities, both to support a better model and to allow temporal tracing of entities; (2) studying an incremental approach to learning the model and (3) developing a unified technique based on this work, that can address the semantic integration problem of both databases and texts and can integrate structured and unstructured information.

Acknowledgment

We are grateful to AnHai Doan for numerous useful comments on this article.
This research is supported by NSF grants IIS-9801638 and ITR IIS-0085836, an ONR MURI Award and an equipment donation from AMD.

References

[AMT04] P. Andritsos, R. J. Miller, and P. Tsaparas. Information-theoretic tools for mining database structure from large data sets. In *Proceedings of the ACM SIGMOD Conference*, 2004.

[BB98] A. Bagga and B. Baldwin. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of COLING*, pages 79–85. Association for Computational Linguistics, 1998.

[BM03] M. Bilenko and R. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of KDD*, 2003.

[CCRR99] A. Carlson, C. Cumby, J. Rosen, and D. Roth. The SNoW learning architecture. Technical Report UIUCDCS-R-99-2101, UIUC Computer Science Department, May 1999.

[CR02] W. Cohen and J. Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *Proceedings of KDD*, 2002.

[CRF03] W. Cohen, P. Ravikumar, and S. Fienberg. A comparison of string metrics for name-matching tasks. In *IIWeb Workshop 2003*, 2003.

[DDH01] A. Doan, P. Domingos, and A. Halevy. Reconciling Schemas of Disparate Data Sources: A Machine Learning Approach. In *Proceedings of the ACM SIGMOD Conference*, 2001.

[DLD+04] R. Dhamankar, Y. Lee, A. Doan, A. Halevy, and P. Domingos. iMAP: Discovering complex matches between database schemas. In *Proceedings of the ACM SIGMOD Conference (SIGMOD)*, 2004.

[DLLH03] A. Doan, Y. Lu, Y. Lee, and J. Han. Profile-based object matching for information integration. *IEEE Intelligent Systems*, 18(5):54–59, 2003.

[GA04] C. Gooi and J. Allan. Cross-document coreference on a large scale corpus. In *Proceedings of HLT-NAACL*, 2004.

[HS95a] M. Hernandez and S. Stolfo. The merge/purge problem for large databases. In *SIGMOD Conference*, pages 127–138, 1995.

[HS95b] M. Hernandez and S. Stolfo. The merge/purge problem for large databases. In *Proceedings of SIGMOD*, 1995.

[Keh02] A. Kehler. *Coherence, Reference, and the Theory of Grammar*. CSLI Publications, 2002.

[LMR04a] X. Li, P. Morie, and D. Roth. Identification and tracing of ambiguous names: Discriminative and generative approaches. In *Proceedings of AAAI*, 2004.

[LMR04b] X. Li, P. Morie, and D. Roth. Robust reading: Identification and tracing of ambiguous names. In *Proceedings of HLT-NAACL*, 2004.

[MBR01] J. Madhavan, P.A. Bernstein, and E. Rahm. Generic schema matching with cupid. In *Proceedings of the International Conference on Very Large Databases (VLDB)*, 2001.

[MMGR02] S. Melnik, H. Molina-Garcia, and E. Rahm. Similarity Flooding: A Versatile Graph Matching Algorithm. In *Proceedings of the International Conference on Data Engineering (ICDE)*, 2002.

[MW03] A. McCallum and B. Wellner. Toward conditional models of identity uncertainty with application to proper noun coreference. In *IJCAI Workshop on Information Integration on the Web*, 2003.

[MY03] G. Mann and D. Yarowsky. Unsupervised personal name disambiguation. In *Computational Natural Language Learning*, 2003.

[MZ98] T. Milo and S. Zohar. Using schema matching to simplify heterogeneous data translation. In *Proceedings of VLDB*, 1998.

[NC02] V. Ng and C. Cardie. Improving machine learning approaches to coreference resolution. In *Proceedings of the Annual Meeting of the ACL*, 2002.

[NHT+02] F. Neumann, CT. Ho, X. Tian, L. Haas, and N. Meggido. Attribute classification using feature analysis. In *Proceedings of the Int. Conference on Data Engineering (ICDE)*, 2002.

[OS99] A. Ouksel and A. P. Seth. Special issue on semantic interoperability in global information systems. *SIGMOD Record*, 28(1), 1999.

[PMM+02] H. Pasula, B. Marthi, B. Milch, S. Russell, and I. Shpitser. Identity uncertainty and citation matching. In *Proceedings of NIPS*, 2002.

[RB01] E. Rahm and P.A. Bernstein. On matching schemas automatically. *VLDB Journal*, 10(4), 2001.

[SNL01] W. Soon, H. Ng, and D. Lim. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics (Special Issue on Computational Anaphora Resolution)*, 27:521–544, 2001.

[TKM02] S. Tejada, C. Knoblock, and S. Minton. Learning domain-independent string transformation weights for high accuracy object identification. In *Proceedings of the 8th SIGKDD Int. Conference (KDD-2002)*, 2002.

[Voo02] E. Voorhees. Overview of the TREC-2002 question answering track. In *Proceedings of the 11th Text Retrieval Conference, NIST*, pages 115–123, 2002.