Identification and Tracing of Ambiguous Names: Discriminative and Generative Approaches

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

Abstract

A given entity – representing a person, a location or an organization – may be mentioned in text in multiple, ambiguous ways. Understanding natural language requires identifying whether different mentions of a name, within and across documents, represent the same entity.

We present two machine learning approaches to this problem, which we call the "Robust Reading" problem. Our first approach is a discriminative approach, trained in a supervised way. Our second approach is a 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. Our extensive experiments exhibit the contribution of relational and structural features and, somewhat surprisingly, that the assumptions made within our generative model are strong enough to yield a very powerful approach, that performs better than a supervised approach with limited supervised information.

Introduction

Reading and understanding text requires the ability to disambiguate at several levels, abstracting away details and using background knowledge in a variety of ways. One of the difficulties that humans resolve instantaneously and unconsciously is that of reading names. Most names of people, locations, organizations and others, have multiple writings that are being used freely within and across documents.

The variability in writing a given concept, along with the fact that different concepts may have very similar writings, poses a significant challenge to progress in natural language processing. Consider, for example, an open domain question answering system (Voorhees 2002) that attempts, given a question like: "When was President Kennedy born?" to search a large collection of articles in order to pinpoint the concise answer: "on May 29, 1917." 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. Ad hoc solutions to this problem, as we show, fail to provide a reliable and accurate solution.

This paper presents two learning approaches to the problem of *Robust Reading*, compares them to existing approaches and shows 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 nature of the use 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 (Cohen, Ravikumar, & Fienberg 2003; Bilenko & Mooney 2003) – allowing us to compare our results with these. This is a standard pairwise classification task, under a supervised learning protocol; 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 by performing some global optimization, with the above mentioned pairwise classifier as the similarity metric. The results of these attempts were not conclusive and we provide some explanation for why that is. We prove that, assuming optimal clustering, this approach reduces the error of a pairwise classifier in the case of two classes (corresponding to two entities); however, in the more general

case, when the number of entities (classes) is greater than 2, global optimization mechanisms could be worse than pairwise classification. Our experiments concur with this. However, as we show, if one first splits the data in some coherent way – e.g., to groups of documents originated at about the same time period – this can aid clustering significantly.

This observation motivates our second model. We developed a global probabilistic model for *Robust Reading*, detailed in (Li, Morie, & Roth 2004). Here we briefly illustrate one of its instantiations and concentrate on its experimental study and a comparison to the discriminative models. At the heart of this 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, that makes sure that at least one mention of a name in a document is easily identifiable (after all, that's the author's goal), and then generates other mentions via (3) an appearance model, governing how mentions are transformed from the "representative" mention.

Our goal is to learn the model from a large corpus and use it to support *Robust Reading*. Given a collection of documents we learn the model in an *unsupervised* way; that is, the system is not told during training whether two mentions represent the same entity. We only assume, as in the discriminative model above, the ability to recognize names, using a named entity recognizer run 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). In the rest of the paper, we further study the discriminative and generative model in an attempt to provide explanations to the success of the unsupervised approach, and shed light on the problem of Robust Reading and hopefully, many related problems. Our study addresses two issues: (1) Learning Protocol: A supervised learning approach is trained on a training corpus, and tested on a different one, necessarily resulting in some degradation in performance. An unsupervised method learns directly on the target corpus. The difference, as we show, is significant. (2) Structural assumptions: we show that our generative model benefits from the assumptions made in designing it. This is shown by evaluating a fairly weak initialization of model parameters, which still results in a model for *Robust Reading* with respectable results. We leave open the question of how much data is required for a supervised learning methods before it can match the results of our supervised learner.

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. From the natural language perspective, there has been a lot of work on the related problem of coreference resolution (Soon, Ng, & Lim 2001; Ng & Cardie 2003; Kehler 2002) – which aims at linking occurrences of noun phrases and pronouns, typically occurring in a close proximity within a short document or a paragraph, based on their appearance and local context. In the context of databases, several works have looked at the problem of record linkage - recognizing duplicate records in a database (Cohen & Richman 2002; Hernandez & Stolfo 1995; Bilenko & Mooney 2003; Doan *et al.* 2003). Specifically, (Pasula *et al.* 2002) considers the problem of identity uncertainty in the context of citation matching and suggests a probabilistic model for that. One of the few works we are aware of that works directly with text data and across documents, is (Mann & Yarowsky 2003), which considers one aspect of the problem – that of distinguishing occurrences of *identical* names in different documents, and only of *people* (e.g., do occurrences of "Jim Clark" in different documents refer to the same person?).

The rest of this paper first presents the experimental methodology and data used in our evaluation; it then presents 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.

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). We collected $8,000$ names from randomly sampled 1998-2000 New York Times articles in the TREC corpus (Voorhees 2002) (about $4,000$ personal names[1], $2,000$ locations and $2,000$ organizations). The documents were annotated by a named entity tagger for these entity types. The annotation was then manually corrected and each name mention was labelled with its corresponding entity by two annotators. 5 pairs of training and test sets with 600 names in each data set are constructed for each of the three entity types by randomly picking from the $8,000$ names.

For the discriminative classifier, given a training set, we generated positive training examples using all co-referring pairs of names, and negative examples by randomly selecting (10 times that number of) pairs of names that do not refer to the same entity. 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%, only the set M_p of examples that are predicated to belong to the same entity (positive predictions) are used in evaluation, and are compared with the set M_a of examples annotated as positive. The performance of an approach is then evaluated by Precision $P = \frac{|M_p \bigcap M_a|}{|M_p|}$, Recall $R = \frac{|M_p \bigcap M_a|}{|M_a|}$, and $F_1 = \frac{2PR}{P+R}$. Only F_1 values are shown and compared in this paper.

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

A Discriminative Model

The *Robust Reading* problem can be formalized as pairwise classification: Find a function $f : N \times N \rightarrow \{0, 1\}$ which classifies two strings (representing entity writings) in the name space N, as to whether they represent the same entity (0) or not (1). Most prior work adopted fixed string similarity metrics and created the desired function by simply thresholding the similarity between two names. (Cohen, Ravikumar, & Fienberg 2003) 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. (Bilenko & Mooney 2003) 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) with fixed measures (but not the ones mentioned above) and showed some improvements in performance.

We propose a learning approach, *LMR*, 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, for each entity type, a linear classifier, that we train using the SNoW (Sparse Network of Winnows (Carlson *et al.* 1999)) learning architecture. A feature extractor automatically extracts active features in a data-driven way for each pair of names. Our decision is thus of the form:

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

where $w_{i,l}$ is the weight of feature f_i in the function f^l.

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.

Each name is modelled as a partition in a bipartite graph, with each token in that name as a vertex (see Fig. 1). 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 use 13 types of token-based features. Some of the interesting types are:

- Honorific Equal: active if both tokens are honorifics and equal. An honorific is a title such as "Mr.", "Mrs.", "President", "Senator" or "Professor".
- Honorific Equivalence: active if both tokens are honorifics, not equal, but equivalent ("Professor", "Prof.").
- Honorific Mismatch : active for different honorifics.
- NickName: active if tokens have a "nickname" relation ("Thomas" and "Tom").

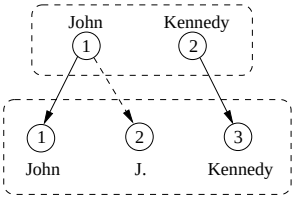

Figure 1: There are two possible features for token 1 in the smaller partition but only the higher priority Equality feature is activated.

$F_1(\%)$	Marlin	SoftTFIDF	LMR
Peop	88.3	89.0	**90.5**
Loc	77.3	90.5	**92.5**
Org	78.1	87.7	**93.0**

Table 1: **Performance of Different Pairwise Classifiers in an Identical Setting.** Results are evaluated in the average F_1 value over the 5 test sets for each entity type with 600 names in each of them. The columns are different classifiers and LMR is our classifier. The learned classifiers are trained using corresponding training sets with 600 names.

- Equality: both names are equal.
- Edit Distance: the tokens have an edit-distance of 1.
- Symbol Map: 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 token-based features as a matching pair. Consider, for example, two names that match exactly and the same two names except that one has an additional token. Our *structural* features were designed to avoid this phenomenon. 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 "$(1, \emptyset, 2)$" and "$(1, 3)$", while the second pair activates "$(1, 2)$" and "$(1, 2)$".

Experimental Comparison

Table 1 presents the average F_1 using three different pairwise classifiers on the 5 test sets we described before. The best results of each classifier using a uniform threshold for each entity type are shown here. The LMR classifier outperforms the non-learning SoftTFIDF classifier and the learned classifier (Marlin), under identical conditions.

Table 2 shows the contribution of our feature types to the performance of LMR. The *Baseline* classifier only uses string-edit-distance features and "Equality" features. The

$F_1(\%)$	Baseline	Token-Based	Structural
Peop	61.0	80.8	90.5
Loc	73.0	88.8	92.5
Org	82.2	88.3	93.0

Table 2: **Contribution of Different Feature Sets** The LMR classifier is trained with different feature sets using the 5 training sets. Results are evaluated in the average F_1 value over the 5 test sets for each entity type with 600 names in each of them.

Token-Based classifier uses all relational 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 the larger amount of overlapping strings in different entities.

Does Clustering Help ?

There is a long-held intuition that using clustering on top of a pairwise classifier might reduce its error; several works (Cohen, Ravikumar, & Fienberg 2003; McCallum & Wellner 2003) have thus applied clustering in similar tasks. However, we prove here 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 classes and $k > 2$, it is not the case.

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 view, the observed data points $D = \{n_1, n_2, \ldots, n_t\}$ are sampled iid 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 difference between two points. Clustering algorithms differ in how they approximate the hidden generative models given D.

Definition 1 *Given two i.i.d sampled data points (n_1, c_1) and $(n_2, c_2) \in N \times C$ with only n_1, n_2 observed and c_1, c_2 hidden, and $P_{N\times C}$, the problem of Robust Reading is to find a function $f : N \times N \rightarrow \{0, 1\}$ which satisfies:*

$$f(n_1, n_2) = 0 \quad \textit{iff} \quad c_1 = c_2 \tag{2}$$

Definition 2 *The pairwise classification in this setting is:*

$$f_p(n_1, n_2) = 0 \quad \equiv \quad dist(n_1, n_2) \leq T \tag{3}$$

where $T \geq 0$ is a threshold. The clustering based decision:

$$f_c(n_1, n_2) = 0 \;\equiv\; argmax_i p(n_1|C_i) = argmax_i p(n_2|C_i) \tag{4}$$

Definition 3 *The error rate of the pair function f is:*

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

where $I(n_1, n_2) = 0$ iff $c_1 = c_2; 1,$ otherwise.

Lemma 1 *Assume data is generated according to a mixture of k Gaussians $G = \{g_1, g_2, \ldots, g_k\}$ with the same covariance (i.e., a data point is generated by first choosing one of k models with probability p_k ($\sum_i p_k = 1$), and then sampling according to the i-th Gaussian chosen.), and suppose we find the correct k Gaussian distributions after clustering, then for $k = 2$ and $\forall T$,*

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

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

Proof (sketch):[2] It's easy to see that the probability density over tuples in $N \times C$ is $f(n, c_i) = \frac{1}{k} * g_i(n)$. The error rates $err(f_c)$ and $err(f_p)$ can be computed using the density function. Thus, for $k = 2$, we get $err(f_c) = \frac{1}{2} - \frac{1}{2}A^2$, where $A = \oint_R [g_1(X) - g_2(X)]dX$, and R is the area in the feature space that satisfies $g_2(X) > g_1(X)$ ($X \in N$). R is a half space here. We also have:

$$err(f_p) = \frac{1}{2} - \frac{1}{2} \oint_R [g_2(X_1) - g_1(X_1)]dX_1 \times \oint_{M(X_1,T)} [g_2(X_2) - g_1(X_2)]dX_2 \; > \; err(f_c)$$

where $M(X_1, T)$ is the sphere area in the feature space whose point X satisfies $dist(X_1, X) \leq T$. This is so since $\oint_{M(X_1,T)} [g_2(X_2) - g_1(X_2)]dX_2 < \oint_R [g_2(X_2) - g_1(X_2)]dX_2$.

For $k > 2$, we can compute $err(f_c)$ and $err(f_p)$ in a similar way and compare them. We found that in this case, each can be smaller than the other in different cases. Specifically, when $\sum_{i=1}^{k} g_i(n) > 2 * g_j(n)$ for all j and n, and $T \rightarrow 0$, we have $err(f_c) > err(f_p)$.

To study this issue experimentally, we designed and compared several clustering schemes for the *Robust Reading* task[3]. In a Direct clustering approach, we cluster named entities from a collection of documents with regard to the target entities, all at once. That is, the entities are viewed as the hidden classes that generate the observed named entities in text. The clustering algorithm used is a standard agglomerative clustering algorithm based on complete-link (which performed better than other clustering methods we attempted). The evaluation, column 2 of Table 3, shows a degradation in the results relative to pairwise classification.

Although, in general, clustering does not help, 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 originated at about the same time period, and clustered again. In this case (columns Hier (Date) – Hierarchically clustering according to Dates), the results are significantly better than Direct clustering. We then performed a control experiment (Hier (Random)), in which we split the document set randomly into three sets of the same sizes as before; the deterioration in the results in this case indicate that exploiting the structure is significant.

The data set used here was different from the one used in other experiments. They were created by randomly selecting 600 names for each entity type from the documents of years $1998 - 2000$. The 1800 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 cluster the test set with LMR evaluated as a similarity metric.

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 (Li, Morie,

[2] We sketch only for the case of a uniform Gaussian mixture; this can be relaxed.

[3] Given the activation values of the classifier, we define the distance metric as follows. Let p, n be the positive and negative activation values, respectively, for two names n_1 and n_2; then, $dist(n_1, n_2) = \frac{e^n}{e^p + e^n}$.

F_1(%)	LMR	Direct	Hier. (Date)	Hier. (Random)
Peop	81.9	78.6	**79.9**	76.7
Loc	89.6	88.1	**88.4**	88.8
Org	92.7	91.9	**96.3**	92.2

Table 3: **Best Performance of Different Clustering Approaches** (Various parameter settings, including different numbers of clusters were experimented with in Direct clustering.) 'LMR' represents our pairwise classifier. The other three columns are different clustering schemes. Results are evaluated by the F_1 values using the best threshold in experiments. The test set has 900 names for

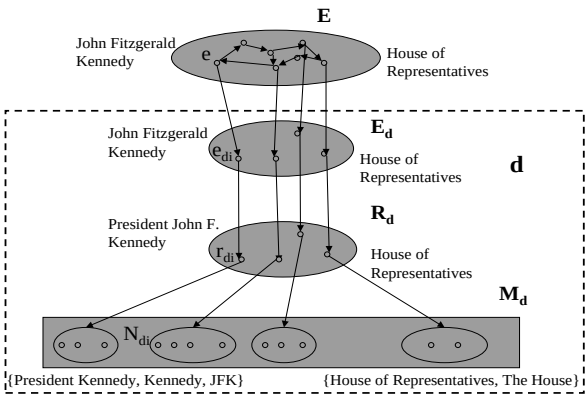

Figure 2: Generating a document

& Roth 2004). Here we briefly describe one instantiation of the model (Model II there) and concentrate on its experimental study and a comparison to the discriminative models.

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 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) One mention of a name 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 in E^d, $size(M_i^d)$, 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. Fig. 2 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

F_1(%)	Marlin	SoftTFIDF	LMR	Generative
Peop	88.3	89.0	90.5	**95.1**
Loc	77.3	90.5	92.5	**87.8**
Org	78.1	87.7	93.0	**96.7**

Table 4: **Discriminative and Generative Models** Results are evaluated by the average F_1 values over the 5 test sets for each entity type. "Marlin", "SoftTFIDF" and "LMR" are the three pairwise classifiers; "Generative" is the generative model.

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 5: **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 5 test sets for each entity type.

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 conditional independency 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:

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

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: $P(D) = \prod_{d \in D} P(d)$.

The model parameters are learned in an unsupervised way given a collection of documents: that is, the information whether two mentions represent the same entity is unavailable during training. A greedy search algorithm modified after the standard EM algorithm is adopted here to simplify the 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

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

Table 6: **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 5 test sets for each entity type.

underlying probabilistic distributions described before) and the structure (that is, E^d and R^d) of each document d by improving $P(D|\theta)$. Unlike 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.

There is an initial step to acquire a coarse guess of document structures and to seek the set of entities E in a closed collection of documents D. A local clustering is performed to group all mentions inside each document. A set of simple heuristics of matching attributes[4] is applied to computing the similarity between mentions, and pairs of mentions with similarity above a threshold are clustered together.

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 Table 4. 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, 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 5).

Structural assumptions: Our generative model benefits from the structural assumptions made in designing the model. 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 6 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.

Conclusion

The paper presents two learning approaches to the "robust reading" problem – cross-document identification of names despite ambiguous writings. In addition to a standard modelling of the problem as a classification task, we developed a model that describes the natural generation process of a document and the process of how names are "sprinkled" into them, taking into account dependencies between entities across types and an "author" model. We have shown that both models gain a lot from incorporating structural and relational information – features in the classification model; coherent data splits for clustering and the natural generation process in the case of the probabilistic model.

The *robust reading* problem is a very significant barrier on our way toward supporting robust natural language processing, and 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 NLP tasks.

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) integrating this work with other aspect of coreference resolution (e.g., pronouns).

[4]E.g., one example of a heuristics is: *If mention n_1 is a substring of mention n_2, then their similarity is* 0.8.

Acknowledgment

This research is supported by NSF grants IIS-9801638, ITR IIS-0085836 and EIA-0224453, an ONR MURI Award and an equipment donation from AMD.

References

Bilenko, M., and Mooney, R. 2003. Adaptive duplicate detection using learnable string similarity measures. In *KDD*.

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

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

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

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

Hernandez, M., and Stolfo, S. 1995. The merge/purge problem for large databases. In *SIGMOD*.

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

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

Mann, G., and Yarowsky, D. 2003. Unsupervised personal name disambiguation. In *CoNLL*.

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

Ng, V., and Cardie, C. 2003. Improving machine learning approaches to coreference resolution. In *ACL*.

Pasula, H.; Marthi, B.; Milch, B.; Russell, S.; and Shpitser, I. 2002. Identity uncertainty and citation matching. In *NIPS*.

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

Voorhees, E. 2002. Overview of the TREC-2002 question answering track. In *TREC*, 115–123.