Unsupervised Learning of Name Structure From Coreference Data*

Eugene Charniak
Brown Laboratory for Linguistic Information Processing
Department of Computer Science
Brown University, Box 1910, Providence, RI
ec@cs.brown.edu

Abstract

We present two methods for learning the structure of personal names from unlabeled data. The first simply uses a few implicit constraints governing this structure to gain a toehold on the problem — e.g., descriptors come before first names, which come before middle names, etc. The second model also uses possible coreference information. We found that coreference constraints on names improve the performance of the model from 92.6% to 97.0%. We are interested in this problem in its own right, but also as a possible way to improve named entity recognition (by recognizing the structure of different kinds of names) and as a way to improve noun-phrase coreference determination.

1 Introduction

We present two methods for the unsupervised learning of the structure of personal names as found in Wall Street Journal text. More specifically, we consider a "name" to be a sequence of proper nouns from a single noun-phrase (as indicated by Penn treebank-style parse trees). For example, "Defense Secretary George W. Smith" would be a name and we would analyze it into the components "Defense Secretary" (a descriptor), "George" (a first name), "W." (a middle name, we do not distinguish between initials and "true" names), and "Smith" (a last name).

We consider two unsupervised models for learning this information. The first simply uses a few implicit constraints governing this structure to gain a toehold on the problem — e.g., descriptors come before first names, which come before middle names, etc. We henceforth call this the "name" model. The second model also uses possible coreference information. Typically the same individual is mentioned several times in the same article (e.g., we might later encounter "Mr. Smith"), and the pattern of such references, and the mutual constraints among them, could very well help our unsupervised methods determine the correct structure. We call this the "coreference" model. We were attracted to this second model as it might offer a small example of how semantic information like coreference could help in learning structural information.

To the best of our knowledge there has not been any previous work on learning personal structure. We are aware of one previous case of unsupervised learning of lexical information from possible coreference, namely that of Ge et. al. [5] where possible pronoun coreference was used to learn the gender of nouns. In this case a program with an approximately 65% accuracy in determining the correct antecedent was used to collect information on pronouns and their possible antecedents. The gender of the pronoun was then used to suggest the gender of the noun-phrase that was proposed as the antecedent. The current work is quite different in both goal and methods, but similar in spirit. More generally this work is part of a growing body of work on learning language-related information from unlabeled corpora [1,2,3,8,9,10, 11].

2 Problem Definition and Data Preparation

We assume that people's names have six (optional) components as exemplified in the following somewhat contrived example:

* This research was supported in part by NSF grant LIS SBR 9720368. The author would like to thank Mark Johnson and the rest of the Brown Laboratory for Linguistic Information Processing (BLLIP) for general advice and encouragement.

Word	Label	Label Number
Defense	descriptor	0
Secretary	descriptor	0
Mr.	honorific	1
John	first-name	2
W.	middle-name	3
Smith	last-name	4
Jr.	close	5

Our models make the following assumptions about personal names:

- all words of label l (the label number) must occur before all words of label $l+1$
- with the exception of descriptors, a maximum of one word may appear for each label
- every name must include either a first name or a last name
- in a loose sense, honorifics and closes are "closed classes", even if we do not know which words are in the classes. We implement this by requiring that words given these labels must appear in our dictionary only as proper names, and that they must have appeared at least three times in the training set used for the lexicon (sections 2-21 of the Penn Wall Street Journal treebank)

Section 5 discusses these constraints and assumptions.

The input to the name model is a noisy list of personal names. This list is approximately 85% correct; that is, about 15% of the word sequences are not personal names, but rather non-names, or the names of other types of entities. We obtained these names by running a program inspired by that of Collins and Singer [4] for unsupervised learning of named entity recognition. This program takes as input possible names plus contextual information about their occurrences. It then categorizes each name as one of person, place, or organization. A possible name is considered to be a sequence of one or more proper nouns immediately dominated by a noun-phrase where the last of the proper nouns is the head (rightmost noun) of the noun phrase. We used as input to this program the parsed text found in the BLLIP WSJ 1987-89 WSJ Corpus — Release 1 [6]. Because of a minor error, the parser used in producing this corpus had a unwarranted propensity to label uncapitalized words as proper nouns. To correct for this we only allowed capitalized words to be considered proper nouns. In section 5 we note an unintended consequence of this decision.

The coreference model for our tasks is also given a list of *all* personal names (as defined above) in each Wall Street Journal article. Although the BLLIP corpus has machine-generated coreference markers, these are ignored.

The output of both programs is an assignment from each name to a sequence of labels, one for each word in the name. Performance is measured by the percent of words labeled correctly and percent of names for which all of the labels are correct.

3 The Probability Models

We now consider the probability models that underlie our learning mechanisms. Both models are generative in that they assign probabilities to all possible labelings of the names. (For the coreference model the model generates all possible labelings *given the proposed antecedent.*) Let $\vec{l}$ be a sequence of label assignments to the name $\vec{n}$ (a sequence of words). For the name model we estimate

$$\arg max_{\vec{l}}\, p(\vec{l} \mid \vec{n}) = \arg max_{\vec{l}}\, p(\vec{l}, \vec{n}) \quad (1)$$

We estimate this latter probability by assuming that the number of words assigned label l, $n(l)$, is independent of which other labels have appeared. Our assumptions imply that with the exception of descriptor, all labels may occur zero or one times. We arbitrarily assume that there may be zero to fourteen descriptors. We then assume that the words in the name are independent of one another given their label. Thus we get the following equation:

$$p(\vec{l}, \vec{n}) = \prod_{l=0,5} p(\mathcal{N}(l) = n(l)) \prod_{i=0,n(l)} p(w(i) \mid l) \quad (2)$$

Here $w(i)$ is the ith word from $\vec{n}$ assigned the label l in $\vec{l}$ and $\mathcal{N}(l)$ is a random variable whose value is the number of words in the name with label l. To put this slightly differently, we first guess the number of words with each label l according to the distribution $p(\mathcal{N}(l) = n(l))$. Given the ordering constraints, this completely determines which words in $\vec{n}$ get which label. We then guess each of the words according to the distribution $p(w(i) \mid l)$. The name model does not use information concerning how often each name occurs. (That is, it implicitly assumes that all names occur equally often.)

We have also considered somewhat more complex approximations to $p(\vec{l}, \vec{n})$. See section 5.

The coreference model is more complicated. Here we estimate

$$\arg max_{\vec{l}} p(\vec{l} \mid \vec{n}, \vec{c}) = \arg max_{\vec{l}}\, p(\vec{l}, \vec{n} \mid \vec{c}) \quad (3)$$

That is, for each name the program identifies zero or one *possible* antecedent name. It does this using a very crude filter. The last word of the proposed antecedent (unless that word is "Jr.", in which case it looks at the second to last word) must also appear in $\vec{n}$ as well. If no such name exists, then $\vec{c} = \triangle$ and we estimate the distribution according to equation 2. If more than one such name exists, we choose the first one appearing in the article.

Even if there is such a name, the program does not assume that the two names are, in fact, coreferent. Rather, a hidden random variable R determines how the two names relate. There are three possibilities:

- $\vec{c}$ is not coreferent ($R = \triangle$), in which case the probability is estimated according to the $\vec{c} = \triangle$ case.
- $\vec{c}$ is not coreferent but is a member of the same family as $\vec{n}$ (e.g., "John Sebastian Bach" and "Carl Philipp Emmanuel Bach"). This case ($R = f$) is computed as the non-coreference case, but the second occurrence is given "credit" for the last name.
- $\vec{c}$ is coreferent to $\vec{n}$, in which case we compute the probability as described below. In this case we assume that any words shared by the two names must appear with the same label, and except for descriptions, labels may not change between them (e.g., if $\vec{c}$ has a first name, then $\vec{n}$ can be given a first name only if it is the same word as that in $\vec{c}$). This does not allow for nicknames and other such cases, but they are rare in the Wall Street Journal.

More formally, we have

$$p(\vec{n}, \vec{l} \mid \vec{c}) = \sum_{R} p(R) p(\vec{n}, \vec{l} \mid R, \vec{c}) \quad (4)$$

We then estimate $p(\vec{n}, \vec{l} \mid R, \vec{c})$ as follows:

- if $R = \triangle$, compute $p(\vec{n}, \vec{l})$ as estimated from equation 2.
- if $R = f$, then $p(\vec{n}, \vec{l}) / p(s \mid \vec{l}(s))$ where s is the word in common that caused the previous name to be selected as a possible coreferent and $\vec{l}(s)$ is the label assigned to s according to $\vec{l}$.
- if $R = \vec{c}$, use equation 8 below.

In equation 4 the $R = \triangle$ case is reasonably straight-forward: we simply use equation 2 as the non-coreferent distribution. For $R = f$, as we noted earlier, we want to claim that the new name is a member of the same family as that of the earlier name. Thus, as we said earlier, we get "credit" for the repeated family name. This is why we take the non-coreferent probability and divide by the probability of what we take to be the family name.

This leaves the coreferent case. The basic idea is that we view the labeling of new name ($\vec{l}$) as a transformation of the labeling of the old one ($\vec{l'}$). However, we do not know $\vec{l'}$ so we have to sum over all possible former labelings $\mathcal{L}'$. This is expressed as

$$p(\vec{n}, \vec{l} \mid \vec{c}) = \sum_{\vec{l'} \in \mathcal{L}'} p(\vec{l'} \mid \vec{c}) p(\vec{n}, \vec{l} \mid \vec{l'}, \vec{c}) \quad (5)$$

The first term, $p(\vec{l'} \mid \vec{c})$, is easy to compute from equation 2 using Bayes law. We now turn our attention to the second term.

To establish a more detailed relationship between the old and new names we compute possible correspondences between the two names, where a correspondence specifies for each word in the old name if it is retained in the new name, and if so, the word in the new name to which it corresponds. Two words may correspond only if they are the same lexical item. (The converse does not hold.) Since in principle there can be multiple correspondences, we introduce the correspondences κ by summing the probability over all of them:

$$p(\vec{n},\vec{l} \mid \vec{l'},\vec{c}) = \sum_{\kappa} p(\vec{n},\vec{l},\kappa \mid \vec{l'},\vec{c}) \quad (6)$$

$$\approx max_{\kappa} p(\vec{n},\vec{l},\kappa \mid \vec{l'},\vec{c}) \quad (7)$$

In the second equation we simplify by making the assumption that the sum will be dominated by one of the correspondences, a very good assumption. Furthermore, as is intuitively plausible, one can identify the maximum κ without actually computing the probabilities: it is the κ with the maximum number of words retained from $\vec{c}$. Henceforth we use κ to denote this maximum-probability correspondence.

By specifying κ we divide the words of the old name into two groups, those ($\mathcal{R}$) that are retained in the new name and those ($\mathcal{S}$) that are subtracted when going to the new name. Similarly we have divided the words of the new name into two classes, those retained and those added ($\mathcal{A}$). We then assume that the probability of a word being subtracted or retained is independent of the word and depends only on its label (e.g., the probability of a subtraction given the label l is $p(s \mid l)$). Furthermore, we assume that the labels of words in $\mathcal{R}$ do not change between $\vec{l'}$ and $\vec{l}$. Once we have pinned down $\mathcal{R}$ and $\mathcal{S}$, any words left in $\vec{l}$ must be added. However, we do not yet "know" the labels of those, so we need a probability term $p(l \mid a)$. Lastly, for words that are added in the new name, we need to guess the particular word corresponding to the label type. This gives us the following distribution:

$$p(\vec{n},\vec{l},\kappa \mid \vec{c},\vec{l'}) = \prod_{w \in \mathcal{S}} p(s \mid \vec{l'}(w)) \prod_{w \in \mathcal{R}} p(r \mid \vec{l'}(w)) \prod_{w \in \mathcal{A}} p(l \mid a) \; p(w \mid l) \quad (8)$$

Taken together, equations 2 — 8 define our probability model.

4 Experiments

From the work on named entity recognition we obtained a list of 145,670 names, of which 87,809 were marked as personal names. A second program creates an ordered list of names that appear in each article in the corpus. The two files, names and article-name occurrences, are the input to our procedures.

With one exception, all the probabilities required by the two models are initialized with flat distributions — i.e., if a random variable can take n possible values, each value is $1/n$. The probabilites so set are:

1. $p(\mathcal{N}(l) = n(l))$ from equation 2 (the probability that label l appears $n(l)$ times),
2. $p(w(i) \mid l)$ from equation 2 (the probability of generating $w(i)$ given it has label l),
3. $p(s \mid \vec{l'}(w))$, $p(r \mid \vec{l'}(w))$, and $p(a \mid \vec{l}(w))$ from equation 8, the probabilities that a the label $\vec{l}(w)$ will be subtracted, retained, or added when going from the old name to the new name.

We then used the expectation-maximization (EM) algorithm to re-estimate the values. We initially decided to run EM for 100 iterations as our benchmark. In practice no change in performance was observed after about 15 iterations.

The one exception to the flat probability distribution rule is the probability distribution $p(R)$, the probability of an antecedent being coreferent, a family relation, or non-coreferent. This distribution was set at .993, .002, and .005 respectively for the three alternatives and the values were not re-estimated by EM.[1] Figure 1 show some of the probabilities for individual words given the possible labels.

The result shown in Figure 1 are basically correct, with "Director" having a high probability as a descriptor, (0.0059), "Ms." having a high probability as honorific (0.058), etc. Some of the small non-zero probabilities are due to genuine ambiguity (e.g., Fisher does occur as a first name as well as a last name) but more of it is due to small confusions in particular cases (e.g., "Director" as a last-name, or "John" as descriptor).

After EM training we evaluated the program on 309 personal names from our names list that we had annotated by hand. These names were obtained by random selection of names labeled

[1] These were the first values we tried and, as they worked satisfactorily, we simply left them alone.

Word	$p(w \mid 0)$	$p(w \mid 1)$	$p(w \mid 2)$	$p(w \mid 3)$	$p(w \mid 4)$	$p(w \mid 5)$
Director	0.0059	0	$2.0\ 10^{-5}$	0	0.00016	0
Ms.	$1.2\ 10^{-7}$	0.058	0.0041	$2.9\ 10^{-14}$	0	0
John	0.0037	$2.9\ 10^{-6}$	0.038	$9.7\ 10^{-6}$	0	0
T.	0.0018	$1.2\ 10^{-12}$	.00032	0.02	0	0
Fisher	0	0	$4.5\ 10^{-5}$	$7.4\ 10^{-5}$	0.00073	0
III	0	0	0	0	$6.8\ 10^{-5}$	0.16

Figure 1: Some example probabilities of words given labels after 15 EM iterations

as personal names by the name-entity recognizer. If the named entity recognizer had mistakenly classified something as a personal name it was not used in our test data.

For the name model we straightforwardly used equation 2 to determine the most probable label sequence $\vec{l}$ for each name. Note, however, that the testing data does not itself include any information on whether or not the test name was a first or subsequent occurrence of an individual in the text. To evaluate the coreference model we looked at the possible coreference data to find if the test-data name was most common as a first occurrence, or if not, which possible antecedent was the most common. If first occurrence prevailed, $\vec{l}$ was determined from equation 2, and otherwise it was determined using equation 3 with $\vec{c}$ set to the most common possible coreferent for this name.

We compare the most probable labels $\vec{l}$ for a test example with the hand-labeled test data. We report percentage of words that are given the correct label and percentage of names that are completely correct. The results of our experiments are as follows:

Model	Label%	Name%
Name	92.6	85.1
Coreference	97.0	94.5

As can be seen, information about possible coreference was a decided help in this task, leading to an error reduction of 59% for the number of labels correct and 63% for names correct.

5 Error Analysis

The errors tend to arise from three situations: the name disobeys the name structure assumptions upon which the program is based, the name is anomalous in some way, or sparse data. We consider each of these in turn.

Many of the names we encounter do not obey our assumptions. Probably the most common situation is last names that, contrary to our assumption, are composed of more than word e.g., "Van Dam". Actually, a detail of our processing has caused this case to be under-represented in our data and testing examples. As noted in Section 2, uncapitalized proper nouns were not allowed. The most common extra last name is probably "van," but all of these names were either truncated or ignored because of our processing step.

In principle, it should be possible to allow for multiple last names, or alternatively have a new label for "first of two last names". In practice, it is of course the case that the more parameters we give EM to fiddle with, the more mischief it can get into. However, for a practical program this is probably the most important extension we envision.

Names may be anomalous while obeying our restrictions at least in the letter if not the spirit. Chinese names have something very much like the first-middle-last name structure we assume, but the family name comes first. This is particularly relevant for the coreferent model, since it will be the family name that is repeated. There is nothing in our model that prevents this, but it is sufficiently rare that the program gets "confused". In a similar way, we marked both "Dr." and "Sir" as honorifics in our test data. However, the Wall Street Journal treats them very differently from "Mr." in that the former tend to be included even in the first mention of a name, while the latter is not. Thus in some cases our program labeled "Dr." and "Sir" as descriptors.

Lastly, there are situations where we imagine that if the program had more data (or if the learning mechanisms were somehow "better") it would get the example right. For example, the name “Mikio Suzuki” appears only once in our corpus, as does the word “Mikio”. “Suzuki” appears two times, the first being in “Yotaro Suzuki” who is mentioned earlier in the same article. Unfortunately, because “Mikio” does not appear elsewhere, the program is at a loss to decide which label to give it. However, because Yotaro is assumed to be a first name, the program makes “Mikio Suzuki” coreferent with “Yotaro Suzuki” by labeling “Mikio” descriptor.

As noted briefly in section 3, we have considered more complicated probabilities models to replace equation 2. The most obvious of these is to allow the distribution over numbers of words for each label to be conditioned on the previous label — e.g., a bi-label model. This model generally performed poorly, although the coreference versions often performed as well as the coreference model reported here. Our hypothesis is that we are seeing problems similar to those that have bedeviled applying EM to tasks like part-of-speech tagging [7]. In such cases EM typically tries to lower probabilities of the corpus by using the tags to encode common word-word combinations. As the models corresponding to equations 2 and 8 do not include any label-label probabilities, this problem does not appear in these models.

6 Other Applications

It is probably clear to most readers that the structure and probabilities learned by these models, particularly the coreferent model, could be used for tasks other than assigning structure to names. For starters, we would imagine that a named entity recognition program that used information about name structure could do a better job. The named entity recognition program used to create the input looks at only a few features of the context in which the name appears, the complete name, and the individual words that appear in the name irrespective of the other words. Since the different kinds of names (person, company and location) differ in structure from one another, a program that simultaneously establishes both structure and type would have an extra source of information, thus enabling it to do a better job.

Our name-structure coreferent model is also learning a lot of information that would be useful for a program whose primary purpose is to detect coreference. One way to see this is to look at some of the probabilities that the program learned. Consider the probability that we will have an honorific in a first occurrence of a name:

$$p(n(1) = 1) = .000044 \qquad (9)$$

This is very low. Contrast this with the probability that we add an honorific in the second occurrence:

$$p(a \mid \mathsf{honorific}) = 1 \qquad (10)$$

These dramatic probabilities are not, in fact, accurate, as EM tends to exaggerate the effects by moving words that do not obey the trend out of the honorific category. They are, however, indicative of the fact that in the Wall Street Journal names are introduced without honorifics, but subsequent occurrences tend to have them (a fact we were not aware of at the start of this research).

Another way to suggest the usefulness of this research for coreference and named-entity recognition is to consider the cases where our program's crude filter suggests a possible antecedent, but the probabilistic model of equation 4 rejects this analysis. The first 15 cases are given in figure 2. As can be seen, except for “Mr. President” and “President Reagan”, all of the examples are either not coreferent or are not people at all.

7 Conclusion

We have presented two methods for the unsupervised discovery of personal-name structure. The two methods differ in that the second uses multiple, possibly coreferent, occurrences of names to constrain the problem. The methods perform at a level of 92.6 and 97.0 percent accuracy respectively.

The methods are of potential interest in their own right, e.g., to improve the level of detail provided by Penn Treebank style parses. As we have also noted, we should also be able to use this research in the quest for better unsupervised learning of named-entity recognition, and the model that attends to coreference information can potentially be useful for programs aimed directly at this latter problem.

Finally, many of us believe that the power of unsupervised learning methods for linguistic

Vice President	President Reagan
Mr. President	President Reagan
Dean P. Guerin	Guerin & Turner
Ronald Reagan White House	House Speaker James Wright
Rev. Leon H. Sullivan	Sullivan Principles
Mr. Sullivan	Sullivan Principles
Rev. Leon Sullivan	Sullivan Principles
South Dakota Republican	Republican Party
Kansas Republican	Republican Party
Cyril Wagner Jr.	Wagner & Brown
General Preston R. Tisch	Laurence A. Tisch
Lt. Col. Oliver North	Whitney North Seymour Jr.
Republican White House	House Republicans
J. Seward Johnson	Johnson & Johnson
Vice President	President Nixon

Figure 2: The first 15 cases in which the coreferent model rejected the coreferent hypothesis

information will be proportional to the depth of semantic or pragmatic analysis that goes into the features they consider. The vastly superior performance of the coreference model over the basic name model moves this believe some small distance from hope to hypothesis.

References

1. Berland, M. and Charniak, E. *Finding parts in very large corpora.* In *Proceedings of the ACL 1999*. ACL, New Brunswick NJ, 1999, 57–64.
2. Brill, E. *Unsupervised learning of disambiguation rules for part of speech tagging.* In *Proceedings of the Third Workshop on Very Large Corpora*. 1995, 1–13.
3. Caraballo, S. A. *Automatic construction of a hypernym-labeled noun hierarchy from text.* In *Proceedings of the ACL 1999*. ACL, New Brunswick NJ, 1999.
4. Collins, M. and Singer, Y. *Unsupervised models for named entity classification.* In *Proceedings of the 1999 Joint Sigdat Conference on Empirical Methods in Natural Language Processing and Very Large Corpora.* Association for Computational Linguistics, 1999.
5. Ge, N., Hale, J. and Charniak, E. *A statistical approach to anaphora resolution.* In *Proceedings of the Sixth Workshop on Very Large Corpora*. 1998, 161–171.
6. Linguistic Data Consortium. *BLLIP 1987-1989 WSJ Corpus Release 1*. 2000.
7. Merialdo, B. Tagging English text with a probabilistic model. *Computational Linguistics 20* (1994), 155–172.
8. Riloff, E. *Automatically generating extraction patterns from untagged text.* In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*. AAAI Press/MIT Press, Menlo Park, 1996, 1044–1049.
9. Riloff, E. and Shepherd, J. *A corpus-based approach for building semantic lexicons.* In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*. 1997, 117–124.
10. Roark, B. and Charniak, E. *Noun-phrase co-occurrence statistics for semi-automatic semantic lexicon construction.* In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*. 1998, 1110–1116.
11. Yarowsky, D. *Unsupervised word sense disambiguation rivaling supervised methods.* In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*. 1995, 189–196.