A Probabilistic Model for Canonicalizing Named Entity Mentions

Dani Yogatama Yanchuan Sim Noah A. Smith
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{dyogatama,ysim,nasmith}@cs.cmu.edu

Abstract

We present a statistical model for canonicalizing named entity mentions into a table whose rows represent entities and whose columns are attributes (or parts of attributes). The model is novel in that it incorporates entity context, surface features, first-order dependencies among attribute-parts, and a notion of noise. Transductive learning from a few seeds and a collection of mention tokens combines Bayesian inference and conditional estimation. We evaluate our model and its components on two datasets collected from political blogs and sports news, finding that it outperforms a simple agglomerative clustering approach and previous work.

1 Introduction

Proper handling of mentions in text of real-world entities—identifying and resolving them—is a central part of many NLP applications. We seek an algorithm that infers a set of real-world entities from mentions in a text, mapping each entity mention token to an entity, and discovers general categories of words used in names (e.g., titles and last names). Here, we use a probabilistic model to infer a structured representation of canonical forms of entity attributes through transductive learning from named entity mentions with a small number of seeds (see Table 1). The input is a collection of mentions found by a named entity recognizer, along with their contexts, and, following Eisenstein et al. (2011), the output is a table in which entities are rows (the number of which is not pre-specified) and attribute words are organized into columns.

This paper contributes a model that builds on the approach of Eisenstein et al. (2011), but also:

- incorporates context of the mention to help with disambiguation and to allow mentions that do not share words to be merged liberally;
- conditions against shape features, which improve the assignment of words to columns;
- is designed to explicitly handle some noise; and
- is learned using elements of Bayesian inference with conditional estimation (see §2).

We experiment with variations of our model, comparing it to a baseline clustering method and the model of Eisenstein et al. (2011), on two datasets, demonstrating improved performance over both at recovering a gold standard table. In a political blogs dataset, the mentions refer to political figures in the United States (e.g., Mrs. Obama and Michelle Obama). As a result, the model discovers parts of names—⟨Mrs., Michelle, Obama⟩—while simultaneously performing coreference resolution for named entity mentions. In the sports news dataset, the model is provided with named entity mentions of heterogenous types, and success here consists of identifying the correct team for every player (e.g., Kobe Bryant and Los Angeles Lakers). In this scenario, given a few seed examples, the model begins to identify simple relations among named entities (in addition to discovering attribute structures), since attributes are expressed as named entities across multiple mentions. We believe this adaptability is important, as the salience of different kinds of names and their usages vary considerably across domains.

Bill	Clinton	Mr.			
George	Bush	Mr.			W.
Barack	Obama		Sen.		Hussein
Hillary	Clinton	Mrs.	Sen.		
Bristol	Palin	Ms.			
Emil	Jones			Jr.	
Kay	Hutchison				Bailey

Ben	Roethlisberger			Steelers
	Bryant	Los	Angeles	
Derek	Jeter	New	York	

Table 1: Seeds for politics (above) and sports (below).

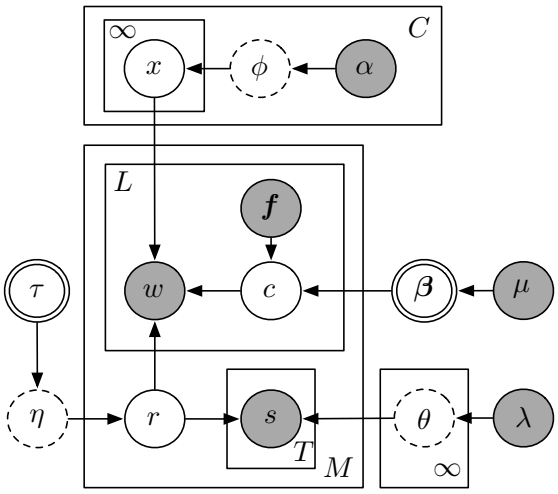

Figure 1: Graphical representation of our model. Top, the generation of the table: C is the number of attributes/columns, the number of rows is infinite, $\boldsymbol{\alpha}$ is a vector of concentration parameters, ϕ is a multinomial distribution over strings, and x is a word in a table cell. Lower left, for choosing entities to be mentioned: τ determines the stick lengths and η is the distribution over entities to be selected for mention. Middle right, for choosing attributes to use in a mention: $\boldsymbol{f}$ is the feature vector, and $\boldsymbol{\beta}$ is the weight vector drawn from a Laplace distribution with mean zero and variance μ. Center, for generating mentions: M is the number of mentions in the data, w is a word token set from an entity/row r and attribute/column c. Lower right, for generating contexts: s is a context word, drawn from a multinomial distribution θ with a Dirichlet prior λ. Variables that are known or fixed are shaded; variables that are optimized are double circled. Others are latent; dashed lines imply collapsing.

2 Model

We begin by assuming as input a set of mention tokens, each one or more words. In our experiments these are obtained by running a named entity recognizer. The output is a table in which rows are understood to correspond to entities (types, not mention tokens) and columns are fields, each associated with an attribute or a part of it. Our approach is based on a probabilistic graphical model that generates the mentions, which are observed, and the table, which is mostly unobserved, similar to Eisenstein et al. (2011). Our learning procedure is a hybrid of Bayesian inference and conditional estimation. The generative story, depicted in Figure 1, is:

- For each column $j \in \{1, \ldots, C\}$:
  - Draw a multinomial distribution ϕ_j over the vocabulary from a Dirichlet process: $\phi_j \sim \mathrm{DP}(\alpha_j, G_0)$. This is the lexicon for field j.
  - Generate table entries. For each row i (of which there are infinitely many), draw an entry $x_{i,j}$ for cell i, j from ϕ_j. A few of these entries (the seeds) are observed; we denote those $\tilde{x}$.
  - Draw weights $\boldsymbol{\beta}_j$ that associate shape and positional features with columns from a 0-mean, μ-variance Laplace distribution.
- Generate the distribution over entities to be mentioned in general text: $\eta \sim \mathrm{GEM}(\tau)$ ("stick-breaking" distribution).
- Generate context distributions. For each row r:
  - Draw a multinomial over the context vocabulary (distinct from mention vocabulary) from a Dirichlet distribution, $\theta_r \sim \mathrm{Dir}(\lambda)$.
- For each mention token m:
  - Draw an entity/row $r \sim \eta$.
  - For each word in the mention w, given some of its features $\boldsymbol{f}$ (assumed observed):
    - ▷ Choose a column $c \sim \frac{1}{Z} \exp(\boldsymbol{\beta}_c^\top \boldsymbol{f})$. This uses a log-linear distribution with partition function Z. In one variation of our model, first-order dependencies among the columns are enabled; these introduce a dynamic character to the graphical model that is not shown in Figure 1.
    - ▷ With probability $1 - \epsilon$, set the text $w_{m\ell}$ to be x_{rc}. Otherwise, generate any word from a unigram-noise distribution.
  - Generate mention context. For each of the $T = 10$ context positions (five before and five after the mention), draw the word s from θ_r.

Our choices of prior distributions reflect our beliefs about the shapes of the various distributions. We expect field lexicons ϕ_j and the distributions over mentioned entities η to be "Zipfian" and so use tools from nonparametric statistics to model them. We expect column-feature weights $\boldsymbol{\beta}$ to be mostly zero, so a sparsity-inducing Laplace prior is used (Tibshirani, 1996).

Our goal is to maximize the *conditional* likelihood of most of the evidence (mentions, contexts, and seeds), $p(w, s, \tilde{x} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \lambda, \tau, \mu, \epsilon, \boldsymbol{f}) =$

$$\sum_r \sum_c \sum_{x \setminus \tilde{x}} \int d\theta \int d\eta \int d\phi \; p(w, s, r, c, x, \theta, \eta, \phi \mid \boldsymbol{\alpha}, \boldsymbol{\beta}, \lambda, \tau, \mu, \epsilon, \boldsymbol{f})$$

with respect to $\boldsymbol{\beta}$ and τ. We fix $\boldsymbol{\alpha}$ (see §3.3 for the values of $\boldsymbol{\alpha}$ for each dataset), $\lambda = 2$ (equivalent to add-one smoothing), $\mu = 2 \times 10^{-8}$, $\epsilon = 10^{-10}$, and each mention word's $\boldsymbol{f}$. Fixing λ, μ, and $\boldsymbol{\alpha}$ is essentially just "being Bayesian," or fixing a hyperparameter based on prior beliefs. Fixing $\boldsymbol{f}$ is quite different; it is conditioning our model on some observable features of the data, in this case word shape features. We do this to avoid integrating over feature vector values. These choices highlight that the design of a probabilistic model can draw from both Bayesian and discriminative tools. Observing some of x as seeds ($\tilde{x}$) renders this approach transductive.

Exact inference in this model is intractable, so we resort to an approximate inference technique based on Markov Chain Monte Carlo simulation. The optimization of $\boldsymbol{\beta}$ can be described as "contrastive" estimation (Smith and Eisner, 2005), in which some aspects of the data are conditioned against for computational convenience. The optimization of τ can be described as "empirical Bayesian" estimation (Morris, 1983) in which the parameters of a prior are fit to data. Our overall learning procedure is a Monte Carlo Expectation Maximization algorithm (Wei and Tanner, 1990).

3 Learning and Inference

Our learning procedure is an iterative algorithm consisting of two steps. In the E-step, we perform collapsed Gibbs sampling to obtain distributions over row and column indices for every mention, given the current value of the hyperparamaters. In the M-step, we obtain estimates for the hyperparameters, given the current posterior distributions.

3.1 E-step

For the mth mention, we sample row index r, then for each word $w_{m\ell}$, we sample column index c.

3.1.1 Sampling Rows

Similar to Eisenstein et al. (2011), when we sample the row for a mention, we use Bayes' rule and marginalize the columns. We further incorporate context information and a notion of noise.

$$p(r_m = r \mid \ldots) \propto p(r_m = r \mid r_{-m}, \eta) \\ (\textstyle\prod_\ell \sum_c p(w_{m\ell} \mid x, r_m = r, c_{m\ell} = c)) \\ (\textstyle\prod_t p(s_{mt} \mid r_m = r))$$

We consider each quantity in turn.

Prior. The probability of drawing a row index follows a stick breaking distribution. This allows us to have an unbounded number of rows and let the model infer the optimal value from data. A standard marginalization of η gives us:

$$p(r_m = r \mid r_{-m}, \tau) = \begin{cases} \frac{N_r^{-m}}{N+\tau} & \text{if } N_r^{-m} > 0 \\ \frac{\tau}{N+\tau} & \text{otherwise,} \end{cases}$$

where N is the number of mentions, N_r is the number of mentions assigned to row r, and N_r^{-m} is the number of mentions assigned to row r, excluding m.

Mention likelihood. In order to compute the likelihood of observing mentions in the dataset, we have to consider a few cases. If a cell in a table has already generated a word, it can only generate that word. This hard constraint was a key factor in the inference algorithm of Eisenstein et al. (2011); we speculate that softening it may reduce MCMC mixing time, so introduce a notion of noise. With probability $\epsilon = 10^{-10}$, the cell can generate any word. If a cell has not generated any word, its probability still depends on other elements of the table. With base distribution G_0,[1] and marginalizing ϕ, we have:

$$p(w_{m\ell} \mid x, r_m = r, c_{m\ell} = c, \alpha_c) = \qquad (1)$$
$$\begin{cases} 1 - \epsilon & \text{if } x_{rc} = w_{m\ell} \\ \epsilon & \text{if } x_{rc} \notin \{w_{m\ell}, \varnothing\} \\ \frac{N_{cw}^{-m\ell}}{N_c^{-m\ell} + \alpha_c} & \text{if } x_{rc} = \varnothing \text{ and } N_{cw} > 0 \\ G_0(w_{m\ell}) \frac{\alpha_c}{N_c^{-m\ell} + \alpha_c} & \text{if } x_{rc} = \varnothing \text{ and } N_{cw} = 0 \end{cases}$$

where $N_c^{-m\ell}$ is the number of cells in column c that are not empty and $N_{cw}^{-m\ell}$ is the number of cells in column c that are set to the word $w_{m\ell}$; both counts excluding the current word under consideration.

Context likelihood. It is important to be able to use context information to determine which row a mention should go into. As a novel extension, our model also uses surrounding words of a mention as its "context"—similar context words can encourage two mentions that do not share any words to be merged. We choose a Dirichlet-multinomial distribution for our context distribution. For every row in the table, we have a multinomial distribution over context vocabulary θ_r from a Dirichlet prior λ.

[1] We let G_0 be a uniform distribution over the vocabulary.

Therefore, the probability of observing the tth context word for mention m is $p(s_{mt} \mid r_m = r, \lambda)$

$$= \begin{cases} \frac{N_{rs}^{-mt} + \lambda_s - 1}{N_r^{-mt} + \sum_v \lambda_v - V} & \text{if } N_r^{-mt} > 0 \\ \frac{\lambda_s - 1}{\sum_v \lambda_v - V} & \text{otherwise,} \end{cases}$$

where N_r^{-mt} is the number of context words of mentions assigned to row r, N_{rs}^{-mt} is the number of context words of mentions assigned to row r that are s_{mt}, both excluding the current context word, and v ranges over the context vocabulary of size V.

3.1.2 Sampling Columns

Our column sampling procedure is novel to this work and substantially differs from that of Eisenstein et al. (2011). First, we note that when we sample column indices for each word in a mention, the row index for the mention r has already been sampled. Also, our model has interdependencies among column indices of a mention.[2] Standard Gibbs sampling procedure breaks down these dependencies. For faster mixing, we experiment with first-order dependencies between columns when sampling column indices. This idea was suggested by Eisenstein et al. (2011, footnote 1) as a way to learn structure in name conventions. We suppressed this aspect of the model in Figure 1 for clarity.

We sample the column index c_1 for the first word in the mention, marginalizing out probabilities of other words in the mention. After we sample the column index for the first word, we sample the column index c_2 for the second word, fixing the previous word to be in column c_1, and marginalizing out probabilities of $c_3, \ldots, c_L$ as before. We repeat the above procedure until we reach the last word in the mention. In practice, this can be done efficiently using backward probabilities computed via dynamic programming. This kind of blocked Gibbs sampling was proposed by Jensen et al. (1995) and used in NLP by Mochihashi et al. (2009). We have: $p(c_{m\ell} = c \mid \ldots) \propto$

$$p(c_{m\ell} = c \mid \boldsymbol{f}_{m\ell}, \beta) p(c_{m\ell} = c \mid c_{m\ell_-} = c_-) \Big(\sum_{c_+} p_b(c_{m\ell} = c \mid c_{m\ell_+} = c_+)\Big) p(w_{m\ell} \mid x, r_m = r, c_{m\ell} = c, \alpha_c),$$

[2] As shown in Figure 1, column indices in a mention form "v-structures" with the row index r. Since every w_ℓ is observed, there is an active path that goes through all these nodes.

where ℓ_- is the preceding word and c_- is its sampled index, ℓ_+ is the following word and c_+ is its possible index, and $p_b(\cdot)$ are backward probabilities. Alternatively, we can perform standard Gibbs sampling and drop the dependencies between columns, which makes the model rely more heavily on the features. For completeness, we detail the computations.

Featurized log linear distribution. Our model can use arbitrary features to choose a column index. These features are incorporated as a log-linear distribution, $p(c_{m\ell} = c \mid \boldsymbol{f}_{m\ell}, \boldsymbol{\beta}) = \frac{\exp(\boldsymbol{\beta}_c^\top \boldsymbol{f}_{m\ell})}{\sum_{c'} \exp(\boldsymbol{\beta}_{c'}^\top \boldsymbol{f}_{m\ell})}$. The list of features used in our experiments is: $\mathbf{1}\{w$ is the first word in the mention$\}$; $\mathbf{1}\{w$ ends with a period$\}$; $\mathbf{1}\{w$ is the last word in the mention$\}$; $\mathbf{1}\{w$ is a Roman numeral$\}$; $\mathbf{1}\{w$ starts with an upper-case letter$\}$; $\mathbf{1}\{w$ is an Arabic number$\}$; $\mathbf{1}\{w \in \{\mathsf{mr}, \mathsf{mrs}, \mathsf{ms}, \mathsf{miss}, \mathsf{dr}, \mathsf{mdm}\}\}$; $\mathbf{1}\{w$ contains ≥ 1 punctuation symbol$\}$; $\mathbf{1}\{w \in \{\mathsf{jr}, \mathsf{sr}\}\}$; $\mathbf{1}\{w \in \{\mathsf{is}, \mathsf{in}, \mathsf{of}, \mathsf{for}\}\}$; $\mathbf{1}\{w$ is a person entity$\}$; $\mathbf{1}\{w$ is an organization entity$\}$.

Forward and backward probabilities. Since we introduce first-order dependencies between columns, we have forward and backward probabilities, as in HMMs. However, we always sample from left to right, so we do not need to marginalize random variables to the left of the current variable because their values are already sampled. Our transition probabilities are as follows:

$$p(c_{m\ell} = c \mid c_{m\ell_-} = c_-) = \frac{N_{c_-,c}^{-m}}{\sum_{c'_-} N_{c'_-,c}^{-m}},$$

where $N_{c_-,c}^{-m}$ is the number of times we observe transitions from column c_- to c, excluding mention m. The forward and backward equations are simple (we omit them for space).

Mention likelihood. Mention likelihood $p(w_{m\ell} \mid x, r_m = r, c_{m\ell} = c, \alpha_c)$ is identical to when we sample the row index (Eq. 1).

3.2 M-step

In the M-step, we use gradient-based optimization routines, L-BFGS (Liu and Nocedal, 1989) and OWL-QN (Andrew and Gao, 2007) respectively, to maximize with respect to τ and $\boldsymbol{\beta}$.

3.3 Implementation Details

We ran Gibbs sampling for 500 iterations,[3] discarding the first 200 for burn-in and averaging counts over every 10th sample to reduce autocorrelation.

For each word in a mention w, we introduced 12 binary features $\boldsymbol{f}$ for our featurized log-linear distribution (§3.1.2).

We then downcased all words in mentions for the purpose of defining the table and the mention words w. Ten context words (5 each to the left and right) define s for each mention token.

For non-convex optimization problems like ours, initialization is important. To guide the model to reach a good local optimum without many restarts, we manually initialized feature weights and put a prior on transition probabilities to reflect phenomena observed in the initial seeds. The initializer was constructed once and not tuned across experiments.[4] The M-step was performed every 50 Gibbs sampling iterations. After inference, we filled each cell with the word that occurred at least 80% of the time in the averaged counts for the cell, if such a word existed.

4 Experiments

We compare several variations of our model to Eisenstein et al. (2011) (the authors provided their implementation to us) and a clustering baseline.

4.1 Datasets

We collected named entity mentions from two corpora: political blogs and sports news. The political blogs corpus is a collection of blog posts about politics in the United States (Eisenstein and Xing, 2010), and the sports news corpus contains news summaries of major league sports games (National Basketball Association, National Football League, and Major League Baseball) in 2009. Due to the large size of the corpora, we uniformly sampled a subset of documents for each corpus and ran the Stanford NER tagger (Finkel et al., 2005), which tagged named entities mentions as person, location, and organization. We used named entity of type person from the political blogs corpus, while we are interested in person and organization entities for the sports news corpus. Mentions that appear less than five times are discarded. Table 2 summarizes statistics for both datasets of named entity mentions.

	Politics	**Sports**
# source documents	3,000	700
# mentions	10,647	13,813
# unique mentions	528	884
size of mention vocabulary	666	1,177
size of context vocabulary	2,934	2,844

Table 2: Descriptive statistics about the datasets.

Reference tables. We use Eisenstein et al.'s manually built 125-entity (282 vocabulary items) reference table for the politics dataset. Each entity in the table is represented by the set of all tokens that appear in its references, and the tokens are placed in its correct column. For the sports data, we obtained a roster of all NBA, NFL, and MLB players in 2009. We built our sports reference table using the players' names, teams and locations, to get 3,642 players and 15,932 vocabulary items. The gold standard table for the politics dataset is incomplete, whereas it is complete for the sports dataset.

Seeds. Table 1 shows the seeds for both datasets.

4.2 Evaluation Scores

We propose both a row evaluation to determine how well a model disambiguates entities and merges mentions of the same entity and a column evaluation to measure how well the model relates words used in different mentions. Both scores are new for this task.

The first step in evaluation is to find a maximum score bipartite matching between rows in the response and reference table.[5] Given the response and

[3] On our moderate-sized datasets (see §4.1), each iteration takes approximately three minutes on a 2.2GHz CPU.

[4] For the politics dataset, we set $C = 6$, $\boldsymbol{\alpha} = \langle 1.0, 1.0, 10^{-12}, 10^{-15}, 10^{-12}, 10^{-8} \rangle$, initialized $\tau = 1$, and used a Dirichlet prior on transition counts such that before observing any data: $N_{0,1} = 10, N_{0,5} = 5, N_{2,0} = 10, N_{2,1} = 10, N_{2,3} = 10, N_{2,4} = 5, N_{3,0} = 10, N_{3,1} = 10, N_{5,1} = 15$ (others are set to zero). For the sports dataset, we set $C = 5$, $\boldsymbol{\alpha} = \langle 1.0, 1.0, 10^{-15}, 10^{-6}, 10^{-6} \rangle$, initialized $\tau = 1$, and used a Dirichlet prior on transition counts $N_{0,1} = 10, N_{2,3} = 20, N_{3,4} = 10$ (others are set to zero). We also manually initialized the weights of some features $\boldsymbol{\beta}$ for both datasets. These values were obtained from preliminary experiments on a smaller sample of the datasets, and updated on the first EM iteration.

[5] Treating each row as a set of words, we can optimize the matching using the Jonker and Volgenant (1987) algorithm. The column evaluation is identical, except that sets of words that are matched are defined by columns. We use the Jaccard similarity—for two sets A and B, $\frac{|A \cap B|}{|A \cup B|}$—for our similarity function, $\text{Sim}(i, j)$.

reference tables, x_{res} and x_{ref}, we can compute:

$$S_{res} = \frac{1}{|x_{res}|} \sum_{i \in x_{res}, j \in x_{ref}: \text{Match}(i,j)=1} \text{Sim}(i,j)$$
$$S_{ref} = \frac{1}{|x_{ref}|} \sum_{i \in x_{res}, j \in x_{ref}: \text{Match}(i,j)=1} \text{Sim}(i,j)$$

where i and j denote rows, $\text{Match}(i,j)$ is one if i and j are matched to each other in the optimal matching or zero otherwise. S_{res} is a precision-like score, and S_{ref} is a recall-like score.[6] Column evaluation is the same, but compares columns instead.

4.3 Baselines

Our simple baseline is an agglomerative clustering algorithm that clusters mentions into entities using the single-linkage criterion. Initially, each unique mention forms its own cluster that we incrementally merge together to form rows in the table. This method requires a similarity score between two clusters. For the politics dataset, we follow Eisenstein et al. (2011) and use the string edit distance between mention strings in each cluster to define the score. For the sports dataset, since mentions contain person and organization named entity types, our score for clustering uses the Jaccard distance between context words of the mentions. However, such clusterings do not produce columns. Therefore, we first match words in mentions of type person against a regular expression to recognize entity attributes from a fixed set of titles and suffixes, and the first, middle and last names. We treat words in mentions of type organization as a single attribute.[7] As we merge clusters together, we arrange words such that all words within a column belong to the same attribute, creating columns as necessary to accomodate multiple similar attributes as a result of merging. We can evaluate the tables produced by each step of the clustering to obtain the entire sequence of response-reference scores.

As a strong baseline, we also compare our approach with the original implementation of the model of Eisenstein et al. (2011), denoted by EEA.

4.4 Results

For both the politics and sports dataset, we run the procedure in §3.3 three times and report the results.

Politics. The results for the politics dataset are shown in Figure 2. Our model consistently outperformed both baselines. We also analyze how much each of our four main extensions (shape features, context information, noise, and first-order column dependencies) to EEA contributes to overall performance by ablating each in turn (also shown in Fig. 2). The best-performing complete model has 415 rows, of which 113 were matched to the reference table. Shape features are useful: removing them was detrimental, and they work even better without column dependencies. Indeed, the best model did not have column dependencies. Removing context features had a strong negative effect, though perhaps this could be overcome with a more carefully tuned initializer.

In row evaluation, the baseline system can achieve a high reference score by creating one entity row per unique string, but as it merges strings, the clusters encompass more word tokens, improving response score at the expense of reference score. With fewer clusters, there are fewer entities in the response table for matching and the response score suffers. Although we use the same seed table in both experiments, the results from EEA are below the baseline curve because it has the additional complexity of inferring the number of columns from data. Our model is simpler in this regard since it assumes that the number of columns is known ($C = 6$). In column evaluation, our method without column dependencies was best.

Sports. The results for the sports dataset are shown in Figure 3. Our best-performing complete model's response table contains 599 rows, of which 561 were matched to the reference table. In row eval-

[6] Eisenstein et al. (2011) used precision and recall for their similarity function. Precision prefers a more compact response row (or column), which unfairly penalizes situations like those of our sports dataset, where rows are heterogeneous (e.g., including people and organizations). Consider a response table made up of two rows: ⟨Kobe, Bryant⟩ and ⟨Los, Angeles, Lakers⟩, and a reference table containing all NBA players and their team names, e.g., ⟨Kobe, Bryant, Los, Angeles, Lakers⟩. Evaluating with the precision similarity function, we will have perfect precision by matching the first row to the reference row for Kobe Bryant and the latter row to any Lakers player. The system is not rewarded for merging the two rows together, even if they are describing the same entity. Our evaluation scores, however, reward the system for accurately filling in more cells.

[7] Note that the baseline system uses NER tags, list of titles and suffixes; which are also provided to our model through the features in §3.1.2.

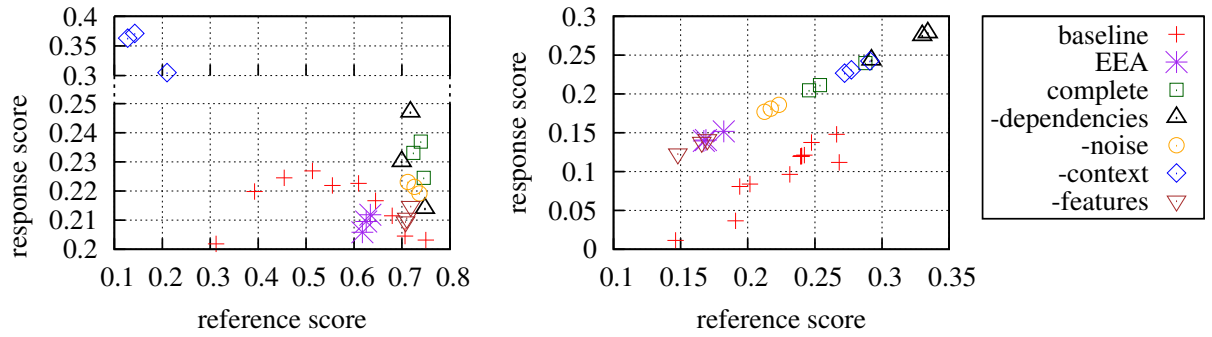

Figure 2: Row (left) and column (right) scores for the politics dataset. For all but "baseline" (clustering), each point denotes a unique sampling run. Note the change in scale in the left plot at $y = 0.25$. For the clustering baseline, points correspond to iterations.

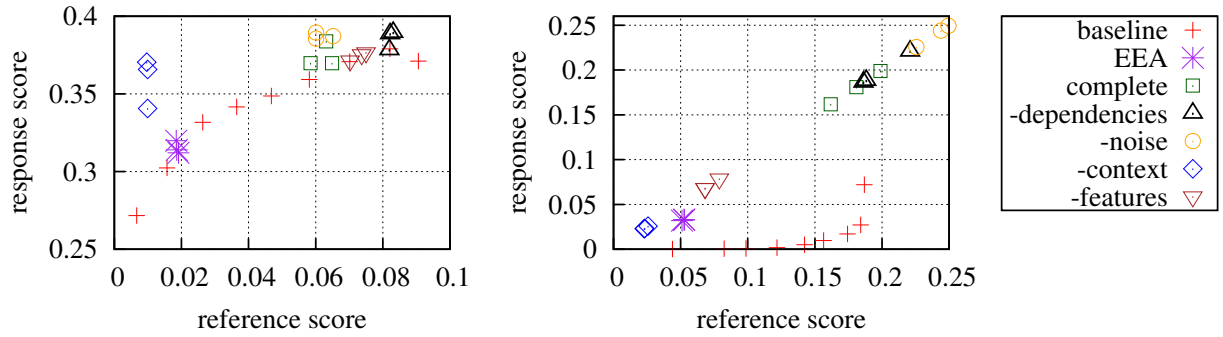

Figure 3: Row (left) and column (right) scores for the sports dataset. Each point denotes a unique sampling run. The reference score is low since the reference set includes all entities in the NBA, NFL, and MLB, but most of them were not mentioned in our dataset.

uation, our model lies above the baseline response-reference score curve, demonstrating its ability to correctly identify and combine player mentions with their team names. Similar to the previous dataset, our model is also substantially better in column evaluation, indicating that it mapped mention words into a coherent set of five columns.

4.5 Discussion

The two datasets illustrate that our model adapts to somewhat different tasks, depending on its input. Furthermore, fixing C (unlike EEA) does appear to have benefits.

In the politics dataset, most of the mentions contain information about people. Therefore, besides canonicalizing named entities, the model also resolves within-document and cross-document coreference, since it assigned a row index for every mention. For example, our model learned that most mentions of John McCain, Sen. John McCain, Sen. McCain, and Mr. McCain refer to the same entity. Table 3 shows a few noteworthy entities from our complete model's output table.

Barack	Obama	Mr.	Sen.		Hussein
Michelle	Obama	Mrs.			
Norm	Coleman		Sen.		
Sarah	Palin	Ms.			
John	McCain	Mr.	Sen.		Hussein

Table 3: A small segment of the output table for the politics dataset, showing a few noteworthy correct (blue) and incorrect (red) examples. Black indicates seeds. Though Ms. is technically correct, there is variation in preferences and conventions. Our data include eight instances of Ms. Palin and none of Mrs. Palin or Mrs. Sarah Palin.

The first entity is an easy example since it only had to complete information provided in the seed table. The model found the correct gender-specific title for Barack Obama, Mr.. The rest of the examples were fully inferred from the data. The model was essentially correct for the second, third, and fourth entities. The last row illustrates a partially erroneous example, in which the model confused the middle name of John McCain, possibly because of a combination of a strong prior to reuse this row and the

Derek	Jeter	New	York	
Ben	Roethlisberger		Pittsburgh	Steelers
Alex	Rodriguez	New	York	Yankees
Michael	Vick		Philadelphia	Eagles
Kevin	Garnett	Los	Angeles	Lakers
Dave	Toub		The	Bears

Table 4: A small segment of the output table for the sports dataset, showing a few noteworthy correct (blue) and incorrect (red) examples. Black indicates seed examples.

introduction of a notion of noise.

In the sports dataset, persons and organizations are mentioned. Recall that success here consists of identifying the correct team for every player. The EEA model is not designed for this and performed poorly. Our model can do better, since it makes use of context information and features, and it can put a person and an organization in one row even though they do not share common words. Table 4 shows a few noteworthy entities from our complete model's output.

Surprisingly, the model failed to infer that Derek Jeter plays for New York Yankees, although mentions of the organization New York Yankees can be found in our dataset. This is because the model did not see enough evidence to put them in the same row. However, it successfully inferred the missing information for Ben Roethlisberger. The next two rows show cases where our model successfully matched players with their teams and put each word token to its respective column. The most frequent error, by far, is illustrated in the fifth row, where a player is matched with a wrong team. Kevin Garnett plays for the Boston Celtics, not the Los Angeles Lakers. It shows that in some cases context information is not adequate, and a possible improvement might be obtained by providing more context to the model. The sixth row is interesting because Dave Toub is indeed affiliated with the Chicago Bears. However, when the model saw a mention token The Bears, it did not have any other columns to put the word token The, and decided to put it in the fourth column although it is not a location. If we added more columns, determiners could become another attribute of the entities that might go into one of these new columns.

5 Related Work

There has been work that attempts to fill predefined templates using Bayesian nonparametrics (Haghighi and Klein, 2010) and automatically learns template structures using agglomerative clustering (Chambers and Jurafsky, 2011). Charniak (2001) and Elsner et al. (2009) focused specifically on names and discovering their structure, which is a part of the problem we consider here. More similar to our work, Eisenstein et al. (2011) introduced a nonparametric Bayesian approach to extract structured databases of entities. A fundamental difference of our approach from any of the previous work is it maximizes conditional likelihood and thus allows beneficial incorporation of arbitrary features.

Our model is focused on the problem of canonicalizing mention strings into their parts, though its r variables (which map mentions to rows) could be interpreted as (within-document and cross-document) coreference resolution, which has been tackled using a range of probabilistic models (Li et al., 2004; Haghighi and Klein, 2007; Poon and Domingos, 2008; Singh et al., 2011). We have not evaluated it as such, believing that further work should be done to integrate appropriate linguistic cues before such an application.

6 Conclusions

We presented an improved probabilistic model for canonicalizing named entities into a table. We showed that the model adapts to different tasks depending on its input and seeds, and that it improves over state-of-the-art performance on two corpora.

Acknowledgements

The authors thank Jacob Eisenstein and Tae Yano for helpful discussions and providing us with the implementation of their model, Tim Hawes for helpful discussions, Naomi Saphra for assistance in developing the gold standard for the politics dataset, and three anonymous reviewers for comments on an earlier draft of this paper. This research was supported in part by the U.S. Army Research Office, Google's sponsorship of the Worldly Knowledge project at CMU, and A*STAR (fellowship to Y. Sim); the contents of this paper do not necessarily reflect the position or the policy of the sponsors, and no official endorsement should be inferred.

References

G. Andrew and J. Gao. 2007. Scalable training of L1-regularized log-linear models. In *Proc. of ICML*.

N. Chambers and D. Jurafsky. 2011. Template-based information extraction without the templates. In *Proc. of ACL-HLT*.

E. Charniak. 2001. Unsupervised learning of name structure from coreference data. In *Proc. of NAACL*.

J. Eisenstein and E. P. Xing. 2010. The CMU 2008 political blog corpus. Technical report, Carnegie Mellon University.

J. Eisenstein, T. Yano, W. W. Cohen, N. A. Smith, and E. P. Xing. 2011. Structured databases of named entities from Bayesian nonparametrics. In *Proc. of EMNLP Workshop on Unsupervised Learning in NLP*.

M. Elsner, E. Charniak, and M. Johnson. 2009. Structured generative models for unsupervised named-entity clustering. In *Proc. of NAACL-HLT*.

J. R. Finkel, T. Grenager, and C. Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proc. of ACL*.

A. Haghighi and D. Klein. 2007. Unsupervised coreference resolution in a nonparametric Bayesian model. In *Proc. of ACL*.

A. Haghighi and D. Klein. 2010. An entity-level approach to information extraction. In *Proc. of ACL Short Papers*.

C. S. Jensen, U. Kjaerulff, and A. Kong. 1995. Blocking Gibbs sampling in very large probabilistic expert system. *International Journal of Human-Computer Studies*, 42(6):647–666.

R. Jonker and A. Volgenant. 1987. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38(4):325–340.

X. Li, P. Morie, and D. Roth. 2004. Identification and tracing of ambiguous names: discriminative and generative approaches. In *Proc. of AAAI*.

D. C. Liu and J. Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming B*, 45(3):503–528.

D. Mochihashi, T. Yamada, and N. Ueda. 2009. Bayesian unsupervised word segmentation with nested Pitman-Yor language modeling. In *Proc. of ACL-IJCNLP*.

C. Morris. 1983. Parametric empirical Bayes inference: Theory and applications. *Journal of the American Statistical Association*, 78(381):47–65.

H. Poon and P. Domingos. 2008. Joint unsupervised coreference resolution with Markov logic. In *Proc. of EMNLP*.

S. Singh, A. Subramanya, F. Pereira, and A. McCallum. 2011. Large-scale cross-document coreference using distributed inference and hierarchical models. In *Proc. of ACL-HLT*.

N. A. Smith and J. Eisner. 2005. Contrastive estimation: training log-linear models on unlabeled data. In *Proc. of ACL*.

R. Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of Royal Statistical Society B*, 58(1):267–288.

G. C. G. Wei and M. A. Tanner. 1990. A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association*, 85(411):699–704.