Collective Segmentation and Labeling of Distant Entities in Information Extraction

Charles Sutton
Department of Computer Science
University of Massachusetts
Amherst, MA 01002
casutton@cs.umass.edu

Andrew McCallum
Department of Computer Science
University of Massachusetts
Amherst, MA 01002
mccallum@cs.umass.edu

Abstract

In information extraction, we often wish to identify all mentions of an entity, such as a person or organization. Traditionally, a group of words is labeled as an entity based only on local information. But information from throughout a document can be useful; for example, if the same word is used multiple times, it is likely to have the same label each time. We present a CRF that explicitly represents dependencies between the labels of pairs of similar words in a document. On a standard information extraction data set, we show that learning these dependencies leads to a 13.7% reduction in error on the field that had caused the most repetition errors.

1 Introduction

Most natural-language systems solve many problem instances in their lifetime. Traditionally, these instances are solved separately, that is, the data is assumed to be independent and identically distributed. But often this assumption does not hold. There is much current interest in collectively making related classification decisions, using dependencies between decisions to increase performance. In Web page classification, for example, Taskar et al. (2002) model the fact that pages that hyperlink to each other are more likely to have the same type, resulting in increased classification performance.

Within information extraction, one important type of error occurs on repeated mentions of the same field. For example, we often wish to identify all mentions of an entity, such as a person or organization, because each mention might contain different useful information. Furthermore, these mentions tend to use similar terms. We can take advantage of this fact by favoring labelings that treat repeated words identically, and by combining features from all occurrences so that the extraction decision can be made based on global information. However, most extraction systems, whether probabilistic or not, do not take advantage of this dependency, instead treating the separate mentions independently.

Recently, Bunescu and Mooney (2004) use a relational Markov network (Taskar et al., 2002) to collectively classify the mentions in a document, achieving increased accuracy by learning dependencies between similar mentions. In their work, however, candidate phrases are extracted heuristically, which can introduce errors if a true entity is not selected as a candidate phrase. Ideally, we would like to perform collective segmentation and labeling simultaneously, so that the system can take into account dependencies between the two tasks. This can be done naturally using probabilistic sequence models.

Traditional probabilistic sequence models, such as HMMs, are generative, in the sense that they represent a joint probability distribution $p(\mathbf{y}, \mathbf{x})$. Because this includes a distribution $p(\mathbf{x})$ over the input features, it is difficult to use arbitrary, overlapping features while maintaining tractability. A

solution to this problem is to model instead the conditional distribution $p(\mathbf{y}|\mathbf{x})$, which is all that is needed for classification anyway. Because the model is conditional, dependencies among the features in $\mathbf{x}$ do not need to be explicitly represented. Popular conditional models include maximum entropy classifiers (Berger et al., 1996) and conditional random fields (Lafferty et al., 2001). Conditionally-trained models have been shown to perform better than generatively-trained models on many tasks, including document classification (Taskar et al., 2002), part-of-speech tagging (Ratnaparkhi, 1996), extraction of data from tables (Pinto et al., 2003), segmentation of FAQ lists (McCallum et al., 2000), and noun-phrase segmentation (Sha and Pereira, 2003).

To perform collective labeling, we need to represent dependencies between distant terms in the input. But this reveals a general limitation of sequence models, whether generatively or conditionally trained. Sequence models usually make a Markov assumption among labels, that is, that any label y_t is independent of all previous labels given its immediate predecessors $y_{t-k} \ldots y_{t-1}$. That is, such models represent dependence only between nearby nodes—for example, between bigrams and trigrams—and cannot represent the higher-order dependencies that arise when identical words occur throughout a document.

In the paper we present a conditional model that collectively segments a document into mentions and classifies the mentions by entity type, taking into account probabilistic dependencies between distant mentions. Although n-gram sequence models cannot represent long-distance dependencies, more general graphical models can, by adding edges between the labels of similar words. We introduce the *skip-chain CRF*, which is a CRF whose structure is a linear chain with additional connections between all pairs of similar words, as shown in Figure 2.

Even though the limitations of n-gram models have been widely recognized within natural language processing, long-distance dependencies are difficult to represent in generative models, because full n-gram models have too many parameters if n is large. We avoid this problem by selecting which skip edges to include based on the input string.

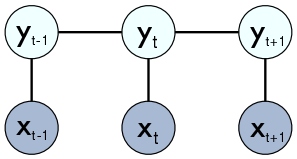

Figure 1: Graphical representation of linear-chain CRF. Although the hidden nodes can depend on observations at any time step, for clarity we have shown links only to observations at the same time step.

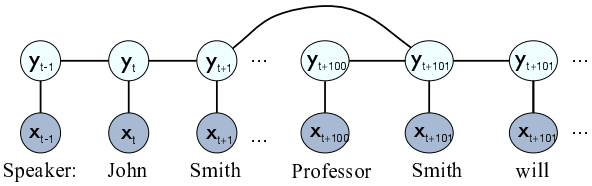

Figure 2: Graphical representation of skip-chain CRF. Identical words are connected because they are likely to have the same label.

This kind of input-specific dependence is difficult to represent in a generative model, which needs to generate the input. In other words, conditional models have been popular because of their flexibility in allowing overlapping features; in this paper, we take advantage of their flexibility in allowing input-specific model structure.

Because our model contains many overlapping loops, exact inference is intractable. For some of our documents, a single probability table in the junction tree would require over 4 GB of memory. Instead, we perform approximate inference using a schedule for loopy belief propagation called tree reparameterization (TRP) (Wainwright et al., 2001).

We evaluate our model on a standard information extraction data set of e-mail seminar announcements. In the field type on which a linear-chain CRF had the most repetition errors, which is speaker name, we show that learning these dependencies with a skip-chain CRF leads to a 13.7% reduction in error. Failure analysis confirms that the reduction in error is due to increased recall on the repeated field mentions.

2 Linear-chain CRFs

Conditional random fields (CRFs) (Lafferty et al., 2001) are undirected graphical models that encode a conditional probability distribution using a given set of features. CRFs are defined as follows. Let $\mathcal{G}$ be an undirected model over sets of random variables $\mathbf{y}$ and $\mathbf{x}$. As a typical special case, $\mathbf{y} = \{y_t\}$ and $\mathbf{x} = \{x_t\}$ for $t = 1, \ldots, T$, so that $\mathbf{y}$ is a labeling of an observed sequence $\mathbf{x}$.

If $C = \{\{\mathbf{y}_c, \mathbf{x}_c\}\}$ is the set of cliques in $\mathcal{G}$, then CRFs define the conditional probability of a state sequence given the observed sequence as:

$$p_\Lambda(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{c \in C} \Phi(\mathbf{y}_c, \mathbf{x}_c), \qquad (1)$$

where Φ is a potential function and the partition function $Z(\mathbf{x}) = \sum_{\mathbf{y}} \prod_{c \in C} \Phi(\mathbf{y}_c, \mathbf{x}_c)$ is a normalization factor over all state sequences for the sequence $\mathbf{x}$. We assume the potentials factorize according to a set of features $\{f_k\}$, which are given and fixed, so that

$$\Phi(\mathbf{y}_c, \mathbf{x}_c) = \exp\left(\sum_k \lambda_k f_k(\mathbf{y}_c, \mathbf{x}_c)\right) \qquad (2)$$

The model parameters are a set of real weights $\Lambda = \{\lambda_k\}$, one weight for each feature.

Most applications use the *linear-chain CRF*, in which a first-order Markov assumption is made on the hidden variables. A graphical model for this is shown in Figure 1. In this case, the cliques of the conditional model are the nodes and edges, so that there are feature functions $f_k(y_t, y_{t+1}, \mathbf{x}, t)$ for each label transition. (Here we write the feature functions as potentially depending on the entire input sequence.) Feature functions can be arbitrary functions of their arguments. For example, a feature function $f_k(y_t, y_{t+1}, \mathbf{x}, t)$ could be a binary test that has value 1 if and only if y_t has the label OTHER, y_{t+1} has the label SPEAKER, and x_t begins with a capital letter.

3 Skip-chain CRFs

3.1 Model

Linear-chain CRFs cannot represent dependencies between distant occurrences of similar words. In this paper, we extend linear-chain CRFs by adding probabilistic connections between similar words, that is, adding edges between them in an undirected linear-chain model such as Figure 2. We call these additional edges *skip edges*. Skip edges can represent dependence between distant nodes, so that similar words can be labeled similarly. Also, the features on skip edges can incorporate information from the context of both endpoints, so that if the label of one endpoint is more certain, it can influence the label of the other.

First, there is the choice of which skip edges to include. We may choose simply to connect identical words, but more generally we could can any pair of words that we believe to be similar, for example, pairs of words that belong to the same stem class, or have small edit distance. Of course, if we simply connected all possible words, inference in the model would require summing over all possible state sequences; no efficient dynamic programming algorithm would be available. So we need to use similarity metrics that result in a sufficiently sparse graph. In this paper, we focus on named-entity recognition, so we connect pairs of identical capitalized words.

This model can be formally defined by adding a second type of potential to the linear-chain model. For an instance $\mathbf{x}$, let $\mathcal{I} = \{(u, v)\}$ be the set of all pairs of sequence positions for which there are skip edges. Then we can write the probability of a label sequence $\mathbf{y}$ given an input $\mathbf{x}$ and parameters Λ as

$$p_\Lambda(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=0}^{T-1} \Phi(y_t, y_{t+1}, \mathbf{x}, t) \prod_{(u,v) \in \mathcal{I}} \Psi(y_u, y_v, \mathbf{x}, u, v), \qquad (3)$$

where Φ is the potential over linear-chain edges, and Ψ is the potential over skip edges. We assume that each of the potentials factorize according to a set of features f_k so that

$$\log \Phi(y_t, y_{t+1}, \mathbf{x}, t) = \sum_k \lambda_k f_k(y_t, y_{t+1}, \mathbf{x}, t) \qquad (4)$$

$$\log \Psi(y_u, y_v, \mathbf{x}, u, v) = \sum_k \lambda'_k f'_k(y_u, y_v, \mathbf{x}, u, v). \qquad (5)$$

Note that each type of clique has its own set of features and weights.

For the short distance edges, we factorize our features as

$$f_k(y_t, y_{t+1}, \mathbf{x}, t) = p_k(y_t, y_{t+1}) q_k(\mathbf{x}, t) \quad (6)$$

where $p_k(y_{t,c})$ is a binary function on the assignment, and $q_k(\mathbf{x}, t)$ is a function solely of the input string, which we call an *input feature*. In general $q_k(\mathbf{x}, t)$ can depend on arbitrary positions of the input string. For example, a useful feature for NER is $q_k(\mathbf{x}, t) = 1$ if and only if x_{t+1} is a capitalized word.

For the skip edges, we factorize our features in in a way that is similar but allows us to combine distant features in the sequence. More specifically, we factorize the features for the skip edges as

$$f'_k(y_u, y_v, \mathbf{x}, u, v) = p'_k(y_u, y_v, u, v) q'_k(\mathbf{x}, u, v) \quad (7)$$

The input features $q'_k(\mathbf{x}, u, v)$ can combine information from the neighborhood of y_u and y_v. For example, one useful feature is $q'_k(\mathbf{x}, u, v) = 1$ if and only if $x_u = x_v =$ "Booth" and $x_{v-1} =$ "Speaker:". This can be a useful feature if the context around x_u, such as "Robert Booth is manager of control engineering...," may not make clear whether or not Robert Booth is presenting a talk, but the context around x_v is clear, such as "Speaker: Robert Booth." [1]

3.2 Inference

The inference problem is, given an input string $\mathbf{x}$, to compute marginal distributions $p(\mathbf{y}_i|\mathbf{x})$ or the most likely (Viterbi) labeling $\arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x})$. Computing marginals is needed for parameter estimation, and the Viterbi labeling is used to label a new sentence. In linear-chain CRFs, exact inference can be performed efficiently by variants of the standard dynamic-programming algorithms for HMMs.

In skip-chain CRFs, inference is much more difficult. Because the loops can be long and overlapping, exact inference is intractable for the data we consider in the next section. Exact inference requires time exponential in the size of a graph's junction tree. For the seminars data, 29 of the 485 instances have a maximum clique size of 10 or greater, and 11 have a maximum clique size of 14 or greater. (The worst instance has a clique with 61 nodes.) For reference, representing a single potential over 14 nodes requires more memory than can be addressed in a 32-bit architecture. Thus, exact inference is not practical for skip-chain CRFs.

Instead, we perform approximate inference using the loopy belief propagation algorithm. Loopy belief propagation is an iterative method that is not guaranteed to converge, but has been found to be reasonably accurate in practice (Murphy et al., 1999). We use an asynchronous tree-based schedule known as TRP (Wainwright et al., 2001).

Loopy belief propagation is a generalization of the forward-backward algorithm for HMMs and linear-chain CRFs, a fact which provides intuition about the skip-chain CRF model. In the forward-backward algorithm, probabilistic information flows only between neighboring nodes, via the α and β recursions. In a skip-chain CRF, belief propagation passes messages not only between neighboring nodes, as in forward-backward, but also along the long distance edges, propagating probabilistic information from distant sequence locations.

3.3 Parameter Estimation

Given training data $\mathcal{D} = \{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^N$, we estimate model parameters $\Lambda = \{\lambda_k\}$ by maximum a posteriori (MAP) estimation. We optimize the the posterior probability

$$\log p(\Lambda|\mathcal{D}) \propto \log p(\mathcal{D}|\Lambda) + \log p(\Lambda) \quad (8)$$
$$= \mathcal{L}(\Lambda) + \log p(\Lambda), \quad (9)$$

where $\mathcal{L}(\Lambda)$ is the log likelihood

$$\mathcal{L}(\Lambda) = \sum_i \log p_\Lambda(\mathbf{y}^{(i)} \mid \mathbf{x}^{(i)}). \quad (10)$$

The derivative of the likelihood with respect to one of the short-distance parameters λ_k is

$$\frac{\partial \mathcal{L}}{\partial \lambda_k} = \sum_i C_k(\mathbf{y}^{(i)}, \mathbf{x}^{(i)}) - E_k(\mathbf{y}^{(i)}, \mathbf{x}^{(i)}) \quad (11)$$

[1] This example is taken from an actual error made by a linear-chain CRF on the seminars data set. We present results from this data set in Section 4.

where C_k and E_k are *constraints* and *expectations* given by

$$C_k(\mathbf{y}^{(i)}, \mathbf{x}^{(i)}) = \sum_t f_k(\vec{y}_t^{(i)}, \vec{y}_{t+1}^{(i)}, x^{(i)}, t) \quad (12)$$

$$E_k(\mathbf{y}^{(i)}, \mathbf{x}^{(i)}) = \sum_t \sum_{\vec{y}_t, \vec{y}_{t+1}} p_\Lambda(\vec{y}_t, \vec{y}_{t+1} \mid x^{(i)}) \\ f_k(\vec{y}_t, \vec{y}_{t+1}, x^{(i)}, t) \quad (13)$$

The derivative $\frac{\partial \mathcal{L}}{\partial \lambda'_k}$ with respect to the long-distance parameters is similar, with λ'_k replacing λ_k and f'_k replacing f_k.

To reduce overfitting, we define a prior $p(\Lambda)$ over parameters. We use a spherical Gaussian prior with mean $\mu = 0$ and covariance matrix $\Sigma = \sigma^2 I$, so that the gradient becomes

$$\frac{\partial p(\Lambda|\mathcal{D})}{\partial \lambda_k} = \frac{\partial \mathcal{L}}{\partial \lambda_k} - \frac{\lambda_k}{\sigma^2}.$$

See Peng and McCallum (2004) for a comparison of different priors for linear-chain CRFs.

The function $p(\Lambda|\mathcal{D})$ is convex, and can be optimized by any number of techniques, as in other maximum-entropy models (Lafferty et al., 2001; Berger et al., 1996). In the results below, we use L-BFGS, which has previously outperformed other optimization algorithms for linear-chain CRFs (Sha and Pereira, 2003; Malouf, 2002).

4 Results

We evaluate skip-chain CRFs on a collection of 485 e-mail messages announcing seminars at Carnegie Mellon University. The messages are annotated with the seminar's starting time, ending time, location, and speaker. This data set is due to Dayne Freitag (Freitag, 1998), and has been used in much previous work.

Often the fields are listed multiple times in the message. For example, the speaker name might be included both near the beginning and later on, in a sentence like "If you would like to meet with Professor Smith..." It can be useful to find both such mentions, because different information can be in the surrounding context of each mention: for example, the first mention might be near an institution affiliation, while the second mentions that Smith is a professor.

$w_t = w$
w_t matches `[A-Z][a-z]+`
w_t matches `[A-Z][A-Z]+`
w_t matches `[A-Z]`
w_t matches `[A-Z]+`
w_t matches `[A-Z]+[a-z]+[A-Z]+[a-z]`
w_t appears in list of first names, last names, honorifics, etc.
w_t appears to be part of a time followed by a dash
w_t appears to be part of a time preceded by a dash
w_t appears to be part of a date
$T_t = T$
$q_k(\mathbf{x}, t+\delta)$ for all k and $\delta \in [-4, 4]$

Table 1: Input features $q_k(\mathbf{x}, t)$ for the seminars data. In the above w_t is the word at position t, T_t is the POS tag at position t, w ranges over all words in the training data, and T ranges over all part-of-speech tags returned by the Brill tagger. The "appears to be" features are based on hand-designed regular expressions that can span several tokens.

Field	Linear-chain	Skip-chain
stime	12.6	17
etime	3.2	5.2
location	6.4	0.6
speaker	30.2	4.8

Table 3: Number of inconsistently mislabeled tokens, that is, tokens that are mislabeled even though the same token is labeled correctly elsewhere in the document. Learning long-distance dependencies reduces this kind of error in the speaker and location fields. Numbers are averaged over 5 folds.

We evaluate a skip-chain CRF with skip edges between identical capitalized words. The motivation for this is that the hardest aspect of this data set is identifying speakers and locations, and capitalized words that occur multiple times in a seminar announcement are likely to be either speakers or locations.

Table 1 shows the list of input features we used. For a skip edge (u, v), the input features we used were simply the disjunction of the input features at u and v, that is,

$$q'_k(\mathbf{x}, u, v) = q_k(\mathbf{x}, u) \oplus q_k(\mathbf{x}, v) \quad (14)$$

where $\oplus$ is binary or. All of our results are averaged over 5-fold cross-validation with an 80/20

System	stime	etime	location	speaker	overall
BIEN (Peshkin and Pfeffer, 2003)	96.0	**98.8**	87.1	76.9	89.7
Linear-chain CRF	**97.5**	97.5	**88.3**	77.3	90.2
Skip-chain CRF	96.7	97.2	88.1	**80.4**	**90.6**

Table 2: Comparison of F_1 performance on the seminars data. The top line gives a dynamic Bayes net that has been previously used on this data set. The skip-chain CRF beats the previous systems in overall F1 and on the speaker field, which has proved to be the hardest field of the four. Overall F1 is simply the average of the F1 scores for the four fields.

split of the data. We report results from both a linear-chain CRF and a skip-chain CRF with the same set of input features.

We calculate precision and recall as[2]

$$P = \frac{\text{\# tokens extracted correctly}}{\text{\# tokens}}$$
$$R = \frac{\text{\# tokens extracted correctly}}{\text{\# tokens}}$$

As usual, we report $F_1 = (2PR)/(P+R)$.

Table 2 compares a skip-chain CRF to a linear-chain CRF and to a dynamic Bayes net used in previous work (Peshkin and Pfeffer, 2003). The skip-chain CRF does much better than all the other systems on speaker, which is the label for which the skip edges would be expected to make the most difference. On the other fields, however, the skip-chain CRF does slightly worse (less than 1% absolute F1).

We expected that the skip-chain CRF would do especially well on the speaker field, because speaker names tend to appear multiple times in a document, and a skip-chain CRF can learn to label the multiple occurrences consistently. To test this hypothesis, we measure the number of *inconsistently mislabeled* tokens, that is, tokens that are mislabeled even though the same token is classified correctly elsewhere in the document. Table 3 compares the number of inconsistently mislabels tokens in the test set between linear-chain and skip-chain CRFs. For the linear-chain CRF, on average 30.2 true speaker tokens are inconsistently mislabeled. Because the linear-chain CRF mislabels 121.6 true speaker tokens, this situation includes 24.7% of the missed speaker tokens. So treating repeated tokens consistently would seem to especially benefit recall on speaker.

In fact, on the speaker field, skip-chain CRFs show a dramatic decrease in inconsistently mislabeled tokens, from 30.2 to 4.8. Because of this, skip-chain CRF has much better recall on speaker tokens than the linear-chain CRF (70.0 R linear chain, 76.8 R skip chain). This explains the increase in F1 between between linear-chain and skip-chain CRFs, for the two have similar precision (average 86.5 P linear chain, 85.1 skip chain).

On the location field, on the other hand, where we might also expect skip-chain CRFs to perform better, there is no benefit. We explain this by observing in Table 3 that inconsistent misclassification occurs much less frequently in this field.

5 Summary

We have demonstrated that modeling long-distance dependencies can be used to obtain more accurate information extraction. Because our model is conditional, its structure is free to depend on the input string. In this paper, we connected pairs of identical words, on the assumption that they would tend to have the same label.

The framework for inference and learning is similar to Relational Markov Networks (Taskar et al., 2002) and Dynamic CRFs (Sutton et al., 2004). Like skip-chain CRFs, RMNs vary their graphical structure based on the input data, but in practice RMNs have been used for classification instead of sequence labeling. DCRFs, on the other hand, are sequence models, but the skip-chain CRF model is not a DCRF, because usually different long-distance edges are used for each input string.

The skip-chain CRF can also be viewed as performing extraction while taking into account

[2]Previous work on this data set has traditionally measured precision and recall per document, that is, from each document the system extracts only one field of each types. In this paper we discuss the problem in which we wish to extract all the mentions in a document, so we cannot compare with this previous work.

a simple form of coreference information, since the reason that identical words are likely to have similar tags is that they are likely to be coreferent. Thus, this model is a step toward joint probabilistic models for extraction and data mining as advocated by McCallum and Jensen (2003). An example of such a joint model is the one of Wellner et al. (2004), which jointly segments citations in research papers and predicts which citations refer to the same paper.

Acknowledgements

This work was supported in part by the Center for Intelligent Information Retrieval, in part by The Central Intelligence Agency, the National Security Agency and National Science Foundation under NSF grant #IIS-0326249, and in part by the Defense Advanced Research Projec ts Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under contract number NBCHD030010. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

References

Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Razvan Bunescu and Raymond J. Mooney. 2004. Collective information extraction with relational Markov networks. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*.

Dayne Freitag. 1998. *Machine Learning for Information Extraction in Informal Domains*. Ph.D. thesis, Carnegie Mellon University.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proc. 18th International Conf. on Machine Learning*.

R. Malouf. 2002. A comparison of algorithms for maximum entropy parameter estimation. In Dan Roth and Antal van den Bosch, editors, *Proceedings of the Sixth Conference on Natural Language Learning (CoNLL-2002)*, pages 49–55.

Andrew McCallum and David Jensen. 2003. A note on the unification of information extraction and data mining using conditional-probability, relational models. In *IJCAI'03 Workshop on Learning Statistical Models from Relational Data*.

Andrew McCallum, Dayne Freitag, and Fernando Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. In *Proc. 17th International Conf. on Machine Learning*, pages 591–598. Morgan Kaufmann, San Francisco, CA.

Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. 1999. Loopy belief propagation for approximate inference: An empirical study. In *Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 467–475.

Fuchun Peng and Andrew McCallum. 2004. Accurate information extraction from research papers using conditional random fields. In *Proceedings of Human Language Technology Conference and North American Chapter of the Association for Computational Linguistics (HLT-NAACL'04)*.

Leonid Peshkin and Avi Pfeffer. 2003. Bayesian information extraction network. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

David Pinto, Andrew McCallum, Xing Wei, and W. Bruce Croft. 2003. Table extraction using conditional random fields. In *Proceedings of the ACM SIGIR*.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proc. of the 1996 Conference on Empirical Methods in Natural Language Proceeding (EMNLP 1996)*.

Fei Sha and Fernando Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of HLT-NAACL 2003*. Association for Computational Linguistics.

Charles Sutton, Khashayar Rohanimanesh, and Andrew McCallum. 2004. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. In *Proceedings of the Twenty-First International Conference on Machine Learning (ICML-2004)*.

Ben Taskar, Pieter Abbeel, and Daphne Koller. 2002. Discriminative probabilistic models for relational data. In *Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI02)*.

M. Wainwright, T. Jaakkola, and A. Willsky. 2001. Tree-based reparameterization for approximate estimation on graphs with cycles. *Advances in Neural Information Processing Systems (NIPS)*.

Ben Wellner, Andrew McCallum, Fuchun Peng, and Michael Hay. 2004. An integrated, conditional model of information extraction and coreference with application to citation graph construction. In *20th Conference on Uncertainty in Artificial Intelligence (UAI)*.