Integrating Probabilistic Extraction Models and Data Mining to Discover Relations and Patterns in Text

Aron Culotta
University of Massachusetts
Amherst, MA 01003
culotta@cs.umass.edu

Andrew McCallum
University of Massachusetts
Amherst, MA 01003
mccallum@cs.umass.edu

Jonathan Betz
Google, Inc.
New York, NY 10018
jtb@google.com

Abstract

In order for relation extraction systems to obtain human-level performance, they must be able to incorporate relational patterns inherent in the data (for example, that one's sister is likely one's mother's daughter, or that children are likely to attend the same college as their parents). Hand-coding such knowledge can be time-consuming and inadequate. Additionally, there may exist many interesting, unknown relational patterns that both improve extraction performance and provide insight into text. We describe a probabilistic extraction model that provides mutual benefits to both "top-down" relational pattern discovery and "bottom-up" relation extraction.

1 Introduction

Consider these four sentences:

1. George W. Bush's father is George H. W. Bush.
2. George H. W. Bush's sister is Nancy Bush Ellis.
3. Nancy Bush Ellis's son is John Prescott Ellis.
4. John Prescott Ellis analyzed George W. Bush's campaign.

We would like to build an automated system to extract the set of relations shown in Figure 1.

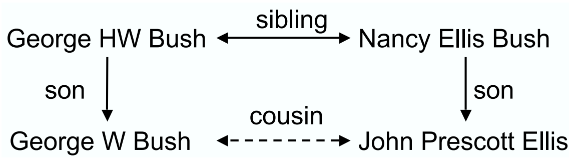

Figure 1: Bush family tree

State of the art extraction algorithms may be able to detect the *son* and *sibling* relations from local language clues. However, the *cousin* relation is only *implied* by the text and requires additional knowledge to be extracted. Specifically, the system requires knowledge of familial relation patterns.

One could imagine a system that accepts such rules as input (e.g. *cousin = father's sister's son*) and applies them to extract implicit relations. However, exhaustively enumerating all possible rules can be tedious and incomplete. More importantly, many relational patterns unknown *a priori* may both improve extraction accuracy and uncover informative trends in the data (e.g. that children often adopt the religion of their parents). Indeed, the goal of data mining is to learn such patterns from database regularities. Since these patterns will not always hold, we would like to handle them probabilistically.

We propose an integrated supervised machine learning method that learns both contextual and relational patterns to extract relations. In particular, we construct a linear-chain conditional random field (Lafferty et al., 2001; Sutton and McCallum, 2006) to extract relations from biographical texts while simultaneously discovering interesting relational patterns that improve extraction performance.

Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL, pages 296–303,
New York, June 2006. ©2006 Association for Computational Linguistics

2 Related Work

This work can be viewed as a step toward the integration of information extraction and data mining technology, a direction of growing interest. Nahm and Mooney (2000) present a system that mines association rules from a database constructed from automatically extracted data, then applies these learned rules to improve data field recall without revisiting the text. Our work attempts to more tightly integrate the extraction and mining tasks by learning relational patterns that can be included probabilistically into extraction to improve its accuracy; also, our work focuses on mining from relational graphs, rather than single-table databases.

McCallum and Jensen (2003) argue the theoretical benefits of an integrated probabilistic model for extraction and mining, but do not construct such a system. Our work is a step in the direction of their proposal, using an inference procedure based on a closed-loop iteration between extraction and relational pattern discovery.

Most other work in this area mines raw text, rather than a database automatically populated via extraction (Hearst, 1999; Craven et al., 1998).

This work can also be viewed as part of a trend to perform joint inference across multiple language processing tasks (Miller et al., 2000; Roth and tau Yih, 2002; Sutton and McCallum, 2004).

Finally, using relational paths between entities is also examined in (Richards and Mooney, 1992) to escape local maxima in a first-order learning system.

3 Relation Extraction as Sequence Labeling

Relation extraction is the task of discovering semantic connections between entities. In text, this usually amounts to examining pairs of entities in a document and determining (from local language cues) whether a relation exists between them. Common approaches to this problem include pattern matching (Brin, 1998; Agichtein and Gravano, 2000), kernel methods (Zelenko et al., 2003; Culotta and Sorensen, 2004; Bunescu and Mooney, 2006), logistic regression (Kambhatla, 2004), and augmented parsing (Miller et al., 2000).

The pairwise classification approach of kernel methods and logistic regression is commonly a two-phase method: first the entities in a document are identified, then a relation type is predicted for each pair of entities. This approach presents at least two difficulties: (1) enumerating all pairs of entities, even when restricted to pairs within a sentence, results in a low density of positive relation examples; and (2) errors in the entity recognition phase can propagate to errors in the relation classification stage. As an example of the latter difficulty, if a person is mislabeled as a company, then the relation classifier will be unsuccessful in finding a *brother* relation, despite local evidence.

We avoid these difficulties by restricting our investigation to *biographical texts*, e.g. encyclopedia articles. A biographical text mostly discusses one entity, which we refer to as the *principal entity*. We refer to other mentioned entities as *secondary entities*. For each secondary entity, our goal is to predict what relation, if any, it has to the principal entity.

This formulation allows us to treat relation extraction as a *sequence labeling* task such as *named-entity recognition* or *part-of-speech tagging*, and we can now apply models that have been successful on those tasks. By anchoring one argument of relations to be the principal entity, we alleviate the difficulty of enumerating all pairs of entities in a document. By converting to a sequence labeling task, we fold the entity recognition step into the relation extraction task. There is no initial pass to label each entity as a person or company. Instead, an entity's label is its relation to the principal entity. Below is an example of a labeled article:

George W. Bush

George is the son of $\underbrace{\text{George H. W. Bush}}_{\textbf{father}}$

and $\underbrace{\text{Barbara Bush}}_{\textbf{mother}}$.

Additionally, by using a sequence model we can capture the dependence between adjacent labels. For example, in our data it is common to see phrases such as "son of the Republican president George H. W. Bush" for which the labels *politicalParty*, *jobTitle*, and *father* occur consecutively. Sequence models are specifically designed to handle these kinds of dependencies. We now discuss the details of our extraction model.

3.1 Conditional Random Fields

We build a model to extract relations using linear-chain conditional random fields (CRFs) (Lafferty et al., 2001; Sutton and McCallum, 2006). CRFs are undirected graphical models (i.e. Markov networks) that are discriminatively-trained to maximize the conditional probability of a set of output variables $\mathbf{y}$ given a set of input variables $\mathbf{x}$. This conditional distribution has the form

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

where ϕ are potential functions parameterized by Λ and $Z_x = \sum_\mathbf{y} \prod_{c \in C} \phi(\mathbf{y}_c, \mathbf{x}_c)$ is a normalization factor. Assuming ϕ_c factorizes as a log-linear combination of arbitrary features computed over clique c, then $\phi_c(\mathbf{y}_c, \mathbf{x}_c; \Lambda) = \exp\left(\sum_k \lambda_k f_k(\mathbf{y}_c, \mathbf{x}_c)\right)$, where f is a set of arbitrary *feature functions* over the input, each of which has an associate model parameter λ_k. Parameters $\Lambda = \{\lambda_k\}$ are a set of real-valued weights typically estimated from labeled training data by maximizing the data likelihood function using gradient ascent.

In these experiments, we make a first-order Markov assumption on the dependencies among $\mathbf{y}$, resulting in a linear-chain CRF.

4 Relational Patterns

The modeling flexibility of CRFs permits the feature functions to be complex, overlapping features of the input without requiring additional assumptions on their inter-dependencies. In addition to common language features (e.g. neighboring words and syntactic information), in this work we explore features that cull relational patterns from a database of entities.

As described in the introductory example (Figure 1), context alone is often insufficient to extract relations. Even in simpler examples, it may be the case that modeling relational patterns can improve extraction accuracy.

To capture this evidence, we compute features from a database to indicate relational connections between entities, similar to the *relational pathfinding* performed in Richards and Mooney (1992).

Imagine that the four sentence example about the Bush family is included in a training set, and the entities are labeled with their correct relations. In this case, the *cousin* relation in sentence 4 would also be labeled. From this data, we can create a relational database that contains the relations in Figure 1.

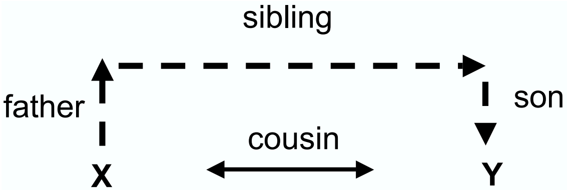

Figure 2: A feature template for the *cousin* relation.

Assume sentence 4 comes from a biography about John Ellis. We calculate a feature for the entity George W. Bush that indicates the path from John Ellis to George W. Bush in the database, annotating each edge in the path with its relation label; i.e. *father-sibling-son*. By abstracting away the actual entity names, we have created a *cousin* template feature, as shown in Figure 2.

By adding these relational paths as features to the model, we can learn interesting relational patterns that may have low precision (e.g. "people are likely to be friends with their classmates") without hampering extraction performance. This is in contrast to the system described in Nahm and Mooney (2000), in which patterns are induced from a noisy database and then applied directly to extraction. In our system, since each learned path has an associated weight, it is simply another piece of evidence to help the extractor. Low precision patterns may have lower weights than high precision patterns, but they will still influence the extractor.

A nice property of this approach is that examining highly weighted patterns can provide insight into regularities of the data.

4.1 Feature Induction

During CRF training, weights are learned for each relational pattern. Patterns that increase extraction performance will receive higher weights, while patterns that have little effect on performance will receive low weights.

We can explore the space of possible conjunctions of these patterns using feature induction for CRFs, as described in McCallum (2003). Search through the large space of possible conjunctions is guided by adding features that are estimated to increase the likelihood function most.

When feature induction is used with relational patterns, we can view this as a type of data mining, in which patterns are created based on their influence on an extraction model. This is similar to work by Dehaspe (1997), where *inductive logic programming* is embedded as a feature induction technique for a maximum entropy classifier. Our work restricts induced features to conjunctions of base features, rather than using first-order clauses. However, the patterns we learn are based on information extracted from natural language.

4.2 Iterative Database Construction

The top-down knowledge provided by data mining algorithms has the potential to improve the performance of information extraction systems. Conversely, bottom-up knowledge generated by extraction systems can be used to populate a large database, from which more top-down knowledge can be discovered. By carefully communicating the uncertainty between these systems, we hope to iteratively expand a knowledge base, while minimizing fallacious inferences.

In this work, the top-down knowledge consists of relational patterns describing the database path between entities in text. The uncertainty of this knowledge is handled by associating a real-valued CRF weight with each pattern, which increases when the pattern is predictive of other relations. Thus, the extraction model can adapt to noise in these patterns.

Since we also desire to extract relations between entities that appear in text but not in the database, we first populate the database with relations extracted by a CRF that does not use relational patterns. We then do further extraction with a CRF that incorporates the relational patterns found in this automatically generated database. In this manner, we create a closed-loop system that alternates between bottom-up extraction and top-down pattern discovery. This approach can be viewed as a type of alternating optimization, with analogies to formal methods such as expectation-maximization.

The uncertainty in the bottom-up extraction step is handled by estimating the confidence of each extraction and pruning the database to remove entries with low confidence. One of the benefits of a probabilistic extraction model is that confidence estimates can be straight-forwardly obtained. Culotta and McCallum (2004) describe the *constrained forward-backward* algorithm to efficiently estimate the conditional probability that a segment of text is correctly extracted by a CRF.

Using this algorithm, we associate a confidence value with each relation extracted by the CRF. This confidence value is then used to limit the noise introduced by incorrect extractions. This differs from Nahm and Mooney (2000) and Mooney and Bunescu (2005), in which standard decision tree rule learners are applied to the unfiltered output of extraction.

4.3 Extracting Implicit Relations

An *implicit relation* is one that does not have direct contextual evidence, for example the *cousin* relation in our initial example. Implicit relations generally require some background knowledge to be detected, such as relational patterns (e.g. rules about familial relations). These are the sorts of relations on which current extraction models perform most poorly.

Notably, these are exactly the sorts of relations that are likely to have the biggest impact on information access. A system that can accurately discover knowledge that is only implied by the text will dramatically increase the amount of information a user can uncover, effectively providing access to the *implications* of a corpus.

We argue that integrating top-down and bottom-up knowledge discovery algorithms discussed in Section 4.2 can enable this technology. By performing pattern discovery in conjunction with information extraction, we can collate facts from multiple sources to infer new relations. This is an example of *cross-document fusion* or *cross-document information extraction*, a growing area of research transforming raw extractions into usable knowledge bases (Mann and Yarowsky, 2005; Masterson and Kushmerik, 2003).

5 Experiments

5.1 Data

We sampled 1127 paragraphs from 271 articles from the online encyclopedia Wikipedia[1] and labeled a to-

[1] http://www.wikipedia.org

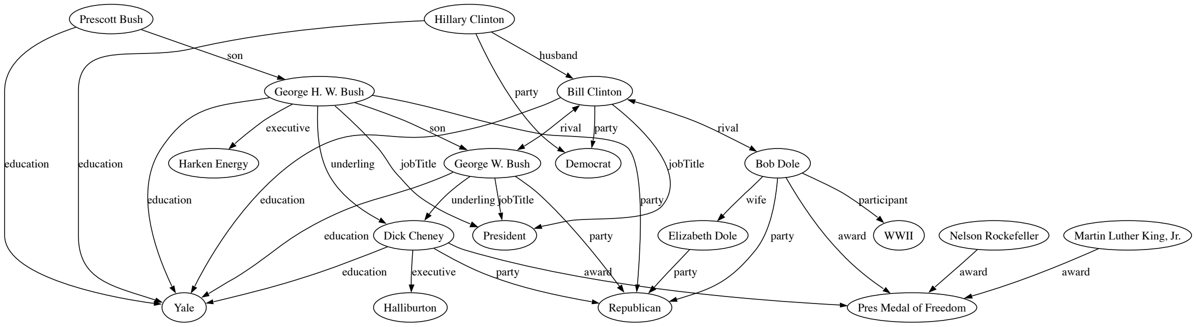

Figure 3: An example of the connectivity of the entities in the data.

birthday	birth year	death day
death year	nationality	visited
birth place	death place	religion
job title	member of	cousin
friend	discovered	education
employer	associate	opus
participant	influence	award
brother	wife	supported idea
executive of	political party	supported person
founder	son	father
rival	underling	superior
role	inventor	husband
grandfather	sister	brother-in-law
nephew	mother	daughter
granddaughter	grandson	great-grandson
grandmother	rival organization	owner of
uncle	descendant	ancestor
great-grandfather	aunt	

Table 1: The set of labeled relations.

tal of 4701 relation instances. In addition to a large set of person-to-person relations, we also included links between people and organizations, as well as biographical facts such as *birthday* and *jobTitle*. In all, there are 53 labels in the training data (Table 1).

We sample articles that result in a high density of interesting relations by choosing, for example, a collection of related family members and associates. Figure 3 shows a small example of the type of connections in the data. We then split the data into training and testing sets (70-30 split), attempting to separate the entities into connected components. For example, all Bush family members were placed in the training set, while all Kennedy family members were placed in the testing set. While there are still occasional paths connecting entities in the training set to those in the test set, we believe this methodology reflects a typical real-world scenario in which we would like to extend an existing database to a different, but slightly related, domain.

The structure of the Wikipedia articles somewhat simplifies the extraction task, since important entities are hyper-linked within the text. This provides an automated way to detect entities in the text, although these entities are not classified by type. This also allows us to easily construct database queries, since we can reason at the entity level, rather than the token level. (Although, see Sarawagi and Cohen (2004) for extensions of CRFs that model the entity length distribution.) The results we report here are constrained to predict relations only for hyperlinked entities. Note that despite this property, we still desire to use a sequence model to capture the dependencies between adjacent labels.

We use the MALLET CRF implementation (McCallum, 2002) with the default regularization parameters.

Based on initial experiments, we restrict relational path features to length two or three. Paths of length one will learn trivial paths and can lead to overfitting. Paths longer than three can increase computational costs without adding much new information.

In addition to the relational pattern features described in Section 4, the list of local features includes **context words** (such as the token identity within a 6 word window of the target token), **lexicons** (such as whether a token appears in a list of cities, people, or companies), **regular expressions** (such as whether the token is capitalized or contains digits or punctuation), **part-of-speech** (predicted by a CRF that was trained separately for part of speech tagging), **prefix/suffix** (such as whether a word ends in *-ed* or begins with *ch-*), and **offset conjunctions** (combinations of adjacent features within a window of size six).

	ME	CRF_0	CRF_r	$CRF_r0.9$	$CRF_r0.5$	CRF_t	$CRF_t0.5$
F1	.5489	.5995	.6100	.6008	**.6136**	.6791	.6363
P	.6475	.7019	.6799	**.7177**	.7095	.7553	.7343
R	.4763	.5232	**.5531**	.5166	.5406	.6169	.5614

Table 2: Results comparing the relative benefits of using relational patterns in extraction.

5.2 Extraction Results

We evaluate performance by calculating the precision (**P**) and recall (**R**) of extracted relations, as well as the **F1** measure, which is the harmonic mean of precision and recall.

CRF_0 is the conditional random field constructed without relational features. Results for CRF_0 are displayed in the second column of Table 2. *ME* is a maximum entropy classifier trained on the same feature set as CRF_0. The difference between these two models is that CRF_0 models the dependence of relations that appear consecutively in the text. The superior performance of CRF_0 suggests that this dependence is important to capture.

The remaining models incorporate the relational patterns described in Section 4. We compare three different confidence thresholds for the construction of the initial testing database, as described in Section 4.2. CRF_r uses no threshold, while $CRF_r0.9$ and $CRF_r0.5$ restrict the database to extractions with confidence greater than 0.9 and 0.5, respectively.

As shown by comparing CRF_0 and CRF_r in Table 2, the relational features constructed from the database with no confidence threshold provides a considerable boost in recall (reducing error by 7%), at the cost of a decrease in precision. Here we see the effect of making fallacious inferences on a noisy database.

In column four, we see the opposite effect for the overly conservative threshold of $CRF_r0.9$. Here, precision improves slightly over CRF_0, and considerably over CRF_r (12% error reduction), but this is accompanied by a drop in recall (8% reduction).

Finally, in column five, a confidence of 0.5 results in the best F1 measure (a 3.5% error reduction over CRF_0). $CRF_r0.5$ also obtains better recall and precision than CRF_0, reducing recall error by 3.6%, precision error by 2.5%.

Comparing the performance on different relation types, we find that the biggest increase from CRF_0 to $CRF_r0.5$ is on the *memberOf* relation, for which the F1 score improves from 0.4211 to 0.6093. We conjecture that the reason for this is that the patterns most useful for the *memberOf* label contain relations that are well-detected by the first-pass CRF. Also, the local language context seems inadequate to properly extract this relation, given the low performance of CRF_0.

To better gauge how much relational pattern features are affected by errors in the database, we run two additional experiments for which the relational features are fixed to be correct. That is, imagine that we construct a database from the true labeling of the testing data, and create the relational pattern features from this database. Note that this does not trivialize the problem, since there are no relational path features of length one (e.g., if X is the wife of Y, there will be no feature indicating this).

We construct two experiments under this scheme, one where the entire test database is used (CRF_t), and another where only half the relations are included in the test database, selected uniformly at random ($CRF_t0.5$).

Column six shows the improvements enabled by using the complete testing database. More interestingly, column seven shows that even with only half the database accurately known, performance improves considerably over both *CRF* and $CRF_r0.5$. A realistic scenario for $CRF_t0.5$ is a semi-automated system, in which a partially-filled database is used to bootstrap extraction.

5.3 Mining Results

Comparing the impact of discovered patterns on extraction is a way to objectively measure mining performance. We now give a brief subjective evaluation of the learned patterns. By examining relational patterns with high weights for a particular label, we can glean some regularities from our dataset. Examples of such patterns are in Table 3.

Relation	Relational Path Feature
mother	father → wife
cousin	mother → husband → nephew
friend	education → student
education	father → education
boss	boss → son
memberOf	grandfather → memberOf
rival	politicalParty → member → rival

Table 3: Examples of highly weighted relational patterns.

From the familial relations in our training data, we are able to discover many equivalences for mothers, cousins, grandfathers, and husbands. In addition to these high precision patterns, the system also generates interesting, low precision patterns. Row 3-7 of Table 3 can be summarized by the following generalizations: friends tend to be classmates; children of alumni often attend the same school as their parents; a boss' child often becomes the boss; grandchildren are often members of the same organizations as their grandparents; and rivals of a person from one political party are often rivals of other members of the same political party. While many of these patterns reflect the high concentration of political entities and familial relations in our training database, many will have applicability across domains.

5.4 Implicit Relations

It is difficult to measure system performance on implicit relations, since our labeled data does not distinguish between explicit and implicit relations. Additionally, accurately labeling all implicit relations is challenging even for a human annotator.

We perform a simple exploratory analysis to determine how relational patterns can help discover implicit relations. We construct a small set of synthetic sentences for which CRF$_0$ successfully extracts relations using contextual features. We then add sentences with slightly more ambiguous language and measure whether CRF$_r$ can overcome this ambiguity using relational pattern features.

For example, we create an article about an entity named "Bob Smith" that includes the sentences "His brother, Bill Smith, was a biologist" and "His *companion*, Bill Smith, was a biologist." CRF$_0$ successfully returns the brother relation in the first sentence, but not the second. After a fact is added to the database that says Bob and Bill have a brother in common named John, CRF$_r$ is able to correctly label the second sentence in spite of the ambiguous word "companion," because CRF$_0$ has a highly-weighted relational pattern feature for brother.

Similar behavior is observed for low precision patterns like "associates tend to win the same awards." A synthetic article for the entity "Tom Jones" contains the sentences "He was awarded the Pulitzer Prize in 1998" and "Tom got the Pulitzer Prize in 1998." Because CRF$_0$ is highly-reliant on the presence of the verb "awarded" or "won" to indicate a prize fact, it fails to label the second sentence correctly. After the database is augmented to include the fact that Tom's associate Jill received the Pulitzer Prize, CRF$_r$ labels the second sentence correctly.

However, we also observed that CRF$_r$ still requires some contextual clues to extract implicit relations. For example, if the Tom Jones article instead contains the sentence "The Pulitzer Prize was awarded to him in 1998," neither CRF labels the prize fact correctly, since this passive construction is rarely seen in the training data.

We conclude from this brief analysis that relational patterns used by CRF$_r$ can help extract implicit relations when (1) the database contains accurate relational information, and (2) the sentence contains limited contextual clues. Since relational patterns are treated only as additional features by CRF$_r$, they are generally not powerful enough to overcome a complete absence of contextual clues. From this perspective, relational patterns can be seen as enhancing the signal from contextual clues. This differs from deterministically applying learned rules independent of context, which may boost recall at the cost of precision.

6 Conclusions and Future Work

We have shown that integrating pattern discovery with relation extraction can lead to improved performance on each task.

In the future, we wish to explore extending this methods to larger datasets, where we expect relational patterns to be even more interesting. Also, we plan to improve upon iterative database construction by performing joint inference among distant relations in an article. Inference in these highly-connected models will likely require approximate methods. Additionally, we wish to focus on extracting implicit relations, dealing more formally with the precision-recall trade-off inherent in applying noisy rules to improve extraction.

7 Acknowledgments

Thanks to the Google internship program, and to Charles Sutton for providing the CRF POS tagger. This work was supported in part by the Center for Intelligent Information Retrieval, in part by U.S. Government contract #NBCH040171 through a subcontract with BBNT Solutions LLC, 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 Projects 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

Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM International Conference on Digital Libraries*.

Sergey Brin. 1998. Extracting patterns and relations from the world wide web. In *WebDB Workshop at 6th International Conference on Extending Database Technology*.

Razvan Bunescu and Raymond Mooney. 2006. Subsequence kernels for relation extraction. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*. MIT Press, Cambridge, MA.

Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew K. McCallum, Tom M. Mitchell, Kamal Nigam, and Seán Slattery. 1998. Learning to extract symbolic knowledge from the World Wide Web. In *Proceedings of AAAI-98, 15th Conference of the American Association for Artificial Intelligence*, pages 509–516, Madison, US. AAAI Press, Menlo Park, US.

Aron Culotta and Andrew McCallum. 2004. Confidence estimation for information extraction. In *Human Langauge Technology Conference (HLT 2004)*, Boston, MA.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *ACL*.

L. Dehaspe. 1997. Maximum entropy modeling with clausal constraints. In *Proceedings of the Seventh International Workshop on Inductive Logic Programming*, pages 109–125, Prague, Czech Republic.

M. Hearst. 1999. Untangling text data mining. In *37th Annual Meeting of the Association for Computational Linguistics*.

Nanda Kambhatla. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In *ACL*.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA.

Gideon Mann and David Yarowsky. 2005. Multi-field information extraction and cross-document fusion. In *ACL*.

D. Masterson and N. Kushmerik. 2003. Information extraction from multi-document threads. In *ECML-2003: Workshop on Adaptive Text Extraction and Mining*, pages 34–41.

Andrew McCallum and David Jensen. 2003. A note on the unification of information extraction and data mining using conditional-probability, relational models. In *IJCAI03 Workshop on Learning Statistical Models from Relational Data*.

Andrew McCallum. 2002. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu.

Andrew McCallum. 2003. Efficiently inducing features of conditional random fields. In *Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI03)*.

Scott Miller, Heidi Fox, Lance A. Ramshaw, and Ralph Weischedel. 2000. A novel use of statistical parsing to extract information from text. In *ANLP*.

Raymond J. Mooney and Razvan Bunescu. 2005. Mining knowledge from text using information extraction. *SigKDD Explorations on Text Mining and Natural Language Processing*.

Un Yong Nahm and Raymond J. Mooney. 2000. A mutually beneficial integration of data mining and information extraction. In *AAAI/IAAI*.

Bradley L. Richards and Raymond J. Mooney. 1992. Learning relations by pathfinding. In *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*, pages 50–55, San Jose, CA.

Dan Roth and Wen tau Yih. 2002. Probabilistic reasoning for entity and relation recognition. In *COLING*.

Sunita Sarawagi and William W. Cohen. 2004. Semi-markov conditional random fields for information extraction. In *NIPS 04*.

Charles Sutton 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)*.

Charles Sutton and Andrew McCallum. 2006. An introduction to conditional random fields for relational learning. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press. To appear.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106.