Joint Inference in Information Extraction

Hoifung Poon **Pedro Domingos**
Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195-2350, U.S.A.
{*hoifung, pedrod*}*@cs.washington.edu*

Abstract

The goal of information extraction is to extract database records from text or semi-structured sources. Traditionally, information extraction proceeds by first segmenting each candidate record separately, and then merging records that refer to the same entities. While computationally efficient, this approach is suboptimal, because it ignores the fact that segmenting one candidate record can help to segment similar ones. For example, resolving a well-segmented field with a less-clear one can disambiguate the latter's boundaries. In this paper we propose a joint approach to information extraction, where segmentation of all records and entity resolution are performed together in a single integrated inference process. While a number of previous authors have taken steps in this direction (e.g., Pasula *et al.* (2003), Wellner *et al.* (2004)), to our knowledge this is the first fully joint approach. In experiments on the CiteSeer and Cora citation matching datasets, joint inference improved accuracy, and our approach outperformed previous ones. Further, by using Markov logic and the existing algorithms for it, our solution consisted mainly of writing the appropriate logical formulas, and required much less engineering than previous ones.

Introduction

AI systems that process sensory input typically have a pipeline architecture: the output of each stage is the input of the next, and there is no feedback from later stages to earlier ones. Although this makes the systems comparatively easy to assemble, and helps to contain computational complexity, it comes at a high price: errors accumulate as information progresses through the pipeline, and an error once made cannot be corrected. Examples of this can be found in information extraction, natural language processing, speech recognition, vision, robotics, etc. Compounding this, systems typically process one object at a time, which again has the benefit of being simpler, but misses opportunities for information from one object to help in processing another.

Ideally, we would like to perform *joint inference* for all relevant tasks simultaneously. Recent progress in probabilistic inference and machine learning has begun to make this possible. For example, the burgeoning field of statistical relational learning is concerned with learning from data points that are not independent and identically distributed (Getoor & Taskar 2007). In natural language processing, joint inference has become a topic of keen interest (e.g., Sutton *et al.* (2006), Punyakanok *et al.* (2005)). However, setting up a joint inference model is usually very complex, and the computational cost of running it can be prohibitive. Further, joint inference can sometimes hurt accuracy, by increasing the number of paths by which errors can propagate. As a result, fully joint approaches are still rare, and even partly-joint ones require much engineering.

In information extraction, the global input is free text or semi-structured sources (e.g., Web pages, emails, or citation lists), and the global output is database records. Two key stages in the pipeline are *segmentation*, which locates candidate fields, and *entity resolution*, which identifies duplicate records. Typically, each candidate record or field is segmented separately, and the output of this process is then passed to the entity resolution stage. To illustrate the shortcomings of this approach, consider the problem of extracting database records from the following two citations in CiteSeer (Lawrence *et al.* 1999):

Minton, S(1993 b). Integrating heuristics for constraint satisfaction problems: A case study. In: Proceedings AAAI.

S. Minton Integrating heuristics for constraint satisfaction problems: A case study. In AAAI Proceedings, 1993.

In the first citation, author and title are clearly separated by a date and period, and extracting them is fairly straightforward. In the second one, there is no clear author-title boundary, and correctly pinpointing it seems very difficult. Large quantities of labeled training data and an extensive lexicon could help, but they are expensive to obtain, and even then are far from a guarantee of success. However, if we notice that the two citations are coreferent and the title of the first one begins with the substring "Integrating heuristics for," we can hypothesize that the title of the second one also begins with this substring, allowing us to correctly segment it.

In this paper, we develop an approach to information extraction that is capable of performing this type of inference, and show that it does indeed improve extraction accuracy. The approach is based on Markov logic, a representation language that combines probabilistic graphical models and first-order logic (Richardson & Domingos 2006). Markov logic enables us to concisely specify very complex models, and is ideally suited for solving joint inference problems. Efficient learning and inference algorithms for it are available in open-source software, and our solution exploits them.

A number of previous authors have taken steps toward joint inference in information extraction, but to our knowledge no fully joint approach has been proposed. For example, Bunescu & Mooney (2004) applied joint segmentation to protein name extraction, but did not do entity resolution. In citation matching, Pasula *et al.* (2003) developed a "collective" model, but performed segmentation in a pre-processing stage, allowing boundaries to occur only at punctuation marks. Wellner *et al.* (2004) extended the pipeline model by passing uncertainty from the segmentation phase to the entity resolution phase (as opposed to just passing the "best guess"), and by including a one-time step from resolution to segmentation, but did not "close the loop" by repeatedly propagating information in both directions. (Due to this, they explicitly refrained from calling their model "joint.") Both of these models required considerable engineering. Pasula *et al.* combined several models with separately learned parameters, a number of hard-wired components, and data from a variety of sources (including a database of names from the 2000 US Census, manually-segmented citations, and a large AI BibTex bibliography). Wellner *et al.* assembled a number of different learning and inference algorithms and performed extensive feature engineering, including computing a variety of string edit distances, TF-IDF measures, global features of the citations, and combinations thereof. As we will see, our approach is considerably simpler, and outperforms both of these.

We begin by briefly reviewing the necessary background in Markov logic. We then describe our MLN for joint citation matching and introduce two kinds of joint inference that can help segmentation. Finally, we present our experiments and results.

Markov Logic

Markov logic is a probabilistic extension of finite first-order logic (Richardson & Domingos 2006). A *Markov logic network (MLN)* is a set of weighted first-order clauses. Together with a set of constants, it defines a Markov network with one node per ground atom and one feature per ground clause. The weight of a feature is the weight of the first-order clause that originated it. The probability of a state x in such a network is given by $P(x) = (1/Z)\exp\left(\sum_i w_i f_i(x)\right)$, where Z is a normalization constant, w_i is the weight of the ith clause, $f_i = 1$ if the ith clause is true, and $f_i = 0$ otherwise.

Markov logic makes it possible to compactly specify probability distributions over complex relational domains. We used the learning and inference algorithms provided in the open-source Alchemy package (Kok *et al.* 2006). In particular, we performed inference using the MC-SAT algorithm (Poon & Domingos 2006), and discriminative weight learning using the voted perceptron algorithm (Singla & Domingos 2005).

MC-SAT is a "slice sampling" Markov chain Monte Carlo algorithm. It uses a combination of satisfiability testing and simulated annealing to sample from the slice. The advantage of using a satisfiability solver (WalkSAT) is that it efficiently finds isolated modes in the distribution, and as a result the Markov chain mixes very rapidly. The slice sampling scheme ensures that detailed balance is (approximately) preserved. MC-SAT is orders of magnitude faster than previous MCMC algorithms like Gibbs sampling and simulated tempering, and makes efficient joint inference possible on a scale that was previously out of reach. In this paper, we use it to perform inference over hundreds of thousands of variables in tens of minutes.

The voted perceptron algorithm optimizes weights by doing gradient descent on the conditional log-likelihood of the query atoms given the evidence ones. The gradient with respect to a weight is the difference between the number of times the corresponding clause is true in the data and its expectation according to the MLN. The latter is approximated by finding a mode of the distribution using a weighted satisfiability solver (MaxWalkSAT) and counting true clause groundings there. To combat overfitting, the weights returned are averages over the learning run.

We modified this algorithm in two ways. First, we used MC-SAT instead of MaxWalkSAT to compute expected counts. Because the distribution can contain many modes, MC-SAT tends to give more reliable results. Second, we used a different learning rate for each weight, by dividing the global learning rate by the number of true groundings of the corresponding clause. This greatly speeds up learning. Because the number of true groundings can vary widely, a learning rate that is small enough to avoid divergence in some weights is too small for fast convergence in others. Having separate learning rates overcomes this problem. We also modified Alchemy to save time and memory in grounding the network.

An MLN for Joint Citation Matching

Citation matching is the problem of extracting bibliographic records from citation lists in technical papers, and merging records that represent the same publication. It has been a major focus of information extraction research (e.g., Pasula *et al.* (2003), Wellner *et al.* (2004)). We use citation matching as a testbed for joint inference with Markov logic. In particular, we focus on extracting titles, authors and venues from citation strings. In this section we present an MLN for this task.[1] In the next section, we learn weights and perform inference with it. While the MLN is specific to citation matching, we believe that the key ideas in it are also applicable to other information extraction tasks (e.g., extraction from Web pages, or from free text).

The main evidence predicate in the MLN is `Token(t, i, c)`, which is true iff token `t` appears in the `i`th position of the `c`th citation. A token can be a word, date, number, etc. Punctuation marks are not treated as separate tokens; rather, the predicate `HasPunc(c, i)` is true iff a punctuation mark appears immediately after the `i`th position in

[1]Available at http://alchemy.cs.washington.edu/papers/poon07.

the cth citation. The query predicates are $\mathtt{InField(i,f,c)}$ and $\mathtt{SameCitation(c,c')}$. $\mathtt{InField(i,f,c)}$ is true iff the $\mathtt{i}$th position of the $\mathtt{c}$th citation is part of field $\mathtt{f}$, where $\mathtt{f} \in \{\mathtt{Title}, \mathtt{Author}, \mathtt{Venue}\}$, and inferring it performs segmentation. $\mathtt{SameCitation(c,c')}$ is true iff citations $\mathtt{c}$ and $\mathtt{c'}$ represent the same publication, and inferring it performs entity resolution.

Isolated Segmentation

We begin by describing our standalone segmentation model. This will form part of the overall MLN for joint inference, and also serve as a baseline for comparison. Our segmentation model is essentially a hidden Markov model (HMM) with enhanced ability to detect field boundaries. This combines two key elements of the state of the art. The observation matrix of the HMM correlates tokens with fields, and is represented by the simple rule

$$\mathtt{Token(+t,i,c)} \Rightarrow \mathtt{InField(i,+f,c)}$$

where all free variables are implicitly universally quantified. The "+t, +f" notation signifies that the MLN contains an instance of this rule for each *(token, field)* pair. If this rule was learned in isolation, the weight of the (t,f)th instance would be $\log(p_{tf}/(1-p_{tf}))$, where p_{tf} is the corresponding entry in the HMM observation matrix.

In general, the transition matrix of the HMM is represented by a rule of the form

$$\mathtt{InField(i,+f,c)} \Rightarrow \mathtt{InField(i+1,+f',c)}$$

However, we (and others, e.g., Grenager *et al.* (2005)) have found that for segmentation it suffices to capture the basic regularity that consecutive positions tend to be part of the same field. Thus we replace $\mathtt{f'}$ by $\mathtt{f}$ in the formula above. We also impose the condition that a position in a citation string can be part of at most one field; it may be part of none.

The main shortcoming of this model is that it has difficulty pinpointing field boundaries. Detecting these is key for information extraction, and a number of approaches use rules designed specifically for this purpose (e.g., Kushmerick (2000)). In citation matching, boundaries are usually marked by punctuation symbols. This can be incorporated into the MLN by modifying the rule above to

$$\mathtt{InField(i,+f,c)} \wedge \neg\mathtt{HasPunc(c,i)} \\ \Rightarrow \mathtt{InField(i+1,+f,c)}$$

The $\neg\mathtt{HasPunc(c,i)}$ precondition prevents propagation of fields across punctuation marks. Because propagation can occur differentially to the left and right, the MLN also contains the reverse form of the rule. In addition, to account for commas being weaker separators than other punctuation, the MLN includes versions of these rules with $\mathtt{HasComma()}$ instead of $\mathtt{HasPunc()}$.

Finally, the MLN contains the following rules: the first two positions of a citation are usually in the author field, and the middle one in the title; initials (e.g., "J.") tend to appear in either the author or the venue field; positions preceding the last non-venue initial are usually not part of the title or venue; and positions after the first venue keyword (e.g., "Proceedings", "Journal") are usually not part of the author or title. Despite its simplicity, this model is quite accurate, and forms a very competitive baseline for joint inference, as we will see in the experimental section.

Entity Resolution

As our starting point for entity resolution, we took the MLN of Singla & Domingos (2006), which assumes pre-segmented citations. It contains rules of the form: if two fields contain many common tokens, they are the same; if the fields of two citations match, the citations also match, and vice-versa; etc. Simply taking the output $\mathtt{InField()}$ predicates of the segmentation MLN as evidence to this MLN would constitute a standard pipeline model. Merging the two MLNs produces a joint model for segmentation and entity resolution. We found, however, that this gives poor results, because entity resolution often leads segmentation astray. Since only a small fraction of citation pairs $(\mathtt{c},\mathtt{c'})$ match, in the absence of strong evidence to the contrary the MLN will conclude that $\mathtt{SameCitation(c,c')}$ is false. If $\mathtt{SameCitation(c,c')}$ is the consequent of a rule (or rule chain) with $\mathtt{InField()}$ in the antecedent, the MLN will (or may) then infer that $\mathtt{InField()}$ is false, even if segmentation alone would correctly predict it to be true. This is an example of how joint inference can hurt accuracy.

Our solution to this problem is to define predicates and rules specifically for passing information between the stages, as opposed to just using the existing $\mathtt{InField()}$ outputs. We want a "higher bandwidth" of communication between segmentation and entity resolution, without letting excessive segmentation noise through. We accomplish this by defining a $\mathtt{SimilarTitle(c,i,j,c',i',j')}$ predicate, which is true if citations $\mathtt{c}$ and $\mathtt{c'}$ contain similar title-like strings at positions $\mathtt{i}$ to $\mathtt{j}$ and $\mathtt{i'}$ to $\mathtt{j'}$, respectively. A string is title-like if it does not contain punctuation and does not match the "title exclusion" rules in the previous section. Two such strings are considered similar if they start with the same trigram and end with the same token. Most importantly, $\mathtt{SimilarTitle()}$ is true only if at least one of the strings is immediately preceded by punctuation, and at least one is immediately followed by punctuation. This greatly reduces the number of potential matches, focusing them on the cases that are most likely relevant for joint inference. In sum, $\mathtt{SimilarTitle()}$ incorporates lower-level segmentation information into entity resolution.

If two citations have similar titles, they are usually the same, unless they appear in different venues (in which case they are usually different versions of the same paper). Hence the basic rule we use to perform entity resolution is

$$\mathtt{SimilarTitle(c,i,j,c',i',j')} \wedge \mathtt{SimilarVenue(c,c')} \\ \Rightarrow \mathtt{SameCitation(c,c')},$$

coupled with the unit clause $\neg\mathtt{SameCitation(c,c')}$, which represents the default. $\mathtt{SimilarVenue(c,c')}$ is true as long as $\mathtt{c}$ and $\mathtt{c'}$ do not contain conflicting venue keywords (e.g., "Journal" in one and "Proceedings" in the other). The rule above ignores whether the citations' authors are the same, which may seem unintuitive. However, the reason is simple. When two citations have the same title and venue, they almost always have the same author as well, and thus comparing authors contributes no additional information. On the other hand, if the authors are the same but are difficult to match, as is often the case, including $\texttt{SimilarAuthor}(\texttt{c}, \texttt{c}')$ as a precondition in the rule above prevents drawing the correct conclusion that the citations are the same. Thus, on balance, including $\texttt{SimilarAuthor}(\texttt{c}, \texttt{c}')$ is detrimental.

As we will see in the experimental section, this simple MLN suffices to outperform the state of the art in citation matching.

Joint Segmentation

Segmenting a citation can help segment similar ones. For example, if in one citation the title is clearly delimited by punctuation, but in a similar one it is not, noticing this similarity can help extract the more difficult title. Incorporating this idea into the MLN described earlier leads to *joint segmentation*, where citations are segmented collectively, as opposed to in isolation. As before, we proceed by defining a predicate for this purpose. $\texttt{JointInferenceCandidate}(\texttt{c}, \texttt{i}, \texttt{c}')$ is true if the trigram starting at position $\texttt{i}$ in citation $\texttt{c}$ also appears somewhere in citation $\texttt{c}'$, the trigrams do not match the "title exclusion" rules, and the trigram in $\texttt{c}$ is not preceded by punctuation, while in $\texttt{c}'$ it is. This rule thus identifies potential opportunities for one title segmentation to help another. We then incorporate this into the segmentation model simply by adding a precondition to the "field propagation" rules. For example:

$$\begin{aligned}&\texttt{InField}(\texttt{i}, +\texttt{f}, \texttt{c}) \wedge \neg\texttt{HasPunc}(\texttt{c}, \texttt{i})\\ &\wedge(\neg\exists \texttt{c}'\ \texttt{JointInferenceCandidate}(\texttt{c}, \texttt{i}, \texttt{c}'))\\ &\Rightarrow \texttt{InField}(\texttt{i}+1, +\texttt{f}, \texttt{c})\end{aligned}$$

The effect of this precondition is to potentially introduce a field boundary in $\texttt{c}$ immediately before a substring if a similar title-like substring is preceded by punctuation in another citation. This may be quite effective when citation lists are "sparse" (i.e., strings usually do not occur both at boundaries and inside fields). However, if the citation lists are "dense," it may produce incorrect field boundaries. Consider the following two citations from the Cora dataset:

> *R. Schapire. On the strength of weak learnability. Proceedings of the 30th I.E.E.E. Symposium on the Foundations of Computer Science, 1989, pp. 28-33.*

> *Robert E. Schapire. 5(2) The strength of weak learnability. Machine Learning, 1990 197-227,*

In the second citation, "The strength of" is immediately preceded by punctuation, which will cause a title boundary to be incorrectly introduced between "On" and "the" in the first citation. To combat this, we can in addition require that the two citations in fact resolve to the same entity:

$$\begin{aligned}&\texttt{InField}(\texttt{i}, +\texttt{f}, \texttt{c}) \wedge \neg\texttt{HasPunc}(\texttt{c}, \texttt{i})\\ &\wedge(\neg\exists \texttt{c}'\ \texttt{JointInferenceCandidate}(\texttt{c}, \texttt{i}, \texttt{c}')\\ &\wedge\texttt{SameCitation}(\texttt{c}, \texttt{c}')) \Rightarrow \texttt{InField}(\texttt{i}+1, +\texttt{f}, \texttt{c})\end{aligned}$$

Table 1: CiteSeer entity resolution: cluster recall on each section.

Approach	Constr.	Face	Reason.	Reinfor.
Fellegi-Sunter	84.3	81.4	71.3	50.6
Lawrence (1999)	89	94	86	79
Pasula (2003)	93	97	96	94
Wellner (2004)	95.1	96.9	93.7	94.7
Joint MLN	96.0	97.1	95.1	96.7

Table 2: Cora entity resolution: pairwise recall/precision and cluster recall.

Approach	Pairwise Rec./Prec.	Cluster Recall
Fellegi-Sunter	78.0 / 97.7	62.7
Joint MLN	94.3 / 97.0	78.1

This is another instance of joint inference between segmentation and entity resolution: in this case, entity resolution helps perform segmentation. However, this rule may sometimes be too strict; consider the following citations from CiteSeer:

> *H. Maruyama Constraint dependency grammar and its weak generative capacity. Computer Software, 1990.*

> *Maruyama, H 1990. Constraint dependency grammar. Technical Report # RT 0044, IBM, Tokyo, Japan.*

In this case, the first rule would correctly place a title boundary before "Constraint", while the second one would not. Both approaches are generally applicable and, as shown in the next section, significantly outperform isolated segmentation. The first one (which we will call Jnt-Seg) is better for sparse datasets (like CiteSeer), and the second (Jnt-Seg-ER) for dense ones (like Cora).

Experiments

Datasets

We experimented on CiteSeer and Cora, which are standard datasets for citation matching. The CiteSeer dataset was created by Lawrence *et al.* (1999) and used in Pasula *et al.* (2003) and Wellner *et al.* (2004). The Cora dataset was created by Andrew McCallum and later segmented by Bilenko & Mooney (2003). We fixed a number of errors in the labels.[2]

In citation matching, a *cluster* is a set of citations that refer to the same paper, and a *nontrivial* cluster contains more than one citation. The CiteSeer dataset has 1563 citations and 906 clusters. It consists of four sections, each on a different topic. Over two-thirds of the clusters are singletons, and the largest contains 21 citations. The Cora dataset has 1295 citations and 134 clusters. Unlike in CiteSeer, almost

[2]Dataset available at http://alchemy.cs.washington.edu/papers/-poon07.

Table 3: CiteSeer segmentation: F1 on each section.

All	Total	Author	Title	Venue
Isolated	94.4	94.6	92.8	95.3
Jnt-Seg	94.5	94.9	93.0	95.3
Jnt-Seg-ER	94.5	94.8	93.0	95.3
Nontrivial	Total	Author	Title	Venue
Isolated	94.4	94.5	92.5	95.5
Jnt-Seg	94.9	95.2	93.6	95.7
Jnt-Seg-ER	94.8	95.1	93.3	95.6
Potential	Total	Author	Title	Venue
Isolated	91.7	91.5	86.5	94.9
Jnt-Seg	94.3	94.5	92.4	95.3
Jnt-Seg-ER	93.9	94.4	92.0	94.9

Table 4: Cora segmentation: F1 on each section.

All	Total	Author	Title	Venue
Isolated	98.2	99.3	97.3	98.2
Jnt-Seg	98.4	99.5	97.5	98.3
Jnt-Seg-ER	98.4	99.5	97.6	98.3
Nontrivial	Total	Author	Title	Venue
Isolated	98.3	99.4	97.4	98.2
Jnt-Seg	98.5	99.5	97.7	98.3
Jnt-Seg-ER	98.5	99.5	97.7	98.3
Potential	Total	Author	Title	Venue
Isolated	97.6	97.9	95.2	98.8
Jnt-Seg	98.7	99.3	97.9	99.0
Jnt-Seg-ER	98.8	99.3	97.9	99.0

every citation in Cora belongs to a nontrivial cluster; the largest cluster contains 54 citations.

Methodology

For CiteSeer, we followed Wellner *et al.* (2004) and used the four sections for four-fold cross-validation. In Cora, we randomly split citations into three subsets so that they were distributed as evenly as possible and no cluster was broken apart (to avoid train-test contamination). We then used these subsets for three-fold cross-validation. In CiteSeer, the largest section yields 0.3 million query atoms and 0.4 million ground clauses; in Cora, 0.26 million and 0.38 million, respectively. For simplicity, we took the original author and title fields and combined other fields such as booktitle, journal, pages, and publishers into venue.

In preprocessing, we carried out standard normalizations like conversion to lower case, simple stemming, and hyphen removal. For token matching, we used edit distance and concatenation. Two tokens match if their edit distance is at most one, and a bigram matches the unigram formed by concatenating the two tokens.

In MLN weight learning, we used 30 iterations of gradient descent, and in each iteration, for each fold, we ran MC-SAT for 20 seconds. This proves to be effective and enables fast learning; see Hinton (2002) for a theoretical justification of a similar method. However, it also introduces fluctuations

Table 5: Segmentation: wins and losses in F1.

CiteSeer	JS-Isolated	JSR-Isolated	JSR-JS
All	8-2	4-2	1-7
Nontrivial	8-2	9-2	1-8
Potential	12-0	9-3	2-9
Cora	JS-Isolated	JSR-Isolated	JSR-JS
All	5-1	6-1	3-1
Nontrivial	4-1	5-1	2-1
Potential	7-1	8-1	5-0

among the runs, and we obviate these by reporting averages of ten runs. The total learning time per run ranged from 20 minutes to an hour. In inference, we used no burn-in and let MC-SAT run for 30 minutes. MC-SAT returns marginal probabilities of query atoms. We predict an atom is true if its probability is at least 0.5; otherwise we predict false.

For entity resolution in CiteSeer, we measured *cluster recall* for comparison with previously published results. Cluster recall is the fraction of clusters that are correctly output by the system after taking transitive closure from pairwise decisions. For entity resolution in Cora, we measured both cluster recall and pairwise recall/precision. In both datasets we also compared with a "standard" Fellegi-Sunter model (see Singla & Domingos (2006)), learned using logistic regression, and with oracle segmentation as the input. For segmentation, we measured F1 for `InField`, F1 being the harmonic mean of recall and precision.

Results

Table 1 shows that our approach outperforms previous ones on CiteSeer entity resolution. (Results for Lawrence (1999), Pasula (2003) and Wellner (2004) are taken from the corresponding papers. The MLN results are for Jnt-Seg; the results for Jnt-Seg-ER are identical, except for 95.5 instead of 96.0 on Constr.) This is particularly notable given that the models of Pasula *et al.* (2003) and Wellner *et al.* (2004) involved considerably more knowledge engineering than ours, contained more learnable parameters, and used additional training data.

Table 2 shows that our entity resolution approach easily outperforms Fellegi-Sunter on Cora, and has very high pairwise recall/precision. (As before, the MLN results are for Jnt-Seg; the results for Jnt-Seg-ER are identical, except for 75.2 instead of 78.1 on cluster recall. Note that in Cora clusters are much more difficult to discriminate than in CiteSeer; also, because Cora contains on average only 44 clusters per fold, each less-than-perfect cluster degrades cluster recall by over 2%.)

Table 3 and Table 4 compare isolated segmentation with the two joint inference methods on CiteSeer and Cora. The "Total" column aggregates results on all `InField` atoms, and the remaining columns show results for each field type. The "All" section shows results for all citations, "Nontrivial" for non-singleton citations, and "Potential" for citations with poor author-title boundary (and which therefore might benefit from joint inference). Approximately 9% of citations fall

Table 6: Percentage of F1 error reduction in segmentation obtained by joint inference.

CiteSeer	Total	Author	Title	Venue
All	2	6	3	0
Nontrivial	9	13	15	4
Potential	31	35	44	8
Cora	Total	Author	Title	Venue
All	11	24	7	6
Nontrivial	12	17	12	6
Potential	50	67	56	17

into the "Potential" category in CiteSeer, and 21% in Cora. All total improvements are statistically significant at the 1% level using McNemar's test. Table 5 summarizes the relative performance of the three methods in terms of number of wins and losses. Each *(field, subset)* pair is one trial. In CiteSeer, there are four subsets and twelve comparisons, and thus 12-0 means that the first method outperformed the second in all comparisons. In Cora, there are three subsets and nine comparisons in total, so 5-0 means the first wins five times and ties four times.

Joint inference consistently outperforms isolated inference. Table 6 shows error reduction from isolated inference by the better-performing joint inference method (Jnt-Seg in CiteSeer, Jnt-Seg-ER in Cora). Improvement among potential citations is substantial: in Cora, error is reduced by half in total, 67% for authors, and 56% for titles; in CiteSeer, error is reduced by 31% in total, 35% for authors, and 44% for titles. Overall in Cora, joint inference reduces error by 11% in total and 24% for authors. The overall improvement in CiteSeer is not as large, because the percentage of potential citations is much smaller. The joint inference methods do not impact performance on citations with good boundaries (not shown separately in these tables).

In CiteSeer, Jnt-Seg outperforms Jnt-Seg-ER, while in Cora the opposite is true. These results are consistent with the different characteristics of the two datasets: Cora is dense, while CiteSeer is sparse. In Cora, there is a lot of word overlap among citations, and often the same trigram occurs both within a field and at the boundary, making Jnt-Seg more prone to misaligning boundaries. In CiteSeer, however, this rarely happens and Jnt-Seg-ER, being conservative, can miss opportunities to improve segmentation. There is no absolute advantage of one joint inference method over the other; the best choice depends on the dataset.

Conclusion

Joint inference is currently an area of great interest in information extraction, natural language processing, statistical relational learning, and other fields. Despite its promise, joint inference is often difficult to perform effectively, due to its complexity, computational cost, and sometimes mixed effect on accuracy. In this paper, we show how these problems can be addressed in a citation matching domain, using Markov logic, the MC-SAT algorithm, and careful interconnection of inference tasks. Experiments on standard datasets show that our approach is efficient and consistently improves accuracy over non-joint inference. It is also more accurate and easier to set up than previous semi-joint approaches.

Directions for future work include identifying more joint inference opportunities in citation matching, applying joint inference to other problems, further improving the scalability of the algorithms, and developing general principles for solving joint inference tasks.

Acknowledgements

We are grateful to Ben Wellner for providing us with the latest CiteSeer dataset. This research was partly supported by DARPA grant FA8750-05-2-0283 (managed by AFRL), DARPA contract NBCH-D030010, NSF grant IIS-0534881, and ONR grant N00014-05-1-0313. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, NSF, ONR, or the United States Government.

References

Bilenko, M. & Mooney, R. 2003. Adaptive duplicate detection using learnable string similarity measures. In *Proc. KDD-03.*

Bunescu, R. & Mooney, R. 2004. Collective information extraction with relational Markov networks. In *Proc. ACL-04.*

Getoor, L. & Taskar, B., eds. 2007. *Introduction to Statistical Relational Learning.* MIT Press.

Grenager, T.; Klein, D.; Manning, D. 2005. Unsupervised learning of field segmentation models for information extraction. In *Proc. ACL-05.*

Hinton, G. 2002. Training products of experts by minimizing contrastive divergence. *Neural Computation* 14:1771–1800.

Kok, S.; Singla, P.; Richardson, M.; Domingos, P.; Sumner, M.; Poon, H. 2006. The Alchemy system for statistical relational AI. http://alchemy.cs.washington.edu/.

Kushmerick, N. 2000. Wrapper induction: efficiency and expressiveness. *Artificial Intelligence* 118:15–68.

Lawrence, S.; Bollacker, K.; Giles, C. L. 1999. Autonomous citation matching. In *Proc. International Conference on Autonomous Agents.*

Pasula, H.; Marthi, B.; Milch, B.; Russell, S.; Shpitser, I. 2003. Identity uncertainty and citation matching. In *Proc. NIPS-03.*

Poon, H. & Domingos, P. 2006. Sound and efficient inference with probabilistic and deterministic dependencies. In *Proc. AAAI-06.*

Punyakanok, V.; Roth, D.; Yih, W. 2005. The necessity of syntactic parsing for semantic role labeling. In *Proc. IJCAI-05.*

Richardson, M. & Domingos, P. 2006. Markov logic networks. *Machine Learning* 62:107–136.

Singla, P. & Domingos, P. 2005. Discriminative training of Markov logic networks. In *Proc. AAAI-05.*

Singla, P. & Domingos, P. 2006. Entity resolution with Markov logic. In *Proc. ICDM-06.*

Sutton, C.; McCallum, A.; Bilmes, J., eds. 2006. *Proc. HLT/NAACL-06 Workshop on Joint Inference for Natural Language Processing.*

Wellner, B.; McCallum, A.; Peng, F.; Hay, M. 2004. An integrated, conditional model of information extraction and coreference with application to citation matching. In *Proc. UAI-04.*