Multi-document Relationship Fusion via Constraints on Probabilistic Databases

Gideon Mann
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
gideon.mann@gmail.com

Abstract

Previous multi-document relationship extraction and fusion research has focused on single relationships. Shifting the focus to multiple relationships allows for the use of mutual constraints to aid extraction. This paper presents a fusion method which uses a probabilistic database model to pick relationships which violate few constraints. This model allows improved performance on constructing corporate succession timelines from multiple documents with respect to a multi-document fusion baseline.

1 Introduction

Single document information extraction of named entities and relationships has received much attention since the MUC evaluations[1] in the mid-90s (Appelt et al., 1993; Grishman and Sundheim, 1996). Recently, there has been increased interest in the extraction of named entities and relationships from multiple documents, since the redundancy of information across documents has been shown to be a powerful resource for obtaining high quality information even when the extractors have access to little or no training data (Etzioni et al., 2004; Hasegawa et al., 2004). Much of the recent work in multi-document relationship extraction has focused on the extraction of isolated relationships (Agichtein, 2005; Pasca et al., 2006), but often the goal, as in single document tasks like MUC, is to extract a template or a relational database composed of related facts.

With databases containing multiple relationships, the semantics of the database impose constraints on possible database configurations. This paper presents a statistical method which picks relationships which violate few constraints as measured by a probabilistic database model. The constraints are hard constraints, and robust estimates are achieved by accounting for the underlying extraction/fusion uncertainty.

This method is applied to the problem of constructing management succession timelines which have a rich set of semantic constraints. Using constraints on probabilistic databases yields F-Measure improvements of 5 to 18 points on a per-relationship basis over a state-of-the-art multi-document extraction/fusion baseline. The constraints proposed in this paper are used in a context of minimally supervised information extractors and present an alternative to costly manual annotation.

2 Semantic Constraints on Databases

This paper considers management succession databases where each record has three fields: a CEO's name and the start and end years for that person's tenure as CEO (Table 1 Column 1). Each record is represented by three binary logical predicates: *ceo*(c,x), *start*(x,y^1), *end*(x,y^2), where c is a company, x is a CEO's name, and y^1 and y^2 are years.[2]

[1] http://www.itl.nist.gov/iaui/894.02/related_projects/muc/

[2] All of the relationships in this paper are defined to be binary relationships. When extracting relationships of higher ar-

Proceedings of NAACL HLT 2007, pages 332–339,
Rochester, NY, April 2007.

Explicit Relationships	Implicit Relationships	Logical Constraints (a partial list)
ceo(c, x)	*precedes*(x^1,x^2)	*ceo*(c,x^1), *precedes*(x^1,x^2) $\Leftrightarrow$ *ceo*(c,x^2), *end*(x^1,y), *start*(x^2, y)
start(x, y)	*inoffice*(x, y)	*start*(x,y^1), *inoffice*(x,y^2), *end*(x, y^3) $\Rightarrow y^1 \leq y^2 \leq y^3$
end(x, y)	*predates*(x^1, x^2)	*inoffice*(x^1, y^1), *inoffice*(x^2, y^2) , $y^1 < y^2 \Rightarrow$ *predates*(x^1,x^2)
		precedes(x^1,x^2) , *inoffice*(x^1,y^1), *inoffice*(x^2,y^2) $\Rightarrow y^1 \leq y^2$

Table 1: Database semantics can provide 1) a method to augment the explicit relationships in the database with implicit relationships and 2) logical constraints on these explicit and implicit relationships. In the above table, c is a company, x^i is a person, and y^i is a time.

In this setting, database semantics allow for the derivation of other implicit relationships from the database: the immediate predecessor of a given CEO (*precedes*(x^1,x^2)), all predecessors of a given CEO (*predates*(x^1,x^2)) and all years the CEO was in office (*inoffice*(x,y^3)), where x^2 is a CEO's name and t^3 a year (Table 1 Column 2).

These implicit relationships and the original explicit relationships are governed by a series of semantic relations which impose constraints on the permissible database configurations (Table 1 Column 3). For example, it will always be true that a CEO's start date precedes their end date:

$$\forall x : start(x, y^1), end(x, y^2) \Rightarrow y^1 \leq y^2 \,.$$

Multi-document extraction of single relationships exploits redundancy and variety of expression to extract accurate information from across many documents. However, these are not the only benefits of extraction from a large document collection. As well as being rich in redundant information, large document collections also contain a wealth of secondary relationships which are related to the relationship of interest via constraints as described above. These secondary relationships can yield benefits to augment those achieved by redundancy.

3 Multi-Document Database Fusion

There are typically two steps in the extraction of single relationships from multiple documents. In the first step, a relationship extractor goes through the corpus, finds all possible relationships r in all sentences s and gives them a score $p(r|s)$. Next, the relationships are fused across sentences to generate one score ϕ_r for each relationship.

This paper proposes a third step which combines the fusion scores across relationships. This section first presents a probabilistic database model generated from fusion scores and then shows how to use this model for multi-document fusion.

3.1 A Probabilistic Database Model

A relationship r is defined to be a 3-tuple $r^{t,a,b} = r(t, a, b)$, where t is the type of the relationship (e.g. *start*), and a and b are the arguments of the binary relationship.[3]

To construct a probabilistic database for a given corpus, the weights generated in relationship fusion are normalized to provide the conditional probability of a relationship given its type:

$$p(r_1^{t,a,b}|t_1) = \frac{\phi_{r_1^{t,a,b}}}{\sum_{r_i : r_i^t = t} \phi_{r_i^{t,a,b}}} \,,$$

where ϕ_r is the fusion score generated by the extraction/fusion system.[4] By applying a prior over types $p(t)$, a distribution $p(r_1, t_1)$ can be derived. Given strong independence assumptions, the probability of an ordered database configuration $R = r_{1..n}$ of types $t_{1..n}$ is:

$$p(r_{1..n}, t_{1..n}) = \prod_{i=1}^{n} p(r_i, t_i) \,. \qquad (1)$$

ity, typically binary relationships are combined (McDonald et al., 2005).

[3] For readibility in future examples, "a" and "b" are replaced by the types of their arguments. For example, for *start* the year in which the CEO starts is referred to as r^{year}.

[4] The following fusion method does not depend on a particular extraction/fusion architecture or training methodology, merely this conditional probability distribution.

As proposed, the model in Equation 1 is faulty since the relationships in a database are not independent. Given a set of database constraints, certain database configurations are **illegal** and should be assigned zero probability. To address this, the model in Equation 1 is augmented with constraints that explicitly set the probability of a database configuration to zero when they are violated.

A database constraint is a logical formula $\eta(r_{1..\pi(\eta)})$, where $\pi(\eta)$ is the arity of the constraint η. For the constraints presented in this paper, all constraints η are modeled with two terms η^α and η^β where:

$$\eta(r_{1..\pi(\eta)}) = \left(\eta^\alpha(r_{1..\pi(\eta)}) \Rightarrow \eta^\beta(r_{1..\pi(\eta)})\right) .$$

For a set of relationships, a constraint **holds** if $\eta(\cdot)$ is true, and the constraint **applies** if $\eta^\alpha(\cdot)$ is true. A constraint $\eta(\cdot)$ can only be **violated** (false) when the constraint applies, since: $(\text{false} \Rightarrow X) = \text{true}$.

In application to a database, each constraint η is quantified over the database to become a quantified constraint $\eta_{r_{1..n}}$. For example, the constraints that a person's start date must come before their end date is universally quantified over all pairs of relationships in a configuration $R = r_{1..n}$:

$$\begin{aligned}\eta_{r_{1..n}} = \forall r_1, r_2 \in R :\ \eta(r_1, r_2) = \\ (r_1^t = start, r_2^t = end, r_1^{ceo} = r_2^{ceo}) \\ \Rightarrow (r_1^{year} < r_2^{year}) .\end{aligned}$$

This constraint applies to *start* and *end* relationships whose CEO argument matches and is violated when the years are not in order. If the quantified constraint $\eta_{r_{1..n}}$ is true for a given database configuration $r_{1..n}$ then it **holds**.

To ensure that only legal database configurations are assigned positive probabilities, Equation 1 is augmented with a factor

$$\phi^\eta_{r_{1..n}} = \begin{cases} 1 & \text{if } \eta_{r_{1..n}} \text{holds} \\ 0 & \text{otherwise} . \end{cases}$$

To include a constraint η, the database model in Equation 1 is extended to be:

$$p_\eta(r_{1..n}, t_{1..n}) = \frac{1}{Z}\left(\prod_i p(r_i, t_i)\right)\phi^\eta_{r_{1..n}} ,$$

where Z is the partition function and corresponds to the total probability of all database configurations. A set of constraints $\eta^{1..Q} = \eta^1..\eta^Q$ can be integrated similarly:

$$p_{\eta^{1..Q}}(r_{1..n}, t_{1..n}) = \frac{1}{Z}\left(\prod_i p(r_i, t_i)\right)\prod_q \phi^{\eta^q}_{r_{1..n}} \tag{2}$$

With these added constraints, the probabilistic database model assigns non-zero probability only to databases which don't violate any constraints.

3.2 Constraints on Probabilistic Databases for Relationship Rescoring

Though the constrained probabilistic database model in Equation 2 is theoretically appealing, it would be infeasible to calculate its partition function which requires enumeration of all legal 2^n databases. This section proposes two methods for re-scoring relationships with regards to how likely they are to be present in a legal database configuration using the model proposed above. The first method is a confidence estimate based on how likely it is that η holds for a given relationship r_1:

$$\begin{aligned}\Lambda_\eta(r_1, t_1) &= E_{p(r_{2..n}, t_{2..n})}\left[\phi^\eta_{r_{1..\pi(\eta)}}\right] \\ &= \frac{\sum_{r_{2..\pi(\eta)}}\left(\prod_{i=2}^{\pi(\eta)} p(r_i, t_i)\right)\phi^\eta_{r_{1..\pi(\eta)}}}{\sum_{r_{2..\pi(\eta)}}\left(\prod_{i=2}^{\pi(\eta)} p(r_i, t_i)\right)} \\ &= \frac{\sum_{r_{2..\pi(\eta)}} p_\eta(r_{1..n}, t_{1..n})}{\sum_{r_{2..\pi(\eta)}} p(r_{1..n}, t_{1..n})} ,\end{aligned}$$

where the expectation that the constraint holds is equivalent to the likelihood ratio between the database probability models with and without constraints. In effect, this model measures the expectation that the constraint holds for a finite database "look-ahead" of size $\pi(\eta) - 1$.

With this method, for a constraint to reduce the confidence in a particular relationship by half, half of all configurations would have to violate the constraint.[5] Since inconsistencies are relatively rare, for a given relationship $\Lambda_\eta(r, t) \approx 1$ (i.e. almost all small databases are legal).

[5] Assuming equal probability for all relationships.

To remedy this, another factor ϕ^{α} is defined similarly to ϕ^{η}, except that it takes a value of 1 only if the constraint applies to that database configuration. An applicability probability model is then defined as:

$$p_{\alpha}(r_{1..n}, t_{1..n}) = \frac{1}{Z}\left(\prod_i p(r_i, t_i)\right)\phi^{\alpha}_{r_{1..n}}.$$

The second confidence estimate is based on how likely it is that the constraint is holds in cases where it applies (i.e. is not violated):

$$\begin{aligned}
&\Lambda_{\eta,\alpha}(r_1, t_1) \\
&\quad = E_{p_{\alpha}(r_{2..n}, t_{2..n})}\left[\phi^{\eta}_{r_{1..\pi(\eta)}}\right] \\
&\quad = \frac{\sum_{r_{2..\pi(\eta)}}\left(\prod_{i=2}^{\pi(\eta)} p(r_i, t_i)\right)\phi^{\alpha}_{r_{1..\pi(\eta)}}\phi^{\eta}_{r_{1..\pi(\eta)}}}{\sum_{r_{2..\pi(\eta)}}\left(\prod_{i=2}^{\pi(\eta)} p(r_i, t_i)\right)\phi^{\alpha}_{r_{1..\pi(\eta)}}}.
\end{aligned}$$

When the constraint doesn't apply it cannot be violated, so this confidence estimate ignores those configurations that can't be affected by the constraint.

Recall that $\Lambda_{\eta}(r, t)$ is the likelihood ratio between the probability of configurations in which r holds for constraint η and all configurations. In contrast, $\Lambda_{\eta,\alpha}(r, t)$ is the likelihood ratio between the database configurations where r applies and holds for η and the database configurations where η applies. In the later ratio, for confidence in a particular relationship to be cut in half, only half of the configurations which might actually contain an inconsistency would be required to produce a violation.[6] As a result, $\Lambda_{\eta,\alpha}(r, t)$ gives a much higher penalty to relationships which create inconsistencies than does $\Lambda_{\eta}(r, t)$.

In order to apply multiple constraints, independent database look-aheads are generated for each constraint q:

$$\Lambda_{\eta^{1..Q},\alpha^{1..Q}}(r_1, t_1) = \prod_q \Lambda_{\eta^q,\alpha^q}(r_1, t_1).$$

For a particular relationship type, these confidence scores are calculated and then used to rank the relationships via:

$$\hat{c}_{\eta^{1..Q},\alpha^{1..Q}}(r_1, t_1) = p(r_1, t_1)\prod_q \Lambda_{\eta^q,\alpha^q}(r_1, t_1) \tag{3}$$

Databases with different precision/recall trade offs can be selected by descending the ranked list.[7]

4 Experiments

In order to test the fusion method proposed above, human annotators manually constructed truth data of complete chief executive histories for 18 Fortune-500 companies using online resources. Extraction from these documents is particularly difficult because these data have vast differences in genre and style and are considerably noisy. Furthermore, the task is complicated to start with.[8]

A corpus was created for each company by issuing a Google query for "CEO-of-Company OR Company-CEO", and collecting the top ranked documents, generating up to 1000 documents per company. The data was then split randomly into training, development and testing sets of 6, 4, and 8 companies.

Training :	Anheuser-Busch, Hewlett-Packard, Lenner, McGraw-Hill, Pfizer, Raytheon
Dev. :	Boeing, Heinz, Staples, Textron
Test :	General Electric, General Motors, Gannett, The Home Depot, IBM, Kroger, Sears, UPS

Ground truth was created from the entire web, but since the corpus for each company is only a small web snapshot, the experimental results are not similar to extraction tasks like MUC and ACE in that the corpus is not guaranteed to contain the information necessary to build the entire database. In particular,

[6] For example, for a *start* relationship and the constraint that a CEO must start before they end, this method would only examine configurations of one *start* and one *end* relationship for the same CEO. The confidence in a particular start date would be halved if half of the proposed end dates for a given CEO occurred before it.

[7] One thing to note is that since all relationships are given confidence estimates separately, this process may result ultimately in a database where constraints are violated. A potential solution, which is not explored here, would be to incrementally add relationships to the database from the ranked list only if their addition doesn't make the database inconsistent.

[8] For example, in certain companies, the title of the chief executive has changed over the years, often going from "President" to "Chief Executive Officer". To make things more complicated, after the change, the role of "President" may still hang on as a subordinate to the CEO!

1) Only one start or end per person. $\forall r_1, r_2 : \eta(r_1, r_2) = (r_1^{type} = r_2^{type} = (start \cup end), r_1^{ceo} = r_2^{ceo}) \Rightarrow (r_1^{year} = r_2^{year})$
2) Only a CEO's start or end dates belong in the database. $\forall r_1 \exists r_2 : \eta(r_1, r_2) = (r_1^{type} = start \cup end, r_2^{type} = ceo) \Rightarrow (r_1^{ceo} = r_2^{ceo})$
3) Start dates come before end dates. $\forall r_1, r_2 : \eta(r_1, r_2) = (r_1^{type} = start, r_2^{type} = end, r_1^{ceo} = r_2^{ceo}) \Rightarrow (r_1^{year} \leq r_2^{year})$
4) Can't be in the middle of someone else's tenure. $\forall r_1, r_2, r_3 : \eta(r_1, r_2, r_3) = (r_1^{type} = start \cup inoffice, r_2^{type} = end \cup inoffice, r_3^{type} = start \cup inoffice \cup end,$ $r_1^{ceo} = r_2^{ceo} \neq r_3^{ceo}, r_1^{year} < r_2^{year}) \Rightarrow (r_3^{year} \leq r_1^{year} \cup r_3^{year} \geq r_2^{year})$
5) CEO's are only in office after their start. $\forall r_1, r_2 : \eta(r_1, r_2) = (r_1^{type} = start, r_2^{type} = inoffice, r_1^{ceo} = r_2^{ceo}) \Rightarrow (r_1^{year} \leq r_2^{year})$
6) CEO's are only in office before their end. $\forall r_1, r_2 : \eta(r_1, r_2) = (r_1^{type} = inoffice, r_2^{type} = end, r_1^{ceo} = r_2^{ceo}) \Rightarrow (r_1^{year} \leq r_2^{year})$
7) Someone's end is the same as their successor's start. $\forall r_1, r_2, r_3 : \eta(r_1, r_2, r_3) =$ $(r_1^{type} = end, r_2^{type} = start, r_3^{type} = precedes, r_1^{ceo} = r_3^{first}, r_2^{ceo} = r_3^{second}) \Rightarrow (r_1^{year} = r_2^{year})$
8) All of the someone's dates (start, inoffice, end) are before their successors. $\forall r_1, r_2, r_3 : \eta(r_1, r_2, r_3) = (r_1^{type} = start \cup end \cup inoffice, r_2^{type} = start \cup inoffice \cup end, r_3^{type} = precedes,$ $r_1^{ceo} = r_3^{first}, r_2^{ceo} = r_3^{second}) \Rightarrow (r_1^{year} \leq r_2^{year})$
9) Only CEO succession in the database. $\forall r_1 \exists r_1, r_2 : \eta(r_1, r_2, r_3) = (r_1^{type} = precedes, r_2^{type} = r_3^{type} = ceo) \Rightarrow (r_1^{first} = r_2^{ceo}, r_1^{second} = r_3^{ceo})$

Table 2: For a CEO succession database like the one presented in Table 1, the above constraints must hold if the database is consistent.

many CEOs from pre-Internet years were either infrequently mentioned or not mentioned at all in the database.[9] In the following experiments, recall is reported for facts that were retrieved by the extraction system.

4.1 Relationship Extraction and Fusion

A two-class maximum-entropy classifier was trained for each relationship type. Each classifier takes a sentence and two marked entities (e.g. a person and a year)[10] and gives the probability that a relationship between the two entities is supported by the sentence. For each relationship type, one of the elements is designated as the "hook" in order to generate likely negative examples.[11] In training, all entity pairs are collected from the corpus. The pairs whose "hook" element doesn't appear in the database are thrown out. The remaining pairs are then marked by exact match to the database. In testing, the relationship extractor yields the probability $p(r|s)$ of an entity pair relationship r in a particular sentence s.

The features used in the classifier are: unigrams between the given information and the target, distance in words between the given information and the target, and the exact string between the given information and the target (if less that 3 words long).

After extraction from individual sentences, the relationships are fused together such that there is one score for each unique entity pair. In the case of person names, normalization was performed to merge coreferent but lexically distinct names (e.g. "Phil Condit" and "Philip M. Condit").

In the following experiments, the baseline fusion score is:

$$\phi_r = \sum_s p(r|s) \tag{4}$$

4.2 Experimental Results

Given the management succession database proposed in Section 2, Table 2 enumerates a set of quantified constraints. Information extraction and fusion were run separately for each company to create a probabilistic database. In this section, various constraint sets are applied, either individually or jointly, and evaluated in two ways. The first measures per-relationship precision/recall using the model pro-

[9] Another consequence is that assessing the effectiveness of the relationships extraction on a per-extraction basis is difficult. Because there are no training sentences where it is known that the sentence contains the relationship of interest, grading per-extraction results can be deceptive.

[10] The person tagger used is the named-entity tagger from OpenNLP tools and the year tagger simply finds any four digit numbers between 1950 and 2010.

[11] For the CEO relationship, the company was taken to be the hook. For the other relationships the hook was the primary CEO.

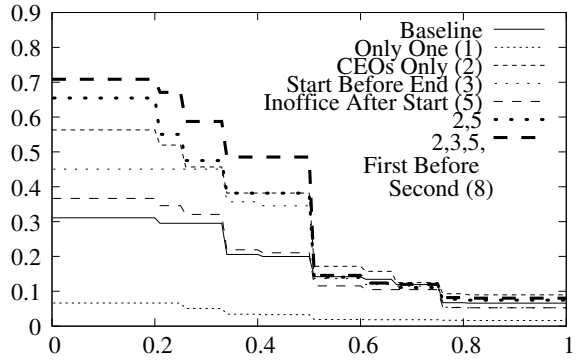

Figure 1: Precision/Recall curve for start(x,t) relationships. The joint constraint "2,3,5,8" is the best performing, even though constraints "3" and "8" (not pictured) alone don't perform well.

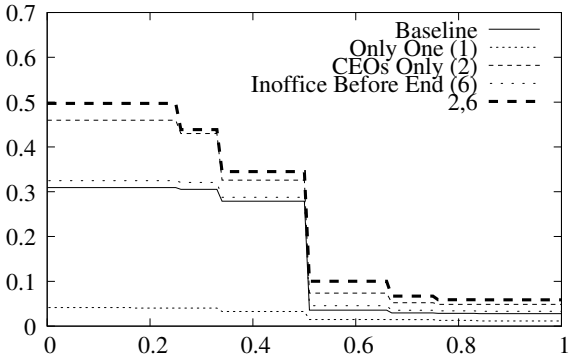

Figure 2: Precision/Recall curve for end(x,t) relationships alone. The joint constraint "2,6" is the best performing.

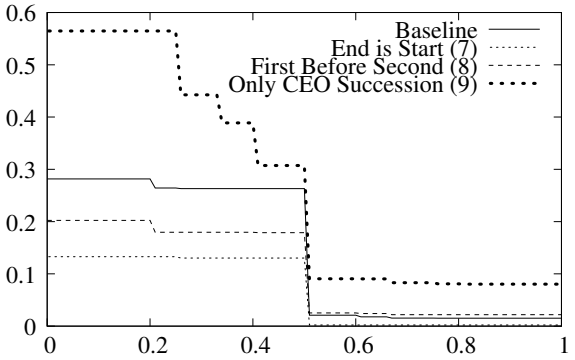

Figure 3: Precision/Recall for precedes(x,y) relationships alone. Though the constraint "First Before Second (8)" helps performance on start(x,t) relationships, the only constraint which aids here is "Only CEOs Succession (9)".

posed and the second looks at the precision/recall of a heterogeneous database with many relationship types. Both evaluations examine the ranked lists of relationships, where the relationships are ranked by rescoring via constraints on probabilistic databases (Equation 3) and compared to the baseline fusion score (Equation 4). The evaluations use two standard metrics, interpolated precision at recall level i (P_{R_i}), and MaxF1:

$$P_{R_i} = \max_{j \geq i} P_{R_j},$$

$$\text{MaxF1} = \max_i \frac{2}{\frac{1}{P_{R_i}} + \frac{1}{R_i}}.$$

Figures 1, 2, and 3 show precision/recall curves for the application of various sets of constraints. Table 3 lists the MaxF1 scores for each of the constraint variants. For *start* and *end*, the majority of constraints are beneficial. For *precedes*, only the constraint that improved performance constraints both people in the relationship to be CEOs. Across all relationships, performance is hurt when using the constraint that there could only be one relationship of each type for a given CEO. The reason behind this is that the confidence estimate based on this constraint favors relationships with few competitors, and those relationships are typically for people who are infrequent in the corpus (and therefore unlikely to be CEOs).

The best-performing constraint sets yield between 5 and 18 points of improvement on Max F1 (Table 3). Surprisingly, the gains from joint constraints are sometimes more than their additive gains. "2,3,5,6,8" is 6 points better for the start relationship than "2,3,5,6", but the gains from "8" alone are negligible.

These performance gains on the individual relationship types also lead to gains when generating an entire database (Figure 4). The highest performing constraint is the "CEOs Only (2)" constraint, which outperforms the joint constraints of the previous section. One reason the joint constraints don't do as well here is that each constraint makes the confidence estimate smaller and smaller. This doesn't have an effect when judging the relationship types individually, but when combining the relationships results, the fused relationships types (*start*, *end*) be-

Constraint Set	Max F1			
	Start	End	Pre.	DB
$\emptyset$ (baseline)	31.2	35.8	34.5	37.9
Only One (1)	10.5	7.2	-	38.1
CEOs Only (2) or (9)	43.3	39.4	**39.4**	**42.9**
Start Before End (3)	40.8	32.8	-	40.9
No Overlaps (4)	31.5	35.9	-	36.8
Inoffice After Start (5)	32.5	-	-	38.2
Inoffice Before End (6)	-	36.5	-	37.4
End is Start (7)	7.3	8.0	20.7	39.2
First before Second (8)	31.4	35.6	26.3	38.1
2,5,6	43.3	40.8	-	42.7
2,3,5,6	43.9	43.3	-	42.2
2,3,5,6,8	**49.3**	**43.9**	26.3	40.9

Table 3: Max F1 scores for three relationships **Start**(x,t), **End**(x,t) and **Pre**cedes(x,y)) in isolation and within the context of whole database **DB**. The joint constraints perform best for the explicit relationships in isolation. Using constraints on implicit derived fields (Inoffice and Precedes) provides additional benefit above constraints strictly on explicit database fields (start, end, ceo).

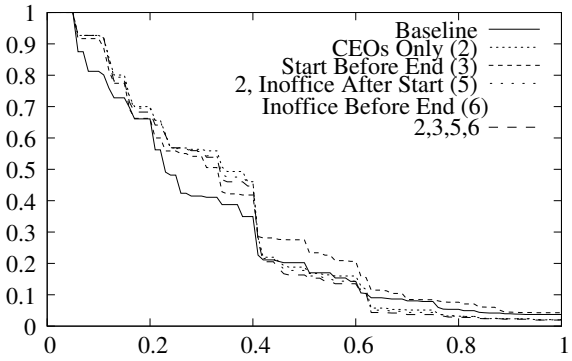

Figure 4: Precision/Recall curve for whole database reconstruction. Performance curves using constraints dominate the baseline.

come artificially lower ranked than the unfused relationship type (*ceo*). The best performing contrained probabilistic database approach beats the baseline by 5 points.

5 Related Work

Techniques for information extraction from minimally supervised data have been explored by Brin (1998), Agichtein and Gravano (2000), and Ravichandran and Hovy (2002). Those techniques propose methods for estimating extractors from example relationships and a corpus which contains instances of those relationships.

Nahm and Mooney (2002) explore techniques for extracting multiple relationships in single document extraction. They learn rules for predicting certain fields given other extracted fields (i.e. a someone who lists Windows as a specialty is likely to know Microsoft Word).

Perhaps the most related work to what is presented here is previous research which uses database information as co-occurrence features for information extraction in a multi-document setting. Mann and Yarowsky (2005) present an incremental approach where co-occurrence with a known relationship is a feature added in training and test. Culotta et al. (2006) introduce a data mining approach where discovered relationships from a database are used as features in extracting new relationships. The database constraints presented in this paper provide a more general framework for jointly conditioning multiple relationships. Additionally, this constraint-based approach can be applied without special training of the extraction/fusion system.

In the context of information fusion of single relationships across multiple documents, Downey et al. (2005) propose a method that models the probabilities of positive and negative extracted classifications. More distantly related, Sutton and McCallum (2004) and Finkel et al. (2005) propose graphical models for combining information about a given entity from multiple mentions.

In the field of question answering, Prager et al. (2004) answer a question about the list of compositions produced by a given subject by looking for related information about the subject's birth and death. Their method treats supporting information as fixed hard constraints on the original questions and are applied in an ad-hoc fashion. This paper proposes a probabilistic method for using constraints in the context of database extraction and applies this method over a larger set of relations.

Richardson and Domingos (2006) propose a method for reasoning about databases and logical constraints using Markov Random Fields. Their

model applies reasoning starting from a known database. In this paper the database is built from extraction/fusion of relationships from web pages and contains a significant amount of noise.

6 Conclusion

This paper has presented a probabilistic method for fusing extracted facts in the context of database extraction when there exist logical constraints between the fields in the database. The method estimates the probability than the inclusion of a given relationship will violate database constraints by taking into account the uncertainty of the other extracted relationships. Along with the relationships explicitly listed in the database, constraints are formed over implicit fields directly recoverable from the explicit listed relationships.

The construction of CEO succession timelines using minimally trained extractors from web text is a particularly challenging problem because of noise resulting from the wide variation in genre in the corpora and errors in extraction. The use of constraints on probabilistic databases is effective in resolving many of these errors, leading to improved precision and recall of retrieved facts, with F-measure gains of 5 to 18 points.

The method presented in this paper combines symbolic and statistical approaches to natural language processing. Logical constraints are made more robust by taking into account the uncertainty of the extracted information. An interesting area of future work is the application of data mining to search for appropriate constraints to integrate into this model.

Acknowledgements

This work was supported in part by DoD contract #HM1582-06-1-2013. Any opinions, findings and conclusions or recommendations expressed in this material belong to the author and do not necessarily reflect those of the sponsor.

References

E. Agichtein and L. Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of ICDL*, pages 85–94.

E. Agichtein. 2005. *Extracting Relations from Large Text Collections*. Ph.D. thesis, Columbia University.

D. Appelt, J. Hobbs, J. Bear, D. Israel, and M. Tyson. 1993. FASTUS: a finite-state processor for information extraction from real-world text. In *Proceedings of IJCAI*.

S. Brin. 1998. Extracting patterns and relations from the world wide web. In *WebDB Workshop at 6th International Conference on Extending Database Technology, EDBT'98*, pages 172–183.

A. Culotta, A. McCallum, and J. Betz. 2006. Integrating probabilistic extraction models and data mining to discover relations and patterns in text. In *HLT-NAACL*, pages 296–303, New York, NY, June.

D. Downey, O. Etzioni, and S. Soderland. 2005. A probabilistic model of redundancy in information extraction. In *IJCAI*.

O. Etzioni, M. Cafarella, D. Downey, S. Kok, A-M. Popescu, T. Shaked, S. Soderland, D. Weld, and A. Yates. 2004. Web-scale information extraction in knowitall. In *WWW*.

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

R. Grishman and B. Sundheim. 1996. Message understanding conference-6: A brief history. In *Proceedings of COLING*.

T. Hasegawa, S. Sekine, and R. Grishman. 2004. Discovering relations amoung named entities from large corpora. In *ACL*.

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

R. McDonald, F. Pereira, S. Kulick, S. Winters, Y. Jin, and P. White. 2005. Simple algorithms for complex relationship extraction with applications to biomedical ie. In *Proceedings of ACL*.

U. Nahm and R. Mooney. 2002. Text mining with information extraction. In *Proceedings of the AAAI 2220 Spring Symposium on Mining Answers from Texts and Knowledge Bases*, pages 60–67.

M. Pasca, D. Lin, J. Bigham, A. Lifchits, and A. Jain. 2006. Organizing and searching the world wide web of facts - step one: The one-million fact extracion challenge. In *AAAI*.

J. Prager, J. Chu-Carroll, and K. Czuba. 2004. Question answering by constraint satisfaction: Qa-by-dossier with constraints. In *Proceedings of ACL*, pages 574–581.

D. Ravichandran and E. Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of ACL*, pages 41–47.

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

C. Sutton and A. McCallum. 2004. Collective segmentation and labeling of distant entities in information extraction. Technical Report TR # 04-49, University of Massachusetts, July. Presented at ICML Workshop on Statistical Relational Learning and Its Connections to Other Fields.