Handling uncertainty in information extraction

Maurice van Keulen[1] and Mena B. Habib[1]

University of Twente, Faculty of EEMCS, Enschede, The Netherlands
{m.vankeulen,m.badiehhabibmorgan}@utwente.nl

Abstract. This position paper proposes an interactive approach for developing information extractors based on the ontology definition process with knowledge about possible (in)correctness of annotations. We discuss the problem of managing and manipulating probabilistic dependencies.

1 Introduction

(Too) much data is still inaccessible for data processing, because it is unstructured, textually embedded in documents, webpages, or text fields in databases. Information extraction (IE) is a technology capable of extracting entities, facts, and relations. IE helps to turn the web into a real 'web of data' [BHBL09].

In the Neogeography-project [HvK11b], we focus on named entity extraction (NEE) from database text fields and short messages. NEE typically consists of phases like recognition (which phrases are named entities), matching and enrichment (lookups in reference databases and dictionaries possibly adding information), and disambiguation (to which real-world object does a phrase refer).

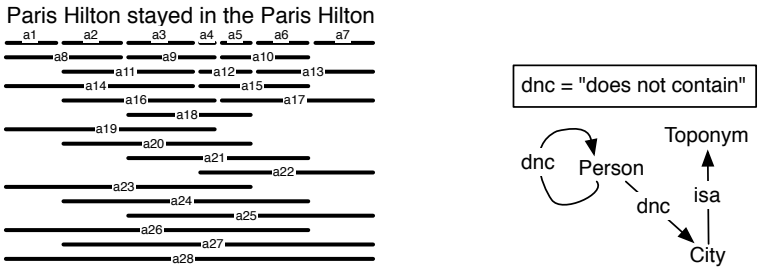
Because natural language is highly ambiguous and computers are still incapable of 'real' semantic understanding, NEE (and IE in general) is a highly imperfect process. For example, it is ambiguous how to interpret the word "Paris": it could be a first name, a city, etc. Even resolving it to a city, a lookup in GeoNames[1] learns that there are numerous other places called "Paris" besides the capital of France. In [HvK11a], we found that around 46% of toponyms[2] have two or more, 35% three or more, and 29% four or more references in GeoNames.

Although many probabilistic and fuzzy techniques abound, some aspects often remain absolute: extraction rules absolutely recognize and annotate a phrase or not, only a top item from a ranking is chosen for a next phase, etc. We envision an approach that *fundamentally* treats annotations and extracted information as uncertain throughout the process. We humans happily deal with doubt and misinterpretation every day, why shouldn't computers?

We envision developing information extractors 'Sherlock Holmes style' — *"when you have eliminated the impossible, whatever remains, however improbable, must be the truth"* — by adopting the principles and requirements below.

- Annotations are uncertain, hence we process both annotations as well as information about the uncertainty surrounding them.

[1] http://www.geonames.org
[2] A *toponym* is any name that refers to a location including, e.g., names of buildings.



(a) All possible annotations for the example sentence (b) Small example ontology

Fig. 1. Example sentence and NEE ontology

- We have an unconventional conceptual starting point, namely not "no annotations" but "there is no knowledge hence anything is possible". Fig.1(a) shows all possible annotations for an example sentence for one entity type.
- A developer gradually and interactively defines an ontology with positive and negative knowledge about the correctness of certain (combinations of) annotations. At each iteration, added knowledge is immediately applied improving the extraction result until the result is good enough (see also [vKdK09]).
- Storage, querying and manipulation of annotations should be scalable. Probabilistic databases are an attractive technology for this.

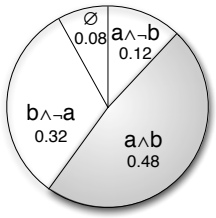
Basic forms of knowledge are the entity types one is interested in and declarations like τ_1 —*dnc*— τ_2 (no subphrase of a τ_1-phrase should be interpreted as τ_2, e.g, Person —*dnc*— City). See Fig.1(b) for a small example. We also envision application of background probability distributions, uncertain rules, etc. We hope these principles and forms of knowledge also allow for more effective handling of common problems (e.g., "you" is also the name of a place; should "Lake Como" or "Como" be annotated as a toponym).

2 Uncertain annotation model

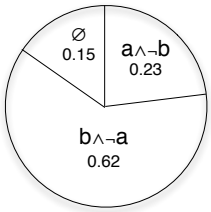
An *annotation* $a = (b, e, \tau)$ declares a phrase φ_e^b from b to e to be interpreted as entity type τ. For example, a_8 in Fig. 1(a) declares $\varphi =$ "Paris Hilton" from $b = 1$ to $e = 2$ to be interpreted as type $\tau =$ Person. An *interpretation* $I = (A, \mathcal{U})$ of a sentence s consists of an annotation set A and a structure $\mathcal{U}$ representing the uncertainty among the annotations. In the sequel, we discuss what $\mathcal{U}$ should be, but for now view it as a set of random variables (RVs) R with their dependencies.

Rather unconventionally, we don't start with an empty A, but with a 'no knowledge' point-of-view where any phrase can have any interpretation. So our initial A is $\{a \mid a = (b, e, \tau) \wedge \tau \in T \wedge \varphi_e^b$ is a phrase of $s\}$ where T is the set of possible types.

With T finite, A is also finite. More importantly, $|A| = O(klt)$ where $k = |s|$ is the length of s, l is the maximum length phrases considered, and $t = |T|$. Hence, A grows linearly in size with each. In the example of Fig.1(a), $T =$



(a) Annotations a and b independent with probabilities $P(a) = 0.6$ and $P(b) = 0.8$



(b) a and b conditioned to be mutually exclusive ($a \wedge b$ not possible)

Fig. 3. Defining a and b to be mutually exclusive means conditioning the probabilities.

$\{\mathsf{Person}, \mathsf{Toponym}, \mathsf{City}\}$ and we have $28 \cdot |T| = 84$ annotations. Even though we envision a more ingenious implementation, no probabilistic database would be severely challenged by a complete annotation set for a typical text field.

3 Knowledge application is conditioning

We explain how to 'apply knowledge' in our approach by means of the example of Fig.1, i.e., with our A with 84 (possible) annotations and an ontology only containing Person, Toponym, and City. Suppose we like to add the knowledge Person —*dnc*— City. The effect should be the removal of some annotations and adjustment of the probabilities of the remaining ones.

An initial promising idea is to store the annotations in an uncertain relation in a probabilistic database, such as MayBMS [HAKO09]. In MayBMS, the existence of each tuple is determined by an associated world set descriptor (wsd) containing a set of RV assignments from a world set table (see Fig.2). RVs are assumed independent. For example, the 3rd annotation tuple only exists when $x_8^1 = 1$ which is the case with a probability of 0.8. Each annotation can be seen as a probabilistic event, which are all independent in our starting point. Hence, we can store A by associating each annotation tuple a_i^j with one boolean RV x_i^j. Consequently, the database size is linear with $|A|$.

annotations

	b	e	type	...	wsd
a_1^1	1	1	Person	...	$\{x_1^1 = 1\}$
a_1^2	1	1	City	...	$\{x_1^2 = 1\}$
a_8^1	1	2	Person	...	$\{x_8^1 = 1\}$
	...	...	...	...	

world_set

x	v	P
x_1^1	0	0.4
x_1^1	1	0.6
x_1^2	0	0.7
x_1^2	1	0.3
x_8^1	0	0.2
x_8^1	1	0.8
...	...	...

Fig. 2. Initial annotation set stored in a probabilistic database (MayBMS-style)

Adding knowledge such as Person —*dnc*— City means that certain RVs become dependent and that certain combinations of RV assignments become impossible. Let us focus on two individual annotations a_1^2 ('"Paris" is a City) and a_8^1 ("Paris Hilton" is a Person). These two annotations become mutually exclusive. The process of adjusting the probabilities is called *conditioning* [KO08]. It boils down to redistributing the remaining probability mass. Fig.3 illustrates this for $a = a_1^2$ and $b = a_8^1$. The remaining probability mass is $1 - 0.48 = 0.52$. Hence, the

distribution of this mass over the remaining possibilities is $P(a \wedge \neg b) = \frac{0.12}{0.52} \approx 0.23$, $P(b \wedge \neg a) = \frac{0.32}{0.52} \approx 0.62$, and $P(\emptyset) = P(\neg a \wedge \neg b) = \frac{0.08}{0.52} \approx 0.15$.

A first attempt is to replace x_1^2 and x_8^1 with one fresh three-valued RV x' with the probabilities just calculated, i.e., $\text{wsd}(a_1^2) = \{x' = 1\}$ and $\text{wsd}(a_8^1) = \{x' = 2\}$ with $P(x' = 0) = 0.15$, $P(x' = 1) = 0.23$, and $P(x' = 2) = 0.62$. Unfortunately, since annotations massively overlap, we face a combinatorial explosion. For this rule, we end up with one RV with up to $2^{2 \cdot 28} = 2^{56} \approx 7 \cdot 10^{16}$ cases.

Solution directions What we are looking for in this paper is a structure that is expressive enough to capture all dependencies between RVs and at the same time allowing for scalable processing of conditioning operations. The work of [KO08] represents dependencies resulting from queries with a tree of RV assignments. We are also investigating the shared correlations work of [SDG08].

4 Conclusions

We envision an approach where information extractors are developed based on an ontology definition process for knowledge about possible (in)correctness of annotations. Main properties are treating annotations as fundamentally uncertain and interactive addition of knowledge starting from a 'no knowledge hence everything is possible' situation. The feasibility of the approach hinges on efficient storage and conditioning of probabilistic dependencies. We discuss this very problem, argue that a trivial approach doesn't work, and propose two solution directions: the conditioning approach of MayBMS and the shared correlations work of Getoor et al.

References

[BHBL09] C. Bizer, T. Heath, and T. Berners-Lee. Linked data: The story so far. *Int'l J. on Semantic Web and Information Systems (IJSWIS)*, 5(3):1–22, 2009.

[HAKO09] J. Huang, L. Antova, C. Koch, and D. Olteanu. MayBMS: a probabilistic database management system. In *Proc. of the 35th SIGMOD Int'l Conf. on Management Of Data, Providence, Rhode Island*, pages 1071–1074, 2009.

[HvK11a] M. B. Habib and M. van Keulen. Named entity extraction and disambiguation: The reinforcement effect. In *Proc. of the 5th Int'l Workshop on Management of Uncertain Data (MUD), Seatle, 29 Aug*, pages 9–16, 2011.

[HvK11b] M. B. Habib and M. van Keulen. Neogeography: The challenge of channelling large and ill-behaved data streams. Technical Report TR-CTIT-11-08, Enschede, 2011. ISSN 1381-3625, http://eprints.eemcs.utwente.nl/19854.

[KO08] C. Koch and D. Olteanu. Conditioning probabilistic databases. *Proc. VLDB Endow.*, 1(1):313–325, 2008.

[SDG08] P. Sen, A. Deshpande, and L. Getoor. Exploiting shared correlations in probabilistic databases. *Proc. VLDB Endow.*, 1(1):809–820, 2008.

[vKdK09] M. van Keulen and A. de Keijzer. Qualitative effects of knowledge rules and user feedback in probabilistic data integration. *The VLDB Journal*, 18(5):1191–1217, 2009.