Using a weakly supervised approach and lexical patterns for the KBP slot filling task

Ludovic Jean-Louis, Romaric Besançon, Olivier Ferret and Wei Wang
CEA, LIST, Vision and Content Engineering Laboratory,
Gif-sur-Yvette, F-91191 France.
{romaric.besancon, olivier.ferret, wei.wang}@cea.fr

Abstract

We present in this article the system we developed for participating to the *slot filling* task in the Knowledge Base Population (KBP) track of the 2011 Text Analysis Conference (TAC). This system is based on a weakly supervised approach and lexical patterns. In this participation, we tested more specifically the integration of an additional unsupervised relation identification component dedicated to the filtering of candidate relations.

1 Introduction

We present in this paper a description of our slot filling system for the Knowledge Base Population (KBP) track of the 2011 Text Analysis Conference (TAC). Our system is inspired by the distant supervision principle introduced by (Mintz et al., 2009) and uses lexical patterns instead of statistical classifiers. The lexical patterns are built automatically by using a pattern generalization process for each relation category. Our system only focuses on intra-sentence relations and does not rely on Web access. Our motivation was to evaluate the impact of applying unsupervised relation extraction techniques to the slot filling task. More precisely, we propose to use unsupervised methods for (1) filtering the sentences that are used for building the lexical patterns and (2) reranking the target entities of the relations.

2 Approach description

Our approach, as presented in Figure 1, is composed of two steps: a first step of *pattern learning* from instances of known relations and a step of *relation extraction* for the discovery of new relations. The first step starts with known instances of relations *R(E1,E2)* and try to find occurrences of these relations in texts, in order to cover as many different ways of expressing them as possible. We then use these occurrences to learn a set of patterns associated with the considered relation type. The second step starts with incomplete relations *R(E1,x)*, where the source entity *E1* is known and the target entity *x* has to be discovered, and searches occurrences of relation *R* involving *E1* in a collection of texts. The entity *x* is then extracted using the patterns learned in the first step. These two steps are described in more details in the following sections.

3 Relation Pattern Learning

3.1 Pattern induction

Our procedure for learning relation patterns relies on the induction of lexical patterns from example sentences containing occurrences of the considered relation. Its objective is to model the different ways a semantic relation between two entities is linguistically expressed. For instance, the two sentences in Figure 2 contain relation occurrences for the type of relation *founded_by* with the entity pairs (Charles Revson, Revlon Cosmetics) and (Mayer Lehman, Lehman Brothers investment).

A lot of algorithms for building and generalizing lexical patterns have been proposed (Ravichandran, 2005; Ruiz-Casado et al., 2007). Our approach is similar to (Pantel et al., 2004) and follows more di-

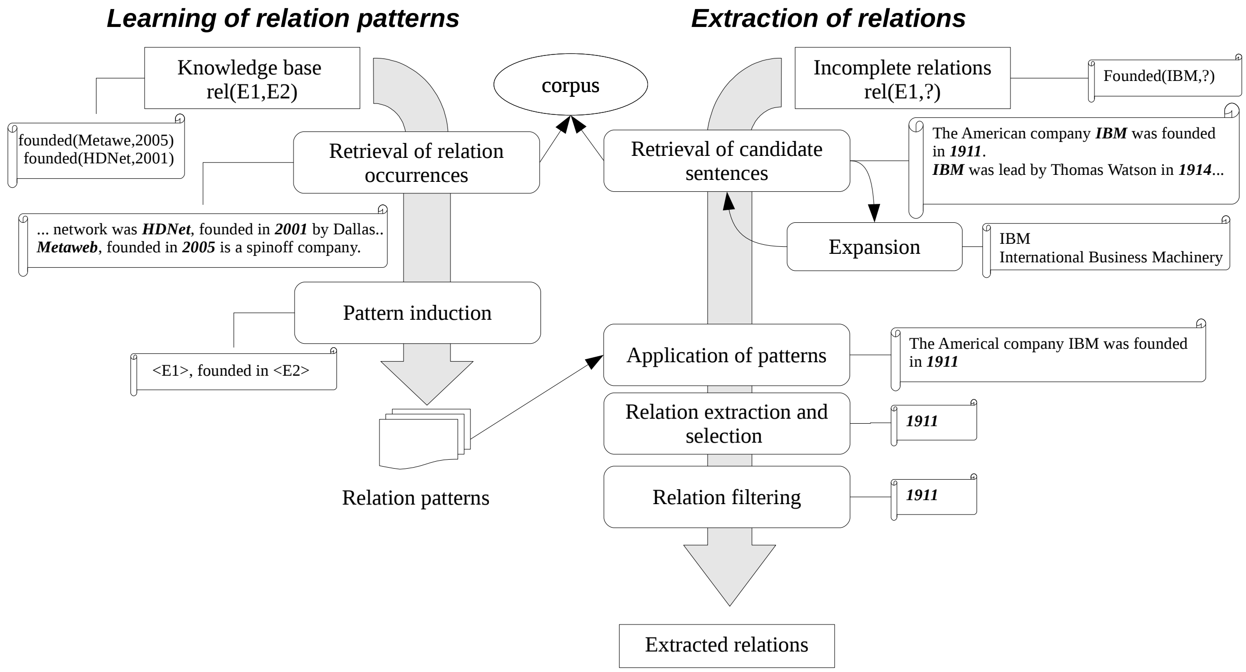

Figure 1: Overview of the system.

The glamourous cabaret chanteuse reportedly had had a romantic liaison with <source>Charles Revson</source>, the founder of <target>Revlon Cosmetics</target> ...
Lehman was a great-grandson of <source>Mayer Lehman</source>, a founder of the <target>Lehman Brothers investment</target> house ...

Figure 2: Example of two occurrences of the same relation

rectly the method of (Embarek and Ferret, 2008). Starting with a pair of entities and two sentences containing these entities and expressing the target relation, its principle is to capture the elements that are shared by the two sentences in the surrounding context of the two entities. More specifically, we identify these shared elements among three levels of linguistic information: inflected form, lemma and part-of-speech category. These levels of information are produced by the OpenNLP[1] tools, that we also use for named entity recognition. Having these three levels enables the building of more expressive patterns that represent an interesting compromise in terms of generalization between the specificity of lexicalized elements and the more general nature of part-of-speech categories.

The induction of a pattern from two occurrences of relations relies more precisely on the three following steps:

- computation of the minimal edit distance between the two example sentences, *i.e.* the minimal number of edit operations (insertion, deletion and substitution) that are necessary to turn one sentence into the other. All the operations are given here the same weight;
- optimal alignment between the two example sentences from the matrix of distances between subsequences resulting from the computation of the edit distance. The classical algorithm for achieving such alignment is enhanced for enabling a match of two words at one of the three available levels of linguistic information when two words are tested for a substitution;
- building the patterns by completing alignments with two wildcard operators when it is neces-

[1] `http://opennlp.sourceforge.net/index.html`

sary: (*s*) stands for 0 or 1 instance of any word while (*g*) represents exactly 1 instance of any word.

Table 1 shows the result of the induction of a pattern for the type of relation *founded_by* from the two sentences of Figure 2.

Table 1: Example of pattern induction

Charles Revson	,	the	founder	of		Revlon Cosmetics
Mayer Lehman	,	a	founder	of	the	Lehman Brothers investment
<source>	***,***	***DET***	***founder***	***of***	***(*s*)***	*<target>*

This example illustrates our different levels of generalization: for a word such as *of*, only the inflected form is taken. In the case of a word such as *founder*, the inflected form is taken here but the lemma level would be selected for an excerpt such as *X, the founders of ...* At a higher level of generalization, the part-of-speech category *DET* (determiner) covers *a* and *the*, which makes the resulting pattern relevant for an excerpt such as *"Charles Kettering, another founder of DELCO ..."*. This example also illustrates the use of wildcards as a substitute for any word, that is to say the highest level of generalization. As it is always possible to generalize two sentences by a pattern only made of wildcards, fixing an upper limit to the number of wildcards that can used in the generalization process is necessary for having patterns that are specific enough to the target type of relation. Moreover, as we work in open domain on general named entities, we prefer to induce a large number of specific patterns rather than a small set of very general patterns to increase precision. This argument also accounts for our choice of not generalizing patterns themselves by applying to them the generalization process described above. In practice, the maximal number of wildcards in a pattern is set to 1 in the submitted runs.

3.2 Selection of example sentences

3.2.1 Principle

In the context of distant supervision in which our work takes place, example sentences are not directly available but result from the mapping onto a corpus of relations given as pairs of entities (for instance, the pair (Ray Charles, Albany) for the type of relation *city_of_birth*). More concretely in our case, they are obtained by querying a search engine with pairs of entities corresponding to relations of the considered relation type. In our experiments, we used the *Lucene*[2] search engine, with an indexing process taking into account the specific needs of our task: documents were segmented into segments of three sentences using a sliding window and the resulting segments were indexed by their plain words, their named entities and the named entity types. The same index is used for pattern learning and relation extraction. The document segments retrieved by *Lucene* are then restricted to sentences that actually contain a pair of entities, since the patterns are only generated for intra-sentence relations. The nature of the restrictions applied to the results of the search engine has of course a direct impact on the quantity and the precision of final patterns: the stricter they are, the less example sentences we get but the better the induced patterns are. (Agirre et al., 2009) adds for instance the constraint that the two entities of a relation pair must not be separated in a sentence by more than ten words.

3.2.2 Filtering with APSS

Another important issue concerning the induction of patterns is its computational cost. This process is performed by considering each pair of example sentences, which can have a too high computational cost when the number of example sentences is significant: for 10,000 examples, around 50 millions of distinct pairs of sentences have to be compared ($n(n-1)/2$ exactly). The most straightforward way to solve this problem is to reduce drastically the number of example sentences before the induction of patterns. However, such solution implies having a smaller coverage of the different linguistic expressions of a type of relation if this reduction is performed blindly. Our solution to this problem exploits the fact that two sentences sharing a small number of words will not lead to an interesting pattern. The distance we use for inducing patterns – the edit distance – was chosen because of its ability to take into account the order of words but of course, it first depends on the number of words the two compared sentences share. As a consequence, the *a pri-*

[2] `http://lucene.apache.org`

ori filtering of example sentence pairs can be based on the computation of a similarity measure between sentences that only exploits a *bag of words* representation of them, as the *cosine* measure, and the application of a minimal threshold to these similarity values for discarding pairs that are not likely to lead to an interesting pattern. The *cosine* measure can be computed efficiently, either approximately, by using methods such as *Local Sensitive Hashing* (Gionis et al., 1999), or without any approximation but the necessity to fix an *a priori* minimal similarity threshold, which corresponds to our case. We chose more precisely the *All Pairs Similarity Search* (APSS) algorithm proposed in (Bayardo et al., 2007) which computes the *cosine* measure only for the pairs of objects – example sentences in our case – whose similarity is higher or equal to a fixed threshold. This algorithm relies on the incremental indexing of the objects whose similarity has to be evaluated and implements a set of optimizations for this indexing process based on both data gathered *a priori* about the features of these objects and their sorting according to these features.

More precisely, we have two levels of filtering based on APSS. Learning patterns from a large number of example sentences often leads to several occurrences of the same pattern, either because an example sentence is found in several documents or because there are several occurrences of the same linguistic expression of a type of relation with different entity values (*Obama's height is 1.87m*; *Sarkozy's height is 1.65m*). As a consequence, we first apply a high similarity threshold for identifying and discarding identical sentences; second, a lower threshold aims at checking that sentences are similar enough for inducing a meaningful pattern. In order to further reduce the number of comparisons between example sentences, the similarity values resulting from APSS are exploited for clustering these sentences by relying on the Markov Clustering algorithm (van Dongen, 2000). Finally, patterns are induced only from sentences that are part of the same cluster.

3.2.3 Filtering with unsupervised relation identification

With the previous method, the sets of relation occurrences used for generating the relation patterns are collected based on a simplistic assumption: *"every sentence containing a given pair of entities associated with a relation, is a valid example for this relation"*. This assumption is not always valid and sometimes implies taking incorrect relation occurrences for inducing the patterns. The following examples show two relation occurrences that do not contain a relation between the entities (1), or do not express one of the relation of interest for the slot filling task (2).

1. *"Larry Page has employed Eric Schmidt, he will be working as the new chairman at Google."*
2. *"Even Larry Page and Sergey Brin wanted to sell Google to Yahoo."*

In order to tackle this issue, we integrated in our second run (LVIC 2) an approach for filtering occurrences of relations that do not express an actual relation between a given pair of entities, similarly to (Riedel et al., 2010). This approach was initially implemented for unsupervised relation extraction and is further described in (Wang et al., 2011). It relies on a statistical classifier (CRF) to tag a given sentence as a valid/invalid occurrence of relation without making any hypothesis about the type of the relation. We used more specifically this component to identify and remove relation occurrences such as (1) in the previous example. The remaining relation occurrences were kept for the pattern induction process described in section 3.1. Our intuition was that the filtering of relation occurrences shall improve the precision of the induced patterns with a limited impact on their recall.

4 Relation Extraction

4.1 Selection of relation candidates

The extraction of new relations is done from the existing types of relations and given entities: we are searching to add knowledge to an existing knowledge base by adding missing attributes to entities already in the KB. The first step of relation extraction is the selection of candidate sentences that are likely to contain the expression of a relation. It starts from a query containing one entity associated with its type and the type of the target entity. The retrieval of the candidate sentences is performed using *Lucene*, with the same index as in the pattern learning step. We also performed a kind of query expansion focusing on the source entity. Indeed, the source entity sometimes appears in the target base of documents under a slightly different form than in the query: for instance, *Bill Clinton* is often used instead of *William Jefferson Blythe III Clinton*, which is the normalized form of the entity in the KB. The expansion is based on an expansion database automatically built from Wikipedia[3]: each entity is expanded by all the formulations extracted from the redirection pages of Wikipedia for this entity. This expansion database contains alternative forms for 2.4 million entities and, starting from an entity such as *Barack Obama*, makes it possible to retrieve documents referring to {*B. Hussein Obama, Barack H. Obama Junior, Barack Obama Jr, Barack Hussein Obama Jr., etc.*}.

As we only deal with intra-sentence relations, we check, after the retrieval of the document segments, that the source entity co-occurs with a possible target entity in a sentence. The detection of the target entity is based on the presence of a named entity of compatible type but also on reference lists of values for types, as in the relation *per:title*, that do not correspond to named entities.

4.2 Applying patterns and selecting target entities

The patterns learned in the first step are applied to all selected candidate sentences. The target entities extracted by these patterns are gathered and sorted. We only keep the most frequent entities: our hypothesis is that the more relevant the target entities are, the more often they appear in documents together with the source entity. For relations with a unique target entity (*e.g. date_of_birth*), we choose the most frequent entity. For relations with several possible target values (*e.g. places_of_residence*), an arbitrary number of three values is taken since we do not have knowledge (either prior knowledge or learned from documents) about the correct number of values.

Finally, a filter is applied to the target entities to check the compatibility of their value with constraints relative to the type of information we search. These constraints are defined by lists of values or regular expression. For instance, we check that the country of birth of a person is part of a list of known countries as the named entity type for the target entity – location – is not specific enough to guarantee the validity of the found information.

4.3 Using unsupervised relation identification to filter target entities

Considering the n most frequent answers as the most relevant answers may lead to erroneous target entities since these answers might be part of invalid occurrences of relations. In order to filter out these possible wrong relation instances, we used an additional filtering of candidate sentences as a criterion for re-ranking answers. Our motivation is that answers that appear more frequently in valid candidate relations shall be ranked prior to others. Precisely, we believe that target entities that are contained in sentences that are tagged as valid by the unsupervised relation classifier presented in section 3.2.3 shall be ranked first. As a consequence, when computing the frequency of a given answer, we add a bonus (equivalent to +1 occurrence) to each target entity that appears in a sentence that is found valid by this classifier.

5 Evaluation

We submitted three runs to the slot filling task. None of these runs had access to the Web and all of them only focus on intra-sentence relations. We first provide a short description of each run and then, a summary of our results.

Our three runs have the following characteristics:

- the first run (LVIC 1) is based on the initial approach presented in section 2 and does not rely on unsupervised relation extraction techniques;
- the second run (LVIC 2) is similar to LVIC 1 but uses unsupervised relation identification to filter out example sentences in the pattern learning process, as described in section 3.2.3;
- the third run (LVIC 3) is similar to LVIC 1 but uses unsupervised relation identification to filter the target entities obtained by the application of the patterns, as described in section 4.3.

[3]More precisely, we used the Wikipedia dump provided by the university of New York `http://nlp.cs.nyu.edu/wikipedia-data`.

We report in Table 2 the results of our three runs together with the top and median scores for participating systems without Web access. We also give the results (LVIC 1_2010), reported in (Jean-Louis et al., 2011), of the application to the 2010 evaluation data of the same system as LVIC 1 without the detection of NIL slot values. In this case, NIL slots and slots for which none 2010 participant had found a target entity were not taken into account. LVIC X_wn are the subsets of the LVIC X results that fulfill the same conditions for the 2011 data.

Table 2: Overall results of our system

	Recall	**Precision**	**F1-score**
Human	86.18%	72.59%	78.81%
Top score	49.17%	12.59%	20.05%
Median	10.31%	16.51%	12.69%
LVIC 1	10.26%	4.61%	6.36%
LVIC 2	9.95%	4.54%	6.24%
LVIC 3	10.16%	4.56%	6.30%
LVIC 1_2010	18.67%	16.87%	17.72%
LVIC 1_wn	12.21%	10.54%	11.31%
LVIC 2_wn	12.03%	10.5%	11.21%
LVIC 3_wn	11.67%	10.07%	10.81%

First, according to the overall results of the evaluation, it seems that using Web access has a significant positive impact as the top score using such an access obtained +9% F1-score compared to the top system without Web access. These results also show that the performance of our system is quite low and below the median score. This may be due to the fact that our system only focuses on intra-sentence relations and does not process any inter-sentence relations (which were considered to form 40% of the relations in KBP 2010). A brief analysis of human annotated answers of KBP 2011 seems to confirm this fact: we noticed that a significant part of the relations were inter-sentence relations. However, even if our results are quite similar for all our runs, they are not exactly identical: we observed that 85% of answers were common between LVIC 2 and LVIC 3, 93% common between LVIC 1 and LVIC 3 and 90% between LVIC 1 and LVIC 2. Hence, the different uses of our unsupervised relation extraction component actually have an impact on found answers but the global low level of our results makes the analysis of this impact difficult.

Table 3 provides more detailed results about two important steps of our system[4]: the initial retrieval of candidate sentences and the extraction of relations resulting from the application of relation patterns. The evaluation of the retrieval of candidate sentences is more precisely characterized by the percentage of documents retrieved by *Lucene* that are part of the reference documents[5], *i.e.* documents in which correct target entities were found by KBP participants[6]. This *Doc. Rec.* measure is given for each type of relations both for our runs of the 2011 evaluation and for the experiments reported in (Jean-Louis et al., 2011) with roughly the same system on the 2010 data. The global value of *Doc. Rec.* for 2010 is equal to 84.24% while it is equal to 71.35% for 2011. This difference can be explained by the number of candidate sentences retrieved for each target entity: 1,000 for the 2010 data and between 50 and 300 for the 2011 data[7]. Taking 1,000 candidate relations for the 2011 data leads to a *Doc. Rec.* of 80.47%, which is not too far from the recall for the 2010 data.

The second main measure of Table 3 is *Rel. Rec.*, the percentage of relations in the reference that are extracted by our system before their filtering and their ranking. For a target entity, these relations are the candidate sentences that match one of the patterns learned for the relation type corresponding to this target entity. As for *Doc. Rec.*, this measure is given for each type of relations and for the 2010 and 2011 data. Moreover, for the 2011 data, we distinguish the results of LVIC 1 and LVIC 3 on one side and the results of LVIC 2 on the other side as their sets of relation patterns are different. Globally, *Rel. Rec.* is equal to 41.33% for LVIC 1 and LVIC 3, to 35.32% for LVIC 2 and to 37.55% for the 2010 data. The difference between LVIC 1 & 3 and LVIC 2 is not surprising and means that our fil-

[4]These results detail the LVIC X_wn lines of Table 2.

[5]Candidate sentences are evaluated indirectly through the documents they come from.

[6]This means that we do not consider slots with NIL as value and slots for which none KBP participant found a value.

[7]50 for the *per:title* relation, 100 for a subset of relations and 300 for the others. Due to a failure of our cluster, we had to limit the number of candidate sentences to process for the 2011 evaluation.

Table 3: Detailed results for different steps, by relation type. *Doc. Rec.* and *Rel. Rec.* are percentages

Relation type	Doc. Rec. 2010	Doc. Rec. 2011	Rel. Rec. 2010	Rel. Rec. 2011 (1,3)	Rel. Rec. 2011 (2)	Nb Ref. 2010	Nb Ref. 2011
org:alternate_names	89.17	76.68	33.33	15.42	15.42	120	244
org:city_of_headquarters	90.12	76.09	59.26	88.1	54.76	81	46
org:country_of_headquarters	91.04	50	55.22	79.25	33.96	67	60
org:dissolved	100	0	25	0	0	4	1
org:founded_by	95.45	40	31.82	85.71	85.71	28	26
org:founded	92.86	80	53.57	81.25	81.25	22	16
org:member_of	100	37.5	100	50	25	2	16
org:members	77.78	100	11.11	40	50	9	10
org:number_of_employees_members	90.48	75	23.81	0	0	21	11
org:parents	96.67	89.13	43.33	38.78	38.78	30	49
org:political_religious_affiliation	78.57	33.33	64.29	0	0	14	3
org:shareholders	66.67	100	33.33	0	0	3	22
org:stateorprovince_of_headquarters	92.65	77.14	63.24	60	40	68	35
org:subsidiaries	82.69	77.14	28.85	36.36	36.36	52	44
org:top_members_employees	91.48	69.3	37.22	50.87	50.87	223	235
org:website	78.26	75	30.43	0	0	23	22
per:age	85.32	82.35	32.11	0	0	109	34
per:alternate_names	61.63	79.1	11.63	0	0	86	67
per:cause_of_death	100	66.67	0	0	0	2	3
per:charges	61.54	81.25	0	0	0	13	23
per:children	72	100	16	0	0	25	25
per:cities_of_residence	77.59	56.52	34.48	8.7	8.7	58	23
per:city_of_birth	69.23	75	15.38	0	0	13	8
per:city_of_death	100	100	100	0	0	1	1
per:countries_of_residence	73.53	62.5	20.59	81.25	45.83	34	50
per:country_of_birth	82.35	14.29	5.88	100	16.67	17	7
per:country_of_death		100		100	0	0	1
per:date_of_birth	90	100	20	0	0	20	5
per:date_of_death	100	100	0	0	0	1	8
per:employee_of	84.21	66.06	29.32	50	51.79	133	182
per:member_of	82.42	71.72	36.26	50.93	50.93	91	112
per:origin	81.58	70.77	42.11	67.21	65.57	76	68
per:other_family	86.67	100	33.33	0	0	30	9
per:parents	78.13	33.33	9.38	0	0	64	4
per:religion	85.71	88.89	57.14	42.86	28.57	7	9
per:schools_attended	87.50	58.82	37.50	0	0	16	22
per:siblings	78.26	88.89	20.29	0	0	69	9
per:spouse	80	61.11	35.56	16.67	16.67	45	18
per:stateorprovince_of_birth	80	100	50	0	0	10	2
per:stateorprovince_of_death	100	0	100	0	0	1	0
per:states_or_provinces_of_residence	84.21	0	50	78.79	72.73	38	0
per:title	84.55	76.55	52.77	48.43	36.61	343	402

Doc. Rec. 201X: recall of sentence retrieval in terms of reference documents for 201X. *Rel. Rec. 2010*: recall of candidate sentences in terms of reference documents for 2010. *Rel. Rec. 2011 (Y)*: recall of candidate sentences of run LVIC Y in terms of reference documents for 2011. *Nb Ref. 201X*: number of reference relations for 201X.

tering of example relations for pattern learning is sometimes too strict. Table 3 shows more precisely that the use of such filtering should be adapted to the type of relations as it is positive for some of them (org:members and per:employee_of), has no effect for a large part of them (such as org:founded or per:member_of) and is negative for the others (such as org:city_of_headquarters or per:country_of_birth). The negative impact for some type of relations certainly comes from the fact that the generic classifier of (Wang et al., 2011) was trained for relations expressed by a verb between the two entities, which is not the main mode of expression of some KBP types of relations.

The difference between 2010 and 2011 results is more surprising as Table 2 clearly shows that the overall results for the 2010 data are significantly higher than the results for the 2011 data, even if we take into account the problem of NIL slots. However, it is easier to interpret by noting that a particular slot can be filled by several identical target entities coming from different documents. As a consequence, *Rel. Rec.* can have a higher value for system 1 than for system 2 while the F1-score of system 2 is higher than the F1-score of system 1 if system 2 finds a correct value for a larger number of slots than system 1 while system 1 finds a larger number of occurrences of a correct slot value than system 2 but for a fewer number of slots. That is what seems to happen for our 2011 results compared to our 2010 results. This analysis is also confirmed by the fact that *Rel. Rec.* has a null value for a significantly higher number of relation types in our 2011 results (LVIC 1 & 3: 20 null values; LVIC 2: 21 null values) than in our 2010 results (3 null values). Further investigations have to be made to understand the origin of this difference.

Finally, another weak point of our system is its basic strategy for processing NIL queries: it returns a NIL answer for a query when it fails to find an answer after the application of the relation patterns (or when no candidate document was found in the corpus). It seems that this strategy is not sufficient as our system did not identify a sufficient number of NIL answers and therefore, provided many incorrect answers: on average, we considered that 30% of the answers were NILs.

6 Conclusion

We have presented in this article our system for the TAC-KBP 2011 *slot filling* task. This system is based on a weakly supervised approach in which the examples are limited to pairs of entities in relation. The extraction of relations is performed by the application of lexico-syntactic patterns that are learned from occurrences of relations automatically selected from the entity pairs of the examples and used to model relation types. The results we obtained with our submitted runs can not be considered as satisfactory but a brief analysis of them pointed out that our weak points for this evaluation are the capacities to handle inter-sentence relations and to spot NIL answers. We will focus on these elements in the future developments of our system.

References

Eneko Agirre, Angel Chang, Daniel Jurafsky, Christopher Manning, Valentin Spitkovsky, and Eric Yeh. 2009. Stanford-UBC at TAC-KBP. In *Second Text Analysis Conference (TAC 2009)*, Gaithersburg, Maryland, USA.

Roberto Bayardo, Yiming Ma, and Ramakrishnan Srikant. 2007. Scaling Up All Pairs Similarity Search. In *16th International Conference on World Wide Web (WWW'07)*, pages 131–140, Banff, Alberta, Canada.

Mehdi Embarek and Olivier Ferret. 2008. Learning patterns for building resources about semantic relations in the medical domain. In *6th Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco.

A. Gionis, P. Indyk, and R. Motwani. 1999. Similarity Search in High Dimensions via Hashing. In *25th International Conference on Very Large Data Bases (VLDB'99)*, pages 518–529, Edinburgh, Scotland, UK.

Ludovic Jean-Louis, Romaric Besançon, Olivier Ferret, and Adrien Durand. 2011. A weakly supervised approach for large-scale relation extraction. In *3rd International Conference on Knowledge Discovery and Information Retrieval (KDIR 2011)*, pages 94–103, Paris, France, october.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL-IJCNLP'09*, pages 1003–1011, Suntec, Singapore.

Patrick Pantel, Deepak Ravichandran, and Eduard Hovy. 2004. Towards Terascale Knowledge Acquisition. In *20th International Conference on Computational*

Linguistics (COLING'04), pages 771–777, Geneva, Switzerland.

Deepak Ravichandran. 2005. *Terascale Knowledge Acquisition*. Ph.D. thesis, University of Southern California, Los Angeles, CA, USA.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling Relations and Their Mentions without Labeled Text. In *Machine Learning and Knowledge Discovery in Databases*, LNCS, pages 148–163.

Maria Ruiz-Casado, Enrique Alfonseca, and Pablo Castells. 2007. Automatising the Learning of Lexical Patterns: an Application to the Enrichment of WordNet by Extracting Semantic Relationships from Wikipedia. *Data Knowledge Engineering*, 61:484–499, June.

Stijn van Dongen. 2000. *Graph Clustering by Flow Simulation*. Ph.D. thesis, University of Utrecht.

Wei Wang, Romaric Besançon, Olivier Ferret, and Brigitte Grau. 2011. Filtering and clustering relations for unsupervised information extraction in open domain. In *20th ACM Conference on Information and Knowledge Management (CIKM 2011)*, pages 1405–1414, Glasgow, Scotland, UK.