A Rote Extractor with Edit Distance-based Generalisation and Multi-corpora Precision Calculation

Enrique Alfonseca[12] **Pablo Castells**[1] **Manabu Okumura**[2] **Maria Ruiz-Casado**[12]

[1]Computer Science Deptartment
Univ. Autónoma de Madrid
Enrique.Alfonseca@uam.es
Pablo.Castells@uam.es
Maria.Ruiz@uam.es

[2]Precision and Intelligence Laboratory
Tokyo Institute of Technology
enrique@lr.pi.titech.ac.jp
oku@pi.titech.ac.jp
maria@lr.pi.titech.ac.jp

Abstract

In this paper, we describe a rote extractor that learns patterns for finding semantic relationships in unrestricted text, with new procedures for pattern generalization and scoring. These include the use of part-of-speech tags to guide the generalization, Named Entity categories inside the patterns, an edit-distance-based pattern generalization algorithm, and a pattern accuracy calculation procedure based on evaluating the patterns on several test corpora. In an evaluation with 14 entities, the system attains a precision higher than 50% for half of the relationships considered.

1 Introduction

Recently, there is an increasing interest in automatically extracting structured information from large corpora and, in particular, from the Web (Craven et al., 1999). Because of the difficulty of collecting annotated data, several procedures have been described that can be trained on unannotated textual corpora (Riloff and Schmelzenbach, 1998; Soderland, 1999; Mann and Yarowsky, 2005). An interesting approach is that of rote extractors (Brin, 1998; Agichtein and Gravano, 2000; Ravichandran and Hovy, 2002), which look for textual contexts that happen to convey a certain relationship between two concepts.

In this paper, we describe some contributions to the training of Rote extractors, including a procedure for generalizing the patterns, and a more complex way of calculating their accuracy. We first introduce the general structure of a rote extractor and its limitations. Next, we describe the proposed modifications (Sections 2, 3 and 4) and the evaluation performed (Section 5).

1.1 Rote extractors

According to the traditional definition of rote extractors (Mann and Yarowsky, 2005), they estimate the probability of a relationship $r(p, q)$ given the surrounding context $A_1pA_2qA_3$. This is calculated, with a training corpus T, as the number of times that two related elements $r(x, y)$ from T appear with that same context $A_1xA_2yA_3$, divided by the total number of times that x appears in that context together with any other word:

$$P(r(p,q)|A_1pA_2qA_3) = \frac{\sum_{x,y \epsilon r} c(A_1xA_2yA_3)}{\sum_{x,z} c(A_1xA_2zA_3)} \quad (1)$$

x is called the *hook*, and y the *target*. In order to train a Rote extractor from the web, this procedure is usually followed (Ravichandran and Hovy, 2002):

1. Select a pair of related elements to be used as seed. For instance, *(Dickens,1812)* for the relationship *birth year*.
2. Submit the query *Dickens AND 1812* to a search engine, and download a number of documents to build the training corpus.
3. Keep all the sentences containing both elements.
4. Extract the set of contexts between them and identify repeated patterns. This may just be the m characters to the left or to the right, (Brin, 1998), the longest common substring of several contexts (Agichtein and Gravano, 2000), or all substrings obtained with a suffix tree constructor (Ravichandran and Hovy, 2002).
5. Download a separate corpus, called *hook corpus*, containing just the hook (in the example, *Dickens*).
6. Apply the previous patterns to the hook corpus, calculate the precision of each pattern

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 9–16,
Sydney, July 2006.

in the following way: the number of times it identifies a target related to the hook divided by the total number of times the pattern appears.

7. Repeat the procedure for other examples of the same relationship.

To illustrate this process, let us suppose that we want to learn patterns to identify birth years. We may start with the pair (*Dickens*, 1812). From the downloaded corpus, we extract sentences such as

Dickens was born in 1812
Dickens (1812 - 1870) was an English writer
Dickens (1812 - 1870) wrote Oliver Twist

The system identifies that the contexts of the last two sentences are very similar and chooses their longest common substring to produce the following patterns:

```
<hook> was born in <target>
<hook> ( <target> - 1870 )
```

In order to measure the precision of the extracted patterns, a new corpus is downloaded using the hook *Dickens* as the only query word, and the system looks for appearances of the patterns in the corpus. For every occurrence in which the hook of the relationship is Dickens, if the target is 1812 it will be deemed correct, and otherwise it will be deemed incorrect (e.g. in *Dickens was born in Portsmouth*).

1.2 Limitations and new proposal

We have identified the following limitations in this algorithm: firstly, to our knowledge, no Rote extractor allows for the insertion of wildcards (e.g. *) in the extracted patterns. Ravichandran and Hovy (2002) have noted that this might be dangerous if the wildcard matches unrestrictedly incorrect sentences. However, we believe that the precision estimation that is performed at the last step of the algorithm, using the *hook corpus*, may be used to rule out the dangerous wildcards while keeping the useful ones.

Secondly, we believe that the procedure for calculating the precision of the patterns may be somewhat unreliable in a few cases. For instance, Ravichandran and Hovy (2002) report the following patterns for the relationships Inventor, Discoverer and Location:

Relation	Prec.	Pattern
Inventor	1.0	<target> 's <hook> and
Inventor	1.0	that <target> 's <hook>
Discoverer	0.91	of <target> 's <hook>
Location	1.0	<target> 's <hook>

In this case, it can be seen that the same pattern (the genitive construction) may be used to indicate several different relationships, apart from the most common use indicating possession. However, they all receive very high precision values. The reason is that the patterns are only evaluated for the same hook for which they were extracted. Let us suppose that we obtain the pattern for Location using the pairs *(New York, Chrysler Building)*. The genitive construction can be extracted from the context *New York's Chrysler Building*. Afterward, when evaluating it, only sentences containing *<target>'s Chrysler Building* are taken into account, which makes it unlikely that the pattern is expressing a relationship other than Location, so the pattern will receive a high precision value.

For our purposes, however, we need to collect patterns for several relations such as *writer-book*, *painter-picture*, *director-film*, *actor-film*, and we want to make sure that the obtained patterns are only applicable to the desired relationship. Patterns like *<target> 's <hook>* are very likely to be applicable to all of these relationships at the same time, so we would like to be able to discard them automatically.

Hence, we propose the following improvements for a Rote extractor:

- A new pattern generalization procedure that allows the inclusion of wildcards in the patterns.
- The combination with Named Entity recognition, so people, locations, organizations and dates are replaced by their entity type in the patterns, in order to increase their degree of generality. This is in line with Mann and Yarowsky (2003)'s modification, consisting in replacing all numbers in the patterns with the symbol ####.
- A new precision calculation procedure, in a way that the patterns obtained for a given relationship are evaluated on the corpus for different relationships, in order to improve the detection of over-general patterns.

2 Proposed pattern generalization procedure

To begin with, for every appearance of a pair of concepts, we extract a context around them. Next, those contexts are generalized to obtain the parts that are shared by several of them. The procedure is detailed in the following subsections.

Birth year:

```
BOS/BOS <hook> (/( <target> -/- number/entity )/) EOS/EOS
BOS/BOS <hook> (/( <target> -/- number/entity )/) British/JJ writer/NN
BOS/BOS <hook> was/VBD born/VBN on/IN the/DT first/JJ of/IN time_expr/entity ,/, <target> ,/, at/IN location/entity ,/, of/IN
BOS/BOS <hook> (/( <target> -/- )/) a/DT web/NN guide/NN
```

Birth place:

```
BOS/BOS <hook> was/VBD born/VBN in/IN <target> ,/, in/IN central/JJ location/entity ,/,
BOS/BOS <hook> was/VBD born/VBN in/IN <target> date/entity and/CC moved/VBD to/TO location/entity
BOS/BOS Artist/NN :/, <hook> -/- <target> ,/, location/entity (/( number/entity -/-
BOS/BOS <hook> ,/, born/VBN in/IN <target> on/IN date/entity ,/, worked/VBN as/IN
```

Author-book:

```
BOS/BOS <hook> author/NN of/IN <target> EOS/EOS
BOS/BOS Odysseus/NNP :/, Based/VBN on/IN <target> ,/, <hook> 's/POS epic/NN from/IN Greek/JJ mythology/NN
BOS/BOS Background/NN on/IN <target> by/IN <hook> EOS/EOS
did/VBD the/DT circumstances/NNS in/IN which/WDT <hook> wrote/VBD "/'' <target> "/'' in/IN number/entity ,/, and/CC
```

Capital-country:

```
BOS/BOS <hook> is/VBZ the/DT capital/NN of/IN <target> location/entity ,/, location/entity correct/JJ time/NN
BOS/BOS The/DT harbor/NN in/IN <hook> ,/, the/DT capital/NN of/IN <target> ,/, is/VBZ number/entity of/IN location/entity
BOS/BOS <hook> ,/, <target> EOS/EOS
BOS/BOS <hook> ,/, <target> -/- organization/entity EOS/EOS
```

Figure 1: Example patterns extracted from the training corpus for each several kinds of relationships.

2.1 Context extraction procedure

After selecting the sentences for each pair of related words in the training set, these are processed with a part-of-speech tagger and a module for Named Entity Recognition and Classification (NERC) that annotates people, organizations, locations, dates, relative temporal expressions and numbers. Afterward, a context around the two words in the pair is extracted, including (a) at most five words to the left of the first word; (b) all the words in between the pair words; (c) at most five words to the right of the second word. The context never jumps over sentence boundaries, which are marked with the symbols BOS (*Beginning of sentence*) and EOS (*End of sentence*). The two related concepts are marked as <hook> and <target>. Figure 1 shows several example contexts extracted for the relationships *birth year*, *birth place*, *writer-book* and *capital-country*.

Furthermore, for each of the entities in the relationship, the system also stores in a separate file the way in which they are annotated in the training corpus: the sequences of part-of-speech tags of every appearance, and the entity type (if marked as such). So, for instance, typical PoS sequences for names of authors are "NNP"[1] (surname) and "NNP NNP" (first name and surname). A typical entity kind for an author is person.

2.2 Generalization pseudocode

In order to identify the portions in common between the patterns, and to generalize them, we apply the following pseudocode (Ruiz-Casado et al., in press):

1. Store all the patterns in a set $\mathcal{P}$.
2. Initialize a set $\mathcal{R}$ as an empty set.
3. While $\mathcal{P}$ is not empty,
   (a) For each possible pair of patterns, calculate the distance between them (described in Section 2.3).
   (b) Take the two patterns with the smallest distance, p_i and p_j.
   (c) Remove them from $\mathcal{P}$, and add them to $\mathcal{R}$.
   (d) Obtain the generalization of both, p_g (Section 2.4).
   (e) If p_g does not have a wildcard adjacent to the hook or the target, add it to $\mathcal{P}$.
4. Return $\mathcal{R}$

At the end, $\mathcal{R}$ contains all the initial patterns and those obtained while generalizing the previous ones. The motivation for step (e) is that, if a pattern contains a wildcard adjacent to either the hook or the target, it will be impossible to know where it starts or ends. For instance, when applying the pattern <hook> wrote * <target> to a text, the wildcard prevents the system from guessing where the title of the book starts.

2.3 Edit distance calculation

So as to calculate the similarity between two patterns, a slightly modified version of the dynamic programming algorithm for *edit-distance* calculation (Wagner and Fischer, 1974) is used. The distance between two patterns A and B is defined as the minimum number of changes (insertion, addition or replacement) that have to be done to the first one in order to obtain the second one.

The calculation is carried on by filling a matrix $\mathcal{M}$, as shown in Figure 2 (left). At the same

[1] All the PoS examples in this paper are done with Penn Treebank labels (Marcus et al., 1993).

A: `wrote the well known novel`
B: `wrote the classic novel`

$\mathcal{M}$	**0**	**1**	**2**	**3**	**4**
0	0	1	2	3	4
1	1	0	1	2	3
2	2	1	0	1	2
3	3	2	1	1	2
4	4	3	2	2	2
5	5	4	3	3	2

$\mathcal{D}$	**0**	**1**	**2**	**3**	**4**
0		I	I	I	I
1	R	**E**	I	I	I
2	R	R	**E**	I	I
3	R	R	R	**U**	I
4	R	R	R	**R**	U
5	R	R	R	R	**E**

Figure 2: Example of the edit distance algorithm. A and B are two word patterns; $\mathcal{M}$ is the matrix in which the edit distance is calculated, and $\mathcal{D}$ is the matrix indicating the choice that produced the minimal distance for each cell in $\mathcal{M}$.

time that we calculate the edit distance matrix, it is possible to fill in another matrix $\mathcal{D}$, in which we record which of the choices was selected at each step: insertion, deletion, replacement or no edition. This will be used later to obtain the generalized pattern. We have used the following four characters:

- `I` means that it is necessary to insert a token in the first pattern to obtain the second one.
- `R` means that it is necessary to remove a token.
- `E` means that the corresponding tokens are equal, so no edition is required.
- `U` means that the corresponding tokens are unequal, so a replacement has to be done.

Figure 2 shows an example for two patterns, A and B, containing respectively 5 and 4 tokens. $\mathcal{M}(5,4)$ has the value 2, indicating the distance between the two complete patterns. For instance, the two editions would be replacing `well` by `classic` and removing `known`.

2.4 Obtaining the generalized pattern

After calculating the edit distance between two patterns A and B, we can use matrix $\mathcal{D}$ to obtain a generalized pattern, which should maintain the common tokens shared by them. The procedure used is the following:

- Every time there is an insertion or a deletion, the generalized pattern will contain a wildcard, indicating that there may be anything in between.
- Every time there is replacement, the generalized pattern will contain a disjunction of both tokens.
- Finally, in the positions where there is no edit operation, the token that is shared between the two patterns is left unchanged.

The patterns in the example will produced the generalized pattern

```
Wrote the well known novel
Wrote the classic    novel
_____________________
Wrote the well|classic * novel
```

The generalization of these two patterns produces one that can match a wide variety of sentences, so we should always take care in order not to over-generalize.

2.5 Considering part-of-speech tags and Named Entities

If we consider the result in the previous example, we can see that the disjunction has been made between an adverb and an adjective, while the other adjective has been deleted. A more natural result, with the same number of editing operations as the previous one, would have been to delete the adverb to obtain the following generalization:

```
Wrote the well known novel
Wrote the classic    novel
_____________________
Wrote the * known|classic novel
```

This is done taking into account part-of-speech tags in the generalization process. In this way, the edit distance has been modified so that a replacement operation can only be done between words of the same part-of-speech.[2] Furthermore, replacements are given an edit distance of 0. This favors the choice of replacements with respect to deletions and insertions. To illustrate this point, the distance between `known|classic/JJ` and `old/JJ`

[2]Note that, although our tagger produces the very detailed PennTreebank labels, we consider that all nouns (NN, NNS, NNP and NNPS) belong to the same part-of-speech class, and the same for adjectives, verbs and adverbs.

Hook	Birth	Death	Birth place	Author of	Director of	Capital of
Charles Dickens	1812	1870	Portsmouth	{Oliver Twist, The Pickwick Papers, Nicholas Nickleby, David Copperfield...}	**None**	**None**
Woody Allen	1935	**None**	Brooklin	**None**	{Bananas, Annie_Hall, Manhattan, ... }	**None**
Luanda	**None**	**None**	**None**	**None**	**None**	Angola

Table 1: Example rows in the input table for the system.

will be set to 0, because both tokens are adjectives. In other words, the d function is redefined as:

$$d(A[i], B[j]) = \begin{cases} 0 & \text{if } PoS(A[i]) = PoS(B[j]) \\ 1 & \text{otherwise} \end{cases} \quad (2)$$

Note that all the entities identified by the NERC module will appear with a PoS tag of *entity*, so it is possible to have a disjunction such as `location|organization/entity` in a generalized pattern (See Figure 1).

3 Proposed pattern scoring procedure

As indicated above, if we measure the precision of the patterns using a *hook corpus*-based approach, the score may be inadvertently increased because they are only evaluated using the same terms with which they were extracted. The approach proposed herein is to take advantage of the fact that we are obtaining patterns for several relationships. Thus, the hook corpora for some of the patterns can be used also to identify errors done by other patterns.

The input of the system now is not just a list of related pairs, but a table including several relationships for the same entities. We may consider it as mini-biographies as in Mann and Yarowsky (2005)'s system. Table 1 shows a few rows in the input table for the system. The cells for which no data is provided have a default value of **None**, which means that anything extracted for that cell will be considered as incorrect.

Although this table can be written by hand, in our experiments we have chosen to build it automatically from the lists of related pairs. The system receives the seed pairs for the relationships, and mixes the information from all of them in a single table. In this way, if *Dickens* appears in the seed list for the *birth year*, *death year*, *birth place* and *writer-book* relationships, four of the cells in its row will be filled in with values, and all the rest will be set to **None**. This is probably a very strict evaluation, because, for all the cells for which there was no value in any of the lists, any result obtained will be judged as incorrect. However, the advantage is that we can study the behavior of the system working with incomplete data.

The new procedure for calculating the patterns' precisions is as follows:

1. For every relationship, and for every *hook*, collect a *hook corpus* from the Internet.
2. Apply the patterns to all the hook corpora collected. Whenever a pattern extracts a relationship from a sentence,
   - If the table does not contain a row for the hook, ignore the result.
   - If the extracted target appears in the corresponding cell in the table, consider it correct.
   - If that cell contained different values, or **None**, consider it incorrect.

For instance, the pattern *<target> 's <hook>* extracted for *director-film* may find, in the Dickens corpus, book titles. Because these titles do not appear in the table as films directed by Dickens, the pattern will be considered to have a low accuracy.

In this step, every pattern that did not apply at least three times in the test corpora was discarded.

4 Pattern application

Finally, given a set of patterns for a particular relation, the procedure for obtaining new pairs is straightforward:

1. For any of the patterns,
2. For each sentence in the test corpus,
   (a) Look for the left-hand-side context in the sentence.
   (b) Look for the middle context.
   (c) Look for the right-hand-side context.
   (d) Take the words in between, and check that either the sequence of part-of-speech tags or the entity type had been

Applied	Prec.	Pattern
3	1.0	BOS/BOS On/IN time_expr/entity TARGET HOOK was/VBD baptized|born/VBN EOS/EOS
15	1.0	"/'' HOOK (/(TARGET -/-
4	1.0	,/, TARGET ,/, */* Eugne|philosopher|playwright|poet/NNP HOOK earned|was/VBD */* at|in/IN
23	1.0	-|--/- HOOK (/(TARGET -/-
12	1.0	AND|and|or/CC HOOK (/(TARGET -/-
48	1.0	By|about|after|by|for|in|of|with/IN HOOK TARGET -/-
4	1.0	On|of|on/IN TARGET ,/, HOOK emigrated|faced|graduated|grew|has|perjured|settled|was/VBD
12	1.0	BOS/BOS HOOK TARGET -|--/-
49	1.0	ABOUT|ALFRED|Amy|Audre|Authors|BY| (...) |teacher|writer/NNPS HOOK (/(TARGET -|--/-
7	1.0	BOS/BOS HOOK (/(born/VBN TARGET)/)
3	1.0	BOS/BOS HOOK ,/, */* ,/, TARGET ,/,
13	1.0	BOS/BOS HOOK ,|:/, TARGET -/-
132	0.98	BOS/BOS HOOK (/(TARGET -|--/-
18	0.94	By|Of|about|as|between|by|for|from|of|on|with/IN HOOK (/(TARGET -/-
33	0.91	BOS/BOS HOOK ,|:/, */* (/(TARGET -|--/-
10	0.9	BOS/BOS HOOK ,|:/, */* ,|:/, TARGET -/-
3	0.67	,|:|;/, TARGET ,|:/, */* Birth|son/NN of/IN */* General|playwright/NNP HOOK ,|;/,
210	0.63	,|:|;/, HOOK (/(TARGET -|--/-
7	0.29	(/(HOOK TARGET)/)

Table 3: Patterns for the relationship *birth year*.

.

Relation	Seeds	Extr.	Gener.	Filt.
Actor-film	133	480	519	10
Writer-book	836	3858	4847	171
Birth-year	492	2520	3220	19
Birth-place	68	681	762	5
Country-capital	36	932	1075	161
Country-president	56	1260	1463	119
Death-year	492	2540	3219	16
Director-film	1530	3126	3668	121
Painter-picture	44	487	542	69
Player-team	110	2903	3514	195

Table 2: Number of seed pairs for each relation, and number of unique patterns after the extraction and the generalization step, and after calculating their accuracy and filtering those that did not apply 3 times on the test corpus.

Relation	Precision	Incl. prec.	Applied
Actor-film	0%	76.84%	95
Writer-book	6.25%	28.13%	32
Birth-year	79.67%	79.67%	477
Birth-place	14.56%	14.56%	103
Country-capital	72.43%	72.43%	599
Country-president	81.40%	81.40%	43
Death-year	96.71%	96.71%	152
Director-film	43.40%	84.91%	53
Painter-picture	-	-	0
Player-team	52.50%	52.50%	120

Table 4: Precision, *inclusion precision* and number of times that a pattern extracted information, when applied to a test corpus.

seen in the training corpus for that role (hook or target). If so, output the relationship.

5 Evaluation and results

The procedure has been tested with 10 different relationships. For each pair in each seed list, a corpus with 500 documents has been collected using Google, from which the patterns are extracted. Table 2 shows the number of patterns obtained. It is interesting to see that for some relations, such as birth-year or birth-place, more than one thousand patterns have been reduced to a few. Table 3 shows the patterns obtained for the relationship *birth-year*. It can also be seen that some of the patterns with good precision contain the wildcard *, which helped extract the correct birth year in roughly 50 occasions. Specially of interest is the last pattern,

(/(HOOK TARGET)/)

which resulted in an accuracy of 0.29 with the procedure here indicated, but which would have obtained an accuracy of 0.54 using the traditional *hook corpus* approach. This is because in other test corpora (e.g. in the one containing soccer players and clubs) it is more frequent to find the name of a person followed by a number that is not his/her birth year, while that did not happen so often in the *birth year* test corpus.

For evaluating the patterns, a new test corpus has been collected for fourteen entities not present in the training corpora, again using Google. The chosen entities are *Robert de Niro* and *Natalie Wood* (actors), *Isaac Asimov* and *Alfred Bester* (writers), *Montevideo* and *Yaounde* (capitals), *Gloria Macapagal Arroyo* and *Hosni Mubarak* (country presidents), *Bernardo Bertolucci* and *Federico Fellini* (directors), *Peter Paul Rubens* and *Paul Gauguin* (painters), and *Jens Lehmann* and *Thierry Henry* (soccer players). Table 4 shows the results obtained for each relationship.

We have observed that, for those relationships in which the target does not belong to a Named

Entity type, it is common for the patterns to extract additional words together with the right target. For example, rather than extracting *The Last Emperor*, the patterns may extract this title together with its rating or its length, the title between quotes, or phrases such as *The classic The Last Emperor*. In the second column in the table, we measured the percentage of times that a correct answer appears inside the extracted target, so these examples would be considered correct. We call this metric *inclusion precision*.

5.1 Comparison to related approaches

Although the above results are not comparable to Mann and Yarowsky (2005), as the corpora used are different, in most cases the precision is equal or higher to that reported there. On the other hand, we have rerun Ravichandran and Hovy (2002)'s algorithm on our corpus. In order to assure a fair comparison, their algorithm has been slightly modified so it also takes into account the part-of-speech sequences and entity types while extracting the hooks and the targets during the rule application. So, for instance, the relationship *birth date* is only extracted between a hook tagged as a person and a target tagged as either a date or a number. The results are shown in Table 5. As can be seen, our procedure seems to perform better for all of the relations except *birth place*. It is interesting to note that, as could be expected, for those targets for which there is no entity type defined (films, books and pictures), Ravichandran and Hovy (2002)'s extracts many errors, because it is not possible to apply the Named Entity Recognizer to clean up the results, and the accuracy remains below 10%. On the other hand, that trend does not seem to affect our system, which had very poor results for *painter-picture*, but reasonably good for *actor-film*.

Other interesting case is that of *birth places*. A manual observation of our generalized patterns shows that they often contain disjunctions of verbs such as that in (1), that detects not just the birth place but also places where the person lived. In this case, Ravichandran and Hovy (2002)'s patterns resulted more precise as they do not contain disjunctions or wildcards.

(1) `HOOK ,/, returned|travelled|born/VBN to|in/IN TARGET`

It is interesting that, among the three relationships with the smaller number of extracted patterns, one of them did not produce any result, and the two others attained a low precision. Therefore, it should be possible to improve the performance of the system if, while training, we augment the training corpora until the number of extracted patterns exceeds a given threshold.

Relation	Our approach	Ravichandran and Hovy's
Actor-film	76.84%	1.71%
Writer-book	28.13%	8.55%
Birth-year	79.67%	49.49%
Birth-place	14.56%	88.66%
Country-capital	72.43%	24.79%
Country-president	81.40%	16.13%
Death-year	96.71%	35.35%
Director-film	84.91%	1.01%
Painter-picture	-	0.85%
Player-team	52.50%	44.44%

Table 5: *Inclusion precision* on the same test corpus for our approach and Ravichandran and Hovy (2002)'s.

6 Related work

Extracting information using Machine Learning algorithms has received much attention since the nineties, mainly motivated by the Message Understanding Conferences (MUC6, 1995; MUC7, 1998). From the mid-nineties, there are systems that learn extraction patterns from partially annotated and unannotated data (Huffman, 1995; Riloff, 1996; Riloff and Schmelzenbach, 1998; Soderland, 1999).

Generalizing textual patterns (both manually and automatically) for the identification of relationships has been proposed since the early nineties (Hearst, 1992), and it has been applied to extending ontologies with hyperonymy and holonymy relationships (Kietz et al., 2000; Cimiano et al., 2004; Berland and Charniak, 1999), with overall precision varying between 0.39 and 0.68. Finkelstein-Landau and Morin (1999) learn patterns for company merging relationships with exceedingly good accuracies (between 0.72 and 0.93).

Rote extraction systems from the web have the advantage that the training corpora can be collected easily and automatically. Several similar approaches have been proposed (Brin, 1998; Agichtein and Gravano, 2000; Ravichandran and Hovy, 2002), with various applications: Question-Answering (Ravichandran and Hovy, 2002), multi-document Named Entity Coreference (Mann and Yarowsky, 2003), and generating

biographical information (Mann and Yarowsky, 2005).

7 Conclusions and future work

We have described here a new procedure for building a rote extractor from the web. Compared to other similar approaches, it addresses several issues: (a) it is able to generate generalized patterns containing wildcards; (b) it makes use of PoS and Named Entity tags during the generalization process; and (c) several relationships are learned and evaluated at the same time, in order to test each one on the test corpora built for the others. The results, measured in terms of precision and *inclusion precision* are very good in most of the cases.

Our system needs an input table, which may seem more complicated to compile that the list of related pairs used by previous approaches, but we have seen that the table can be built automatically from the lists, with no extra work. In any case, the time to build the table is significantly smaller than the time needed to write the extraction patterns manually.

Concerning future work, we are currently trying to improve the estimation of the patterns accuracy for the pruning step. We also plan to apply the obtained patterns in a system for automatically generating biographical knowledge bases from various web corpora.

References

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

M. Berland and E. Charniak. 1999. Finding parts in very large corpora. In *Proceedings of ACL-99*.

S. Brin. 1998. Extracting patterns and relations from the World Wide Web. In *Proceedings of the WebDB Workshop at the 6th International Conference on Extending Database Technology, EDBT'98*.

P. Cimiano, S. Handschuh, and S. Staab. 2004. Towards the self-annotating web. In *Proceedings of the 13th World Wide Web Conference*, pages 462–471.

M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. 1999. Learning to construct knowledge bases from the world wide web. *Artificial Intelligence*, 118(1–2):69–113.

M. Finkelstein-Landau and E. Morin. 1999. Extracting semantic relationships between terms: supervised vs. unsupervised methods. In *Workshop on Ontological Engineering on the Global Info. Infrastructure*.

M. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING-92*.

S. Huffman. 1995. Learning information extraction patterns from examples. In *IJCAI-95 Workshop on New Approaches to Learning for NLP*.

J. Kietz, A. Maedche, and R. Volz. 2000. A method for semi-automatic ontology acquisition from a corporate intranet. In *Workshop "Ontologies and text"*.

G. S. Mann and D. Yarowsky. 2003. Unsupervised personal name disambiguation. In *CoNLL-2003*.

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

M. Marcus, B. Santorini, and M.A. Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

MUC6. 1995. *Proceedings of the 6^{th} Message Understanding Conference (MUC-6)*. Morgan Kaufman.

MUC7. 1998. *Proceedings of the 7^{th} Message Understanding Conference (MUC-7)*. Morgan Kaufman.

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

E. Riloff and M. Schmelzenbach. 1998. An empirical approach to conceptual case frame acquisition. In *Proceedings of WVLC*, pages 49–56.

E. Riloff. 1996. Automatically generating extraction patterns from untagged text. In *AAAI*.

M. Ruiz-Casado, E. Alfonseca, and P. Castells. in press. Automatising the learning of lexical patterns: an application to the enrichment of wordnet by extracting semantic relationships from wikipedia. *Data and Knowledge Engineering*.

S. Soderland. 1999. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34(1–3):233–272.

R. Wagner and M. Fischer. 1974. The string-to-string correction problem. *Journal of Association for Computing Machinery*, 21.