Learning sure-fire rules for Named Entity Recognition

Enrique Alfonseca and Maria Ruiz-Casado

Department of Computer Science
Universidad Autónoma de Madrid
28049 Madrid
{Enrique.Alfonseca,Maria.Ruiz}@uam.es

Abstract

This paper describes a procedure for obtaining and generalising, automatically, patterns that can be used as *sure-fire* rules, i.e. rules that have a very high precision (albeit, possibly, low recall) for Named Entity Recognition. The experiments performed on the CoNLL-2003 training and test corpora show that, for people and locations, the patterns obtained attain very high precisions just by themselves. The precisions obtained for organisations and miscellaneous entities are somewhat lower, a fact which indicates that they should not be used without the aid of auxiliary gazetteers listing common entities belonging to those two classes.

1 Introduction

Named Entity (NE) Recognition is usually defined as the task of identifying and annotating instances of particular Named Entity categories, such as people, organisations or locations, inside unrestricted text. It was originally defined as a subtask of Information Extraction (IE) in the Message Understanding Conferences (MUC)[1], competitions in which most of the early research in NE recognition was accomplished (MUC6 95; MUC7 98). Since then, there have appeared many new applications of NE recognition, including Text Summarisation, Question Answering, Ontology Population with instances, and semantic annotation for ontology-based search engines.

Therefore, the interest on improving the current technology has not decreased. The MUC competitions have been followed by the CoNLL-2002[2] and CoNLL-2003[3] conferences (Tjong-Kim-Sang & Meulder 03), which addressed NE recognition from texts written in Spanish, Dutch, English and German; and the Automatic Content Extraction (ACE) program[4], which includes tasks on Entity Detection and Tracking across documents, and Time Expressions Recognition and Normalisation.

We can distinguish three types of systems for NE Recognition:

- Knowledge-based systems, which are based in the use of rules, patterns or grammars (Califf 98; Soderland 99; Freitag 00; Maynard *et al.* 02; Arevalo *et al.* 04). These rules are generally handcrafted, so there exist pattern languages, such as JAPE (Cunningham *et al.* 02), that simplify the task of writing the rules and the parsers.
- Those that apply Machine Learning techniques, either alone or in combination, including Memory-Based Learning, Maximum Entropy models and Hidden Markov Models (Freitag & McCallum 00; Klein *et al.* 03; Florian *et al.* 03; Kozareva *et al.* 05), Error-Driven Transformation-Based Learning (Black & Vasilakopoulos 02), boosting algorithms (Carreras *et al.* 03), and Support Vector Machines (Isozaki & Kazawa 02; Mayfield *et al.* 03; Li *et al.* 05).
- Those that combine knowledge-based methods and ML techniques (Mikheev *et al.* 98; Mikheev *et al.* 99).

In the MUC-7 competition (MUC7 98), the best results were obtained by (Mikheev *et al.* 98). It attained a precision of 93.39%, close to the 96.95% obtained by the worst human annotator. It can be considered a hybrid system with several stages. In the first one, the words in the texts were looked up in gazetteers (lists of common place, people and organisation names). Every time a candidate entity was found in a list, the system applied rules with a very high precision (they call *sure-fire rules*) before annotating it. For instance, the following rules

```
Xxxxx+ is a?  JJ* PROF
   shares of Xxxxx+
      Xxxxx+ area
```

indicate that one or more consecutive capitalised words can be considered as a person, an organisation or a location if they appear in the place of `Xxxxx+` in these three patterns, respectively. If the system, for example, found the word *Washington*, which may appear in the gazetteers for people and locations, it checked whether the context matched any of the rules for people and for places. If it is seen in the phrase *in the Washington area*, then the sure-fire rule for locations triggers, and it is marked up. After the sure-fire rules had applied on the corpus, the system continues with a maximum-entropy model.

In summary, we can consider that there are some contexts which strongly indicate the presence of a particular Named Entity. Therefore, the approach of using these *sure-fire rules* to annotate entities before

[1] http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_toc.html
[2] http://www.cnts.ua.ac.be/conll2002/ner/
[3] http://www.cnts.ua.ac.be/conll2003/ner/
[4] http://www.nist.gov/speech/tests/ace/

executing a second step, possibly based on Machine Learning techniques, seems very sound. However, a disadvantage of the pattern-based annotation is that the rules are generally designed manually and, thus, they should be difficult to port to new domains and to new kinds of entities.

This work addresses that problem by proposing a new algorithm for automatically extracting and generalising the sure-fire rules when a training corpus is available. The objective is to find a set of high-precision patterns with which a few Named Entities inside a corpus can be annotated, so the corpus provided to the ML techniques already contains some entities identified and classified inside it. This new approach will be incorporated soon to the *wraetlic* tools[5].

This paper is structured as follows: first, Section 2 describes the procedure used for extracting and generalising the sure-fire rules; next Section 3 describes the evaluation performed and the results obtained. Finally, Section 4 summarises the conclusions and describes open lines for future work.

2 Procedure

The general procedure to obtain the *sure-fire* rules is the following: first of all, for every appearance of a kind of entity (e.g. people) in the training corpus, we extract its context window. Next, in a second step, we generalise those contexts to obtain the patterns shared by them. The following sections describe these steps in detail.

2.1 Pattern extraction

The purpose of the first step is to extract a set of very specific context patterns obtained from the training corpus. We have worked with the CoNLL-2003 English dataset, which was built from the REUTERS corpus. In this dataset, all the words are annotated with part-of-speech tags. The sentences are chunked in noun, verb, prepositional and adverbial phrases, and four kinds of entities are annotated: person (`PER`), organisation (`ORG`), location (`LOC`), and miscellaneous names (`MISC`).

In this step, we simply look for all the instances of each of the four entity types, and extract a context around them. The context we have used is a window that includes up to five words to the left of the entity, and five words to its right. The window never jumps over sentence boundaries. For this experiment, we have only considered the words and their p-o-s tags; chunking information is discarded. Figure 1 shows several example contexts extracted for each of the four kinds of entities. The abbreviations `-BOS-` and `-EOS-` mark the beginning-of-sentence and the end-of-sentence, respectively.

Furthermore, for each entity type, we also collect, from the training corpora, the sequences of part-of-speech tags of every known entity. So, for instance, `NNP NNP` is a common sequence for people, as they are usually represented with two proper names (first name and family name), and `NNP CC NNP` will be a common sequence for organisations, as they sometimes include conjunctions in their names.

[5] http://www.ii.uam.es/~ealfon/eng/research/wraetlic.html

2.2 Pattern generalisation (I): Algorithm

The previous patterns might be directly applied on the test corpus to annotate entities inside it. However, for the moment, most of them are far too specific and the probability that we will find exactly the same 10-word context in a new text is very small. Therefore, we would like to substitute the patterns that have commonalities with more general patterns with a larger coverage. The following algorithm is used in order to generate the final set of generalised patterns:

1. Store all the patterns in a set $\mathcal{P}$.
2. Initialise a set $\mathcal{D}$ 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 the next section).
   (b) Take the two patterns with the smallest distance, p_i and p_j.
   (c) Remove them from $\mathcal{P}$.
   (d) Obtain the generalisation of both, p_g.
   (e) If the precision of p_g in the training corpus is over a threshold θ, add p_g to $\mathcal{P}$.
   Otherwise, add p_1 and p_2 to $\mathcal{D}$.
4. Return $\mathcal{D}$

The previous algorithm is repeated, separately, to generalise the patterns for people, locations, organisations and miscellaneous names. The output is the set containing all the rules that have been obtained by combining pairs of original rules. The purpose of the parameter θ is to ensure that we do not generalise patterns that are too different, resulting in rules that match in many places in the texts with a low precision. If we set θ to a high value, say 0.9, we are guiding the search towards high-precision rules; on the other hand, if we set it to a lower value, e.g. 0.75, then we'll obtain a smaller set of rules, which will have a larger coverage but a lower precision.

The next sections describes how the distance between the patterns is calculated for step (3a), and the procedure to generalise them in step (3d).

2.3 Pattern generalisation (II): Edit distance calculation

In order to generalise two patterns, the general idea is to look for the similarities between them, and to remove all those things that they do not have in common.

The procedure used to obtain a similarity metric between two patterns, consists of a slightly modified version of the dynamic programming algorithm for *edit-distance* calculation (Wagner & Fischer 74). This procedure has already been used successfully to extract

Person:

by/IN Hendrix/NNP 's/POS former/JJ girlfriend/NN ENTITY ,/, who/WP lived/VBD with/IN him/PRP
green/JJ light/NN to/TO Prime/NNP Minister/NNP ENTITY to/TO call/VB snap/VB elections/NNS ,/,
elections/NNS ,/, its/PRP$ general/JJ secretary/NN ENTITY told/VBD reporters/NNS ./. -EOS-/-EOS-
-BOS-/-BOS- ENTITY ,/, who/WP as/IN Israel/NNP 's/POS
-BOS-/-BOS- ENTITY is/VBZ winding/VBG up/RP his/PRP$ term/NN
He/PRP will/MD be/VB replaced/VBN by/IN ENTITY ,/, a/DT former/JJ Israeli/JJ envoy/NN

Location:

China/NNP 's/POS top/JJ negotiator/NN with/IN ENTITY ,/, Tang/NNP Shubei/NNP ,/, as/IN
-BOS-/-BOS- ENTITY accused/VBD Israel/NNP on/IN Wednesday/NNP of/IN
Iraqi/JJ forces/NNS were/VBD ousted/VBN from/IN ENTITY in/IN the/DT 1991/CD Gulf/NNP War/NNP
positions/NNS in/IN Qasri/NNP region/NN in/IN ENTITY province/NN near/IN the/DT Iranian/JJ border/NN
expected/VBN to/TO travel/VB to/TO the/DT ENTITY before/IN Monday/NNP ,/, "/" Nabil/NNP
minister/NN Shimon/NNP Peres/NNP in/IN the/DT ENTITY town/NN of/IN Ramallah/NNP on/IN Thursday/NNP

Organisation:

the/DT talks/NNS ,/, the/DT official/NN ENTITY news/NN agency/NN quoted/VBN Tang/NNP Shubei/NNP
executive/JJ vice/NN chairman/NN of/IN the/DT ENTITY ,/, as/IN saying/VBG late/RB on/RB
the/DT year-earlier/JJ period/NN ,/, the/DT ENTITY said/VBD on/IN Thursday/NNP ./. -EOS-/-EOS-
-BOS-/-BOS- ENTITY won/VBD 77,719/CD registrations/NNS ,/, slightly/RB
-BOS-/-BOS- Third/JJ was/VBD ENTITY with/IN 35,563/CD registrations/NNS ,/, or/CC

Miscellaneous:

-BOS-/-BOS- ENTITY farmers/NNS denied/VBN on/IN Thursday/NNP there/EX
,/, but/CC expressed/VBD concern/NN that/IN ENTITY government/NN advice/NN to/TO consumers/NNS to/TO
to/TO Ukraine/NNP this/DT week/NN by/IN ENTITY Vice/NNP President/NNP Lien/NNP ./. -EOS-/-EOS-
-BOS-/-BOS- ENTITY July/NNP car/NN registrations/NNS up/RB 14.2/CD

Figure 1: Example patterns extracted from the training corpus for each kind of entity.

patterns for identifying hyperonymy and meronymy relationships inside text (Ruiz-Casado *et al.* 05). The *edit distance* between two strings A and B is defined as the minimum number of changes (character insertion, addition or replacement) that have to be done to the first string in order to obtain the second one. The algorithm can be implemented as filling in a matrix $\mathcal{M}$ with the following procedure:

$$\mathcal{M}[0,0] = 0 \tag{1a}$$

$$\mathcal{M}[i,0] = \mathcal{M}[\ i-1,0] + 1 \tag{1b}$$

$$\mathcal{M}[0,j] = \mathcal{M}[0,j-1] + 1 \tag{1c}$$

$$\begin{aligned}\mathcal{M}[i,j] = min(&\mathcal{M}[i-1,j-1] + d(A[i],B[j]),\\ &\mathcal{M}[i-1,j] + 1,\\ &\mathcal{M}[i,j-1] + 1)\end{aligned} \tag{1d}$$

where $i\ [1...|A|], j\ [1...|B|]$
and

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

In these equations, $\mathcal{M}$[i,j] will contain the edit distance between the first i elements of A and the first j elements of B. Equation (1a) indicates that, if A and B are both empty strings, the edit distance should be 0. Equations (1b) and (1c) mean that the edit distance between an empty string, and a string with N symbols must be N. Finally, equation (1d) uses the fact that, in order to obtain a string[6] $A\sigma$ from a string $B\gamma$, we may proceed in three possible ways:

- We may obtain $A\gamma$ from $B\gamma$, and next substitute γ by σ. If γ and σ are the same, no edition will be required.
- We may obtain $A\sigma\gamma$ from $B\gamma$, and next delete γ at the end.
- We may obtain A from $B\gamma$, and next insert the symbol σ in the end.

In the end, the value at the rightmost lower position of the matrix is the edit distance between both strings. The same algorithm can be implemented for word patterns, if we consider that the basic element of each pattern is not a character but a whole token.

At the same time, while filling matrix $\mathcal{M}$, it is possible to fill in another matrix $\mathcal{D}$, in which we record which of the choices was selected as minimum in equation (1d). This can be used afterwards in order to have in mind which were the characters that both strings had in common, and in which places it was necessary to add, remove or replace characters. We have used the following four characters:

- `I` means that it is necessary to insert a token, in order to transform the first string into the second one.
- `R` means that it is necessary to remove a token.
- `E` means that the corresponding tokens are equal, so it is not necessary to edit them.
- `U` means that the corresponding tokens are unequal, so it is necessary to replace one by the other.

Figure 2 shows an example for two patterns, A and B, containing respectively 5 and 4 tokens. The first row and the first column in $\mathcal{M}$ would be filled during the initialisation, using Formulae (1b) and (1c). The corresponding cells in matrix $\mathcal{D}$ are filled in the following way: the first row is all filled with `I`'s, indicating that it is necessary to insert tokens to transform an empty string into B; and the first column is all filled with `R`'s indicating that it is necessary to remove tokens to transform A into an empty string. Next, the remaining cells would be filled by the algorithm, looking, at each step, which is the choice that minimises the edit distance. $\mathcal{M}(5,4)$ has the value 2, indicating the distance between the two complete patterns. For instance, the two editions would be:

- Replacing `a` by `nice`.

[6] $A\sigma$ represents the concatenation of string A with character σ.

A: It is a kind of
B: It is nice of

$\mathcal{M}$	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 $\mathcal{M}$.

- Removing `kind`.

2.4 Pattern generalisation (III): Algorithm

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

1. Initialise the generalised pattern G as the empty string.
2. Start at the last cell of the matrix $\mathcal{M}(i, j)$. In the example, it would be $\mathcal{M}(5, 4)$.
3. While we have not arrived to $\mathcal{M}(0, 0)$,
   (a) If ($\mathcal{D}(i, j) =$ `E`), then the two patterns contained the same token A[i]=B[j].
   - Set $G = A[i]\ G$
   - Decrement both i and j.

   (b) If ($\mathcal{D}(i, j) =$ `U`), then the two patterns contained a different token.
   - $G = A[i]|B[j]\ G$, where | represents a disjunction of both terms.
   - Decrement both i and j.

   (c) If ($\mathcal{D}(i, j) =$ `R`), then the first pattern contained tokens not present in the other.
   - Set $G = *\ G$, where $*$ represents any sequence of terms.
   - Decrement i.

   (d) If ($\mathcal{D}(i, j) =$ `I`), then the second pattern contained tokens not present in the other.
   - Set $G = *\ G$
   - Decrement j

If the algorithm is followed, the patterns in the example will produced the generalised pattern

```
It is a kind   of
It is nice     of
-----------------
It is a|nice * of
```

This pattern may match phrases such as *It is a kind of*, *It is nice of*, *It is a subset of*, or *It is a type of*. As can be seen, the generalisation of these two rules produces one that can match a wide variety of sentences, so we should always take care in order not to over-generalise.

2.5 Pattern generalisation (IV): Generalisation with part-of-speech tags

As shown in the previous example, sometimes, when two patterns are combined, the result is too general and matches more contexts than expected. Part-of-speech tags have been used to modify the edit distance calculation, in a way such that the edit distance of two patterns which do not differ in their sequences of part-of-speech tags will remain small even though their words are all different.

Our patterns are, therefore, sequences of terms annotated with part-of-speech labels, as in the following examples:

(a) `It/PRP is/VBZ a/DT kind/NN of/IN`
(b) `It/PRP is/VBZ nice/JJ of/IN`
(c) `It/PRP is/VBZ the/DT type/NN of/IN`

The calculation is modified in the following way: the system only allows replacement actions if the words from the two patterns A and B belong to the same general part-of-speech (nouns, verbs, adjectives, adverbs, etc.). Also, if this is the case, we consider that there is no edit distance between the two patterns. The d function, therefore, is redefined as:

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

The insertion and deletion actions are defined as before. Therefore, patterns (a) and (b) above would have an edit distance of 3: two deletions, `a` and `kind`, and one insertion, `nice`. Note that it is not possible to do any replacement, because those words have different p-o-s tags. The result of their generalisation is:

`It/PRP is/VBZ * of/IN`

On the other hand, the patterns (a) and (c) would have an edit distance of 0, and the result of their generalisation would be the following:

`It/PRP is/VBZ a|the/DT kind|type/NN of/IN`

2.6 Application of the generalised patterns for NE recognition

Finally, given a set of patterns for a particular named entity, the procedure for annotating is straightforward:

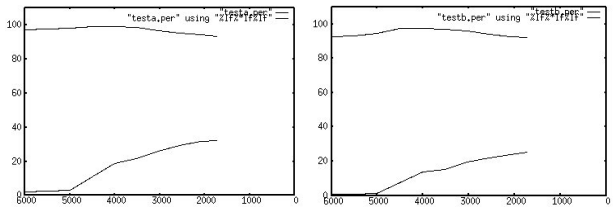

Figure 3: Recall and precision identifying people on test sets A and B, depending on the number of rules during the generalisation.

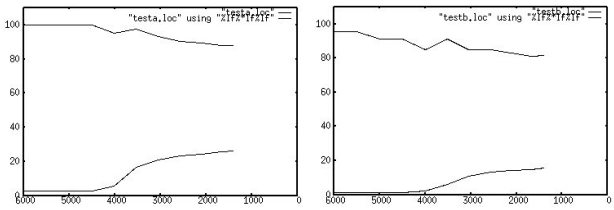

Figure 4: Recall and precision identifying locations on test sets A and B, depending on the number of rules during the generalisation.

1. For any of the rules in the set $\mathcal{D}$, defined for a particular entity type,
2. For each sentence in the corpus:
   (a) Look for the left-hand side of the rule in the sentence.
   (b) Look for the right-hand side of the rule afterwards in the sentence.
   (c) Take the words that are in between. If the sequence of part-of-speech tags has been seen in the training corpus for that kind of entity, annotate it.

For instance, the following pattern

```
,/, ENTITY announced|said/VBD */* I|We|he|it/PRP
came|did|have|think|wanted/VBD
```

matches with the sentence *Today, John Smith announced Mary he wanted a car*. First of all, it is necessary to find, in a sentence, a comma. Later on, the program finds that *announced Mary he wanted* matches the last part of the pattern. In between we find the words *John Smith*, with p-o-s tags `NNP NNP`, which is valid for people according to the training corpus. Therefore, *John Smith* will be annotated as a person inside that sentence.

3 Evaluation and results

We have tried the rules on the CoNLL evaluation data in several experiments:

- Applying the whole set of rules obtained with the rule generalisation procedure.
- Applying a pruned set of rules that only included those that applied at least a certain number of times on the training data.
- Applying the pruned set of rules, and adding a simple heuristic about a few words that were clearly mis-tagged.

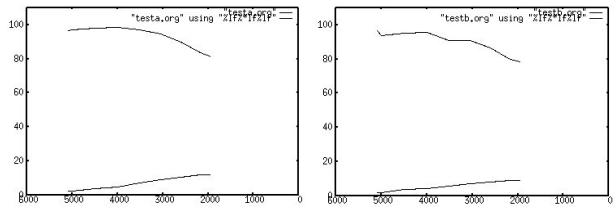

Figure 5: Recall and precision identifying organisations on test sets A and B, depending on the number of rules during the generalisation.

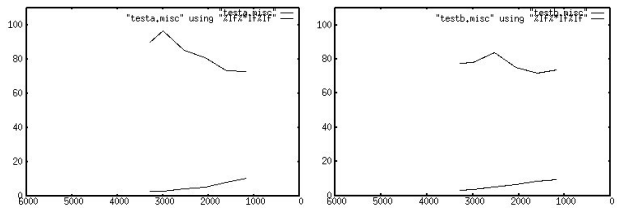

Figure 6: Recall and precision identifying miscellaneous entities on test sets A and B, depending on the number of rules during the generalisation.

The results are described in the following sections.

Direct application of the rules After extracting the patterns from the CoNLL training data, they have been generalised using the previous algorithm. We have set the threshold θ to 0.9, because we want to obtain patterns with high precision. In this way, we can be sure that all the rules obtained at the end have a precision of at least 0.9 when applied on the training corpus.

The CoNLL competition provided two different test corpora, named A and B. Figures 3, 4, 5, and 6 show how the precision and the recall of the set of patterns varies as we generalise them.

For instance, in the case of people, the system starts with roughly 6000 very specific patterns, which, applied on test set A, have a very high precision (96.88%), but a very low recall (1.68%). Note that the precision is not 100%, i.e. a few patterns may make mistakes. This is because not all the patterns contain a 10-word window if the Named Entity was very near the sentence boundaries. In the extreme case, an extracted pattern might be

```
-BOS-/-BOS- ENTITY -EOS-/-EOS-
```

which will surely have a low precision on any corpus.

As the patterns are being generalised by twos, the total number of patterns in the set decreases. It can be seen that precision drops slightly down to 93.5%, and recall increases up to 32.19%. With test set B, the performance is rather similar.

The case of locations is also alike to that of people. In test set A, when the system finishes generalising the patterns, the resulting set of rules attains a precision of 87.8% and a recall of 26.35%. In this case as well we can consider that the precision of the rules has been preserved from the training set to the test sets, as it is very near the value of 90% that we had set as threshold

for the generalisation. The results for test set B are slightly worse, 81.5% precision and 15.9% recall.

The results obtained by the rules for organisations also behave in the same way: before generalising, precision is near 95% and recall is very small; and, as we proceed, recall improves and precision decreases. With the final set, the precisions obtained are 81.4% on set A, and 78.3% on set B; and the recall is 11.8% and 8.9%, respectively.

Finally, the patterns for the miscellaneous entities have proven the most difficult to learn. This is the only case in which the most specific patterns, without any generalisation, do not have a very high precision. For instance, in test B, the original patterns obtained from the training corpus just attain a precision of 77.78%. After they have been generalised, precision drops to 73%, and recall reaches 10.5%. This set is particularly difficult because it does not contain a very precise kind of entities; it includes things such as names of the inhabitants of countries (e.g. German) or illnesses (e.g. Bovine Spongiform Encephalopathy).

Application of the pruned set of rules In a second experiment, we have pruned the obtained set of rules to remove those that applied just a few number of times in the training corpus. The motivation for this experiment is that, if a rule applies many times in the training corpus with a very high precision, then we can be confident that it will continue having a high precision in the test corpus; whereas if the rule has only applied once in the training corpus, we have much less evidence to assert that it will extract correct entities in other corpus.

We have varied the threshold for discarding a rule from 1 to 8. In this way, in the first run, we keep all the patterns that applied more than once in the training corpus; in the second run, those that applied more than twice; and so on. The results can be seen in Table 1, columns 5 and 6. The pruning increases the precision for all entities, at the cost of a small decrease in recall.

Application with a heuristic Finally, a manual observation of the results allowed us to identify the most common errors for people and for locations. Firstly, the obtained patterns may extract mistakenly people's titles together with their names. For instance, the pattern

`17-year-old/JJ ENTITY and|or/CC`

may extract, from the sentence

`17-year-old Secretary John Smith and`

the entity `Secretary John Smith`, while it will be tagged in the test set as just `John Smith`. Secondly, in the case of locations, we found that some patterns for locations were able to extract weekday and month names as well. Therefore, in this last run, we post-processed the corpus to remove all the people titles (found in a list) and weekday and month names. The results for test set A are shown in Table 1. It

				No heuristic		Heuristic	
Type	**T**	**Patterns**		**Prec.**	**Recall**	**Prec.**	**Recall**
PER	0	1720	1720	93.53	32.19	95.40	32.63
	1	1720	391	94.20	29.10	95.41	29.37
	2	1720	262	94.46	28.72	95.52	28.94
	3	1720	199	95.13	28.61	96.20	28.83
	4	1720	157	95.63	28.50	96.71	28.72
	5	1720	120	95.77	28.28	96.86	28.50
	6	1720	102	96.12	28.23	97.22	28.45
	7	1720	93	96.46	28.07	97.57	28.28
	8	1720	81	96.64	28.07	**97.75**	28.28
LOC	0	1387	1387	87.84	26.35	90.98	26.35
	1	1387	382	93.43	20.14	95.36	20.14
	2	1387	257	93.15	19.98	95.08	19.98
	3	1387	183	93.49	19.54	95.48	19.54
	4	1387	146	93.72	19.49	95.72	19.49
	5	1387	115	93.93	19.38	95.96	19.38
	6	1387	89	93.90	19.27	95.93	19.27
	7	1387	76	94.10	19.11	95.90	19.11
	8	1387	67	94.57	18.94	**96.13**	18.94
ORG	0	1943	1943	81.44	11.78	81.96	11.86
	1	1943	425	80.00	8.95	80.67	9.02
	2	1943	266	80.42	8.58	81.12	8.65
	3	1943	191	80.77	7.83	81.54	7.90
	4	1943	146	80.00	7.16	80.83	7.23
	5	1943	120	80.83	7.23	81.67	7.31
	6	1943	102	83.64	6.86	84.55	6.94
	7	1943	82	86.41	6.64	87.38	6.71
	8	1943	70	88.12	6.64	89.11	6.71
MISC	0	1166	1166	72.93	10.52	72.93	10.52
	1	1166	271	78.21	6.62	78.21	6.62
	2	1166	184	78.21	6.62	78.21	6.62
	3	1166	141	78.08	6.18	78.08	6.18
	4	1166	111	79.41	5.86	79.41	5.86
	5	1166	89	85.48	5.75	85.48	5.75
	6	1166	70	85.25	5.64	85.25	5.64
	7	1166	53	84.48	5.31	84.48	5.31
	8	1166	48	87.27	5.21	87.27	5.21

Table 1: Results after pruning the sets of rules, on test set A. Columns indicate entity type, the threshold (T) for the pruning, and the precision and the recalled obtained with the pruned set of rules with or without the heuristic. T=0 means no pruning.

can be seen that the precision of the rules for people and locations are somewhat improved, and become similar to Mikheev's results (99% for people and 96% for locations after just the sure-fire rules; 97% for people and 93% for locations after the complete run of his system). In a few cases, the precision for organisations changes slightly, due to the fact that something that has been untagged as a person or a location may next be tagged as an organisation if a pattern for ORG matches with its context.

The final results for test set B are similar: for people, the precision is 97.07%. In the case of locations, the precision obtained was worse (88.70%), due to a single pattern that classified all football teams as locations. The removal of that pattern boosts the precision for locations up to 96.88%.

4 Conclusions and future work

This paper describes a procedure for obtaining from a corpus, and next generalising, automatically, patterns that can be used as *sure-fire* rules in an Information Extraction system, in a similar way as the one described in (Mikheev *et al.* 98). The system obtains the generalised patterns for one entity in ∼3 hours in a Pentium 2.4GHz, and the test corpora can be tagged in less than one minute.

We have shown that, in our case, the patterns learnt for people and for locations have a very high precision, combined with a simple heuristic, even without the use

of gazetteers of people and location names. (Mikheev *et al.* 99) quantifies the precision of a sure-fire rule in the rage 96-98%, so these fall inside the interval. In the case of organisations and miscellaneous entities, precision is just 87-89%, so it might be better to use them combined with gazetteers to increase their precisions. At this point of the work, we may only recommend to use the system for people and locations, as the patterns for the other entities should be improved a little more. On the other hand, the recall is low, probably due to a sparse-data problem. We believe that the training corpus includes a minimal fraction of all the possible contexts in which an entity can appear.

Concerning these results, it should be noted that (Mikheev *et al.* 98) just applied the *sure-fire* rules when the entity that matched the rule *also* appeared in a gazetteer. (Mikheev *et al.* 99) reports that the precision decreases somewhat if a small gazetteer is used, and may decrease very much (in the case of locations) if no gazetteer is present. In our case, even without using a gazetteer, precision is very high for people and locations, and rather high for organisations. We expect that, combined in this way with lists of people, companies and place names, the precision of the rules will be even higher than the one obtained.

For future work, we would like to extend the generalisation procedure, to see if both the precision and recall of the rules can be further improved if a more expressive encoding is used. Along this line, we plan (a) to be able to substitute disjunctions of words with the same part-of-speech, such as `big|small|large/JJ` by any word of that p-o-s, `?/JJ`, something that is currently not implemented; (b) to extend the generalisation with semantic classes, so patterns such as

```
The/DT Spain|France|Italy|Japan/NNP Minister/NNP
```

can be substituted by

```
The/DT {country:1}/NNP Minister/NNP
```

using the hyperonymy relationship in WordNet, in a similar way as in (Soderland 99); (c) to test the system in unsupervised settings. From a set of seed words that we know to pertain to a given Named Entity class (e.g. person names or locations names), we could learn and generalise the patterns and, using a bootstrapping procedure, apply those patterns to augment the set of seed words from which to extend the set of patterns; (d) to test the effect of gazetteers on the accuracy of the learnt rules; and (e) to test the influence of this system as an initial step in a complete system.

5 Acknowledgements

This paper has been funded by CICYT, project numbers TIC2002-01948 and TIN2004-03140. We wish to thank the anonymous reviewers for their comments, which have been very useful for improving this paper.

References

(Arevalo *et al.* 04) M. Arevalo, M. Civit, and M. A. Martí. MICE: A module for named entity recognition and clasification. *International Journal of Corpus Linguistics*, 9(1):53–68, 2004.

(Black & Vasilakopoulos 02) W. J. Black and A. Vasilakopoulos. Language-independent named entity classification by modified transformation-based learning and by decision tree induction. In *Proceedings of CoNLL-2002*, pages 159–162, Taipei, Taiwan, 2002.

(Cali 98) Mary Elaine Cali. *Relational Learning Techniques for Natural Language Extraction*. PhD thesis, University of Texas at Austin, 1998.

(Carreras *et al.* 03) X. Carreras, L. Márquez, and L. Padró. A simple named entity extractor using adaboost. In *Proceedings of CoNLL-2003*, pages 152–155, Edmonton, Canada, 2003.

(Cunningham *et al.* 02) H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, and C. Ursu. The GATE user guide, 2002.

(Florian *et al.* 03) R. Florian, A. Ittycheriah, H. Jing, and T. Zhang. Named entity recognition through classifier combination. In *Proceedings of CoNLL-2003*, pages 168–171, Edmonton, Canada, 2003.

(Freitag & McCallum 00) Dayne Freitag and Andrew McCallum. Information extraction with HMM structures learned by stochastic optimization. In *AAAI/IAAI*, pages 584–589, 2000.

(Freitag 00) Dayne Freitag. Machine learning for information extraction in informal domains. *Machine Learning*, 39(2-3):169–202, 2000.

(Isozaki & Kazawa 02) Hideki Isozaki and Hideto Kazawa. E cient support vector classifiers for named entity recognition. In *Proceedings of the 19th international conference on Computational Linguistics*, pages 1–7, Morristown, NJ, USA, 2002. Association for Computational Linguistics.

(Klein *et al.* 03) D. Klein, J. Smarr, H. Nguyen, and C. Manning. Named entity recognition with character-level models. In *Proceedings of CoNLL-2003*, pages 180–183, Edmonton, Canada, 2003.

(Kozareva *et al.* 05) Z. Kozareva, O. Ferrández, and A. Montoyo. Combining data-driven systems for improving named entity recognition. In *Natural Language Processing and Information Systems*, volume 3513 of *Lecture Notes in Computer Science*, pages 80–90. Springer, 2005.

(Li *et al.* 05) Y. Li, K. Bontcheva, and H. Cunningham. Using uneven margins SVM and perceptron for information extraction. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 72–79, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

(Mayfield *et al.* 03) James Mayfield, Paul McNamee, and Christine Piatko. Named entity recognition using hundreds of thousands of features. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 184–187. Edmonton, Canada, 2003.

(Maynard *et al.* 02) D. Maynard, H. Cunningham, K. Bontcheva, and M. Dimitrov. Adapting a robust multi-genre ne system for automatic content extraction. In *Artificial Intelligence: Methodology, Systems, and Applications*, volume 2443 of *Lecture Notes in Artificial Intelligence*, pages 264–273. Springer-Verlag, 2002.

(Mikheev *et al.* 98) A. Mikheev, C. Grover, and M. [illegible]