ESSENCE: a Portable Methodology for Acquiring Information Extraction Patterns

Neus Català[1], **Núria Castell**[1] and **Mario Martin**[2]

Abstract. One important issue when constructing Information Extraction systems is how to obtain the knowledge needed for identifying relevant information in a document. In most approaches to this issue, the human expert intervention is necessary in many steps of the acquisition process. In this paper we describe ESSENCE, a new methodology that reduces significantly the need for human intervention. It is based on ELA, a new algorithm for acquiring information extraction patterns.

The distinctive features of ESSENCE and ELA are that 1) allow to automatically acquire IE patterns from unrestricted text corpus representative of the domain, due to 2) the ability of identifying surrounding context regularities for semantically relevant concept-words for the IE task by using non domain specific lexical knowledge tools and semantic relations from WordNet, and 3) restricting the human intervention to only the definition of the task and the validation and typification of the set of IE patterns obtained.

The use of a general purpose ontology and syntactic tools of general application allows the easy portability of the methodology and reduces the expert effort. Results of the application of this methodology for acquiring extraction patterns in a MUC-like task are also shown.

1 INTRODUCTION

Information Extraction (IE) is a Natural Language Processing (NLP) task whose goal is to extract a prespecified kind of information from a document. In the tradition of the *Message Understanding Conferences* (MUC) [7], an *IE system*: 1) identifies and 2) extracts specific information located in non-structured textual data, and 3) generates the output as has been requested. IE systems are domain specific because they extract particular entities or events from a particular domain skipping over the irrelevant ones. The kind of information to extract consists in a prespecified set of entities and their attributes, as well as relationships and events relating those entities. For instance, in the aircraft crashes domain that we show in section 4, an IE system must extract information about the location and the date of the crash, the number of victims and the aircraft involved in. This information is usually represented in the form of templates which slots must be filled. Our work shares this notion of IE.

IE is usually compared with the more widely known area of Information Retrieval (IR). Whereas Information Retrieval systems, given a keyword list, return a set of relevant documents that contain them, IE systems return the required information in a predetermined format [2, 8].

A common way to extract the desired information is by using *IE patterns* (also known as *extraction rules* or *conceptual patterns*) that consist in a set of lexical, syntactic and semantic constraints that a piece of text must satisfy in order to extract information from it, along with an indication of which information must be extracted.

2 IE PATTERNS ACQUISITION

When building an IE system, the task of acquiring information extraction patterns has to be faced. The procedures proposed for this task must reduce as much as possible the time cost of manual effort. In the last years, different approaches have been proposed in order to semi-automatize this task. Some IE systems have tried to include Machine Learning (ML) components intended to easy the move of the system to a new domain or to a new task definition.

AutoSlog [9] generates *concept node* definitions from the information in the answer keys (manually filled output templates)[3] of training texts. A concept node is activated by a keyword when it appears in a certain linguistic context, and it is able to infer the role played by the targeted information in this context. CRYSTAL [12] system inductively generates a dictionary of conceptual patterns (another name for IE patterns) that covers all the examples of the preprocessed[4] training texts. In addition of the preprocessed training texts, it also makes use of a semantic hierarchy and associated lexicon created by the expert for the task. Bagga et al. [1] system generalizes from sentences selected by an expert by using also an annotated corpora (in order to discover the best generalizations made from these sentences). LIEP [3] does not need annotated training texts, but rather has an interface that allows a user to identify over the text relevant entities and relationships between them. LIEP patterns are induced from positive training instances and the generalization step allows to expand the patterns by including a disjunctive list of terms expressing the same semantics. Finally, PALKA [4] builds inductively IE patterns but requires answer keys for the training texts, a predefined semantic hierarchy, and an associated lexicon.

All these systems acquire IE patterns without the hard work of hand writing them. However, they need an annotated training corpus. This is also a tedious work that must be done by a human expert.

The only exceptions we know are AutoSlog-TS [11], that acquires conceptual patterns for IE using only a preclassified training corpus without text annotations, and Riloff et al. [10] that present a new method to learn simultaneously a dictionary of extraction patterns and a domain specific (semantic) lexicon.

In addition, all the mentioned systems (centered in IE on free

[1] TALP Research Center, Universitat Politècnica de Catalunya. Jordi Girona 1-3, E-08034, Barcelona, Spain. email:{ncatala,castell}@talp.upc.es

[2] Departament de Llenguatges i Sistemes Informàtics, UPC. C/Jordi Girona 1-3, E-08034, Barcelona, Spain. email:mmartin@lsi.upc.es

[3] Or from an annotated corpus in which the targeted information is marked and tagged with domain specific semantic tags.

[4] Texts annotated with domain specific tags.

texts), need semantic knowledge in order to generalize in the right way (unlike IE systems for structured texts, where syntactic information is usually enough to build IE patterns). This semantic knowledge needed is usually represented as a domain specific ontology that (with the exception of Bagga et al. [1]) must be created by an expert.

Our goal is to minimize the effort of the expert to only defining the task and supervising the final results. This implies to acquire patterns without annotated corpora and with general domain ontologies (WordNet [6] in our case). This is achieved with the Essence methodology that lies in the ELA algorithm.

3 ESSENCE

The ESSENCE methodology is intended to reduce human expert intervention when acquiring IE patterns. This goal is achieved by means of a pattern generalization (learning) algorithm, named ELA, which delays as much as possible the expert involvement simplifying the amount of information he/she has to deal with. Nevertheless, a human expert is still required in order to validate the results and specify the kind of information to extract.

In order to make ESSENCE a portable methodology, needed knowledge sources and NLP tools also require this property. For that reason, a general-purpose lexicon such as WordNet, has been selected. WordNet offers lexical, syntactic and semantic information which is decisive in the generalization process. However, the lack of coverage for some domain specific words such as entity names, has been overcome by using gazetteers and domain specific word lists. Portability of NLP tools, such as the syntactic parser, is assured by our system because they are also domain independent.
The distinctive features of ESSENCE can be synthesized as:

- The training corpus has no annotations, neither syntactic tags nor semantic tags, but must include positive examples of information to be extracted.
- Human intervention is restricted to the task definition and typification and validation of patterns.
- For the generalization (learning) process a semantic hierarchy is needed. ESSENCE makes use of WordNet, able to cover multi-domain vocabulary instead of a hand built semantic hierarchy tailored for each new domain.

The overall performance of the ESSENCE methodology, depicted in Figure 1, is summarized here by giving a short description of its component modules.

Since IE is domain specific by nature and also oriented to an specific task, the first step of the methodology is to define the specific kind of target information. Therefore, a supplementary module, named `Task Definition` module, have been designed to assist the expert in this work. Basically, for each slot of the output template he/she defines a set of keywords (words that commonly appear together with the kind of information that the slot defines) and a set of synset numbers (WordNet tags that typify semantically the values that the slot can take). The set of keywords is not so difficult to find as it could seem. The expert can explore the training set in order to find an initial set of words that is automatically completed by finding in WordNet synonyms and hyponyms of these words. The set of synsets is also easy to find. When an IE task is defined, the client must define which kind of information wants to obtain. The type of this information is simply expressed in synsets of WordNet. In case the expert was not familiar with WordNet, he/she is aided with an easy-to-use interface.

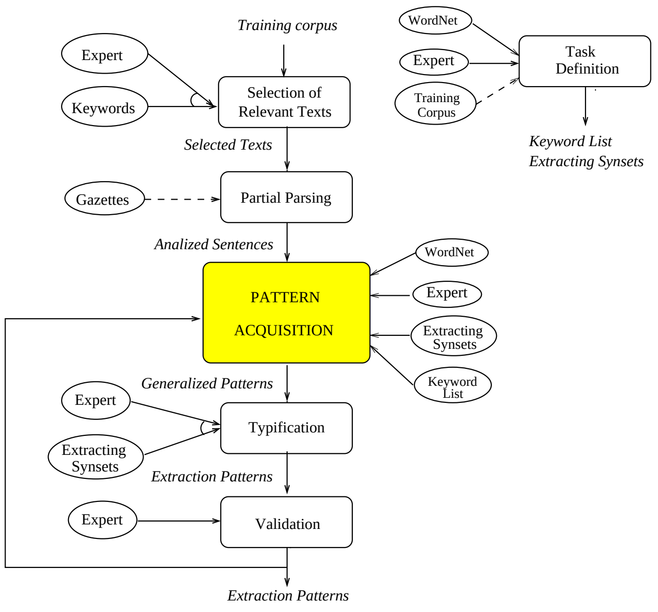

Figure 1. Overview of the ESSENCE methodology

The `Selection of Relevant Texts` module, selects from the training corpus a set of texts based on either expert criteria or useful known keywords relative to the given domain.

Major syntactic constituents such as noun phrases, verb phrases or preposition phrases, are identified by the `Partial Parsing` module. These high-level constituents will be the syntactic components of patterns and become their syntactic constraints when generalizing.

The multi-module `Pattern Acquisition` represents the core of the methodology and produces the generalized patterns starting from a set of analyzed sentences. It comprises five sub-modules (not depicted in the figure):

1. `Filter for Relevant Sentences` module: relevant sentences will be those that contain one or more keywords. This filtering allows the system to focus on the specific kind of target information.
2. `Windowing` module: from relevant sentences we collect parameter-sized context windows. A window is the context surrounding an occurrence of a keyword, and the size (width) of a window is the maximum number of syntactic groups it includes. For each syntactic group we have the list of words it contains along with its corresponding part-of-speech.
3. `Semantic Tagging` module: links to each group's headword[5] semantic labels which are represented by senses or word meanings corresponding to WordNet's synsets. This module provides a collection of *specific patterns* that will feed the learning algorithm.
4. `Learning Algorithm` module: from the set of specific patterns it generates a set of general patterns. This module is explained in 3.1 to a greater detail.
5. `Filtering` module: patterns obtained with all non-keyword components not generalized will be discarded. This process allows to discard too specific patterns retained by its high frequency when in fact correspond to literally repeated pieces of text.

The `Typification` module, "give names" to different pattern

[5] At this moment, semantic labels are reserved to nouns and verbs because they are the only that present the hyponymy/hyperonymy hierarchical relations needed for the generalization process.

components (that, in fact, are the roles they play), indicative of the kind of information they will extract. It determines which component of a pattern will fill which slot of the final output template.

Finally, the set of extraction patterns obtained must be validated applying them to a test corpus. Results in this stage will serve as feed-back for a new execution of the generalization process. For example, the expert can change the window size parameter or enlarge the keyword list. This process is done in the `Validation` module.

3.1 The ESSENCE Learning Algorithm: ELA

The learning algorithm is the core of the ESSENCE methodology. Inspired in ML techniques, the algorithm explores the set of specific patterns provided by the `Semantic Tagging` module (see third submodule of the `Pattern Acquisition` module), finding regularities in them that will be used to build a set of general patterns useful for the IE task. The learning algorithm proposed has been designed for this specific task. It builds extraction patterns by generalizing in a existing ontology which is not usual in other learning tasks. It learns from a set of observations instead of a set of examples -thus, the algorithm is not supervised (though an expert will check the final results). These features make the algorithm different from previous approaches, resulting in some aspects more close to a data mining algorithm on a NLP task than the classical concept learning approaches.

In our approach, an IE pattern can be composed of `Noun groups`, `Verb groups` and `Preposition groups`. Each group is tagged with a set of WordNet's semantic tags (called synsets) that is determined by the generalization procedure described bellow. This set of groups defines which kind of groups must contain a sentence in order to satisfy the IE pattern.

Groups in an IE pattern are either *contextual* or *informative*. The first kind of groups indicates that a sentence must contain groups matching these contextual groups in order to be covered by the IE pattern. We say that a group of a sentence *matches* a group of the IE pattern when the head of the sentence's group has a semantic tag that is a hyponym in WordNet of the synset of the pattern's group[6]. The second kind of groups not only indicates which groups must contain the sentence but also that those groups carry relevant information for filling one slot of the IE task. Informative groups are automatically identified because they carry as semantic information a synset that is the same or a hyponym of one extracting synset defined for one slot in the definition of the IE task (see `Task Definition` module in Section 3).

Information from a sentence is extracted using an IE pattern when the sentence contains, for each group of the pattern, a group that matches with it. In this way, the information extracted is the contained in the groups of the sentence that match with the informative groups of the IE pattern. As it may seems not only informative groups are important in a IE pattern. Context groups are useful in order to increase precision of the pattern.

In our approach, IE patterns are build from specific patterns. As it is described in the ESSENCE methodology, an specific pattern is a windowed sentence tokenized in syntactic groups and with the head-word of each syntactic group tagged semantically with the corresponding set of WordNet synsets. This structure, considered as an IE pattern, is too specific because describes a set of constraints that only can be satisfied by the windowed sentence that originated it. We need more general patterns.

[6] Additional constraints can be added in order to consider the matching of two groups. In our experiments we also ask for the same preposition in preposition groups. In other experiments we could also ask for the same voice in verb groups or the same number in noun groups.

```
Function cycle (maximum relaxation, set of specific patterns)
              returns general_pattern
 Set $current\_pattern$ to a specific pattern randomly selected
 Set $pattern\_list$ as the list with only $current\_pattern$
 Set $l\_pats$ to the empty list
 While not empty $pattern\_list$ do
    Remove the first pattern from the $pattern\_list$
    Add the $current\_pattern$ to $l\_pats$
    Create, for each specific pattern $sp_i$ (not covered by $current\_pattern$)
       and $current\_pattern$, all generalizations that contain at least one
       informative group in $sp_i$ not extractable by patterns previously
       learned. They are stored in $pattern\_list$.
    Sort the $pattern\_list$ with the relaxation measure
    If not empty $pattern\_list$ Then
       Set $current\_pattern$ to the first pattern of the $pattern\_list$
    EndIf
 EndWhile
 Evaluation of the list of patterns $l\_pats$ and selection of the best pattern
 If positive evaluation of $l\_pats$
    Then Return the best pattern
    Else Return NIL
 EndIf
EndFunction
```

Figure 2. Function `cycle`. See explanation on text.

A general pattern is an IE pattern describing a set of constraints that are fulfilled by several specific patterns (that is, for each group of the general pattern, there exists a group in these specific patterns that matches with it). In this case we say that all these specific patterns are *covered* by the general pattern.

The way to obtain general IE patterns that we propose starts by initially setting it to a randomly selected specific pattern and then, repeatedly generalizing it in order to cover at each repetition a new specific pattern. Generalization is done by relaxing the semantic tag associated to groups of the pattern and/or by removing groups of the pattern when the former is not possible.

Relaxation of the semantic tag is made in order to allow the match between two groups. For instance, assume that the general pattern has a group with the semantic tag AVALANCHE and the specific pattern has a group tagged semantically as CRASH. If the semantic ontology shows that both cases are hyponyms of ACCIDENT, then the two groups can match if the semantic tag of the general pattern is relaxed (generalized) to ACCIDENT.

The selection of the new pattern to cover at each time is made by searching in the set of all specific patterns the closest to the general pattern on hands. The metric that measures this similarity (that we call *relaxation measure*) takes into account first the number of groups that match and later the generalization that has to be made in the semantic ontology in order to allow the match between groups. Specifically, the generalization in the ontology is measured by counting the number of concepts in the ontology that must be climbed in order to find a concept that covers the groups of both patterns.

Figure 2 shows the implementation of this idea in a function called `cycle` that, from a randomly selected pattern, tries to obtain a general IE pattern. The function takes into account the information covered by other general IE patterns learned previously in order to obtain a general IE pattern that does not cover redundantly the same information.

Figures 3 and 4 illustrate how generalization is performed. The first figure shows two sentences extracted from the corpus used in our experiments which are selected because they present keywords, in this case `crashed` and `slammed`. The figure also shows the corresponding specific patterns from these sentences obtained by tok-

Sentences:

1- The flier whose Navy F-14A fighter plunged into a Nashville suburb *on Monday, killing himself and four other people,* **crashed** *another jet into the sea last April.*

53- Commerce Secretary Ron Brown and 32 others on a Balkan trade mission were presumed killed *when their plane* **slammed** *into a Croatian hillside during heavy storms Wednesday.*

```
(1 ((PP ((PREP ON NIL) (DATE @@@MONDAY@@@ NIL)))
    (PP ((PREP KILLING NIL) (NPST HIMSELF NIL) (CONJ AND
         NIL) (NOUN FOUR (9896114)) (ADV OTHER NIL)(NOUN
         PEOPLE (5957883 6069040 6080290 5976176))))
    (VP ((VERB CRASHED (1342612 1431218 1076088
         1379139 1076294 1813216 1378886 1076442 1035304
         303160 12099))))
    (NP ((NPST ANOTHER NIL)
         (NOUN JET (2875044 5531909 10710122 2717915))))
    (PP ((PREP INTO NIL) (NPST THE NIL)
         (NOUN SEA (6781925 9922052 7845203))))
    (PP ((DATE @@@LAST_APRIL@@@ NIL)))))

(53 ((PP ((CONN WHEN NIL)))
     (NP ((NPST THEIR NIL) (NOUN PLANE (2174460 9985988
          10046013 3137218 3136725))))
     (VP ((VERB SLAMMED (847148 846778 1295221 847023))))
     (PP ((PREP INTO NIL) (NPST A NIL) (NOUN CROATIAN
          (7052505)) (NOUN HILLSIDE (6724431))))
     (PP ((PREP DURING NIL) (NOUN HEAVY (7314549 4553181)
          ) (NOUN STORMS (7803078 10069810 627161))))
     (PP ((DATE @@@WEDNESDAY@@@ NIL)))))
```

Figure 3. Example of specific patterns. Note that sentence number 53 is divided in two by the syntactic analyzer, being the second one (with the keyword) relative to the main one.

enizing them syntactically and limiting them with the window-size parameter set to six. The headword of each syntactic group is tagged with the corresponding synset tags obtained from WordNet. The `NIL` label indicates that a word is not defined in WordNet as noun or verb. Some words are surrounded by "@". This indicates that the word has been recognized by auxiliary linguistic modules as a relevant semantic information that is not described in WordNet (in our experiments dates, companies and specific airplane models).

From these two specific patterns, the generalization procedure returns the set of all possible generalizations that is shown in Figure 4. They are not still sorted by the relaxation measure. Note that all patterns have a relaxation value below 13 because this has been selected as the maximum relaxation allowed. Note also that they present 3 matching fields, because we set the number of matches to this value.

From this set of general patterns, the `cycle` procedure will select the third one because it shows the minimum relaxation value. The obtained IE pattern covers sentences that describe a crash and that also cite the date and present a noun group with a word described in WordNet as an hyponym of the `airplane` concept. The `cycle` function will repeat the process again by searching for another specific pattern to cover, generalizing the IE pattern in order to cover this new specific pattern, and so on until no generalization that covers new specific patterns is possible.

The `cycle` function must be wisely called in order to generate a complete set of general IE patterns for the current IE task. The point is to call the `cycle` function with different initial specific patterns until no specific patterns remain to be used as the seed for obtaining a general pattern or until all informative fields of each specific pattern are covered by the set of generated IE general patterns[7]. The resulting algorithm is called ELA and is shown in figure 5.

[7] Thus, the selection procedure in the cycle function used to choose randomly the initial specific pattern will also consider that the specific pattern selected has to be not previously used as the initial pattern in a previous cycle.

Structure:
((number of matches, measure of relaxation),
list of general fields that compose this general pattern)

```
((3 6)
   ((VP ((VERB (CRASHED SLAMMED) NUCLI)))
    (PP ((PREP INTO NIL) (NOUN (SEA HILLSIDE) (9457))))
    (PP ((PREP NIL NIL) (DATE (@@@LAST_APRIL@@@
          @@@WEDNESDAY@@@) NIL)))))
((3 7)
   ((VP ((VERB (CRASHED SLAMMED) NUCLI)))
    (PP ((PREP INTO NIL) (NOUN (SEA HILLSIDE) (9457))))
    (NP ((NOUN (JET PLANE) (2174460))))))
((3 1)
   ((VP ((VERB (CRASHED SLAMMED) NUCLI)))
    (PP ((PREP NIL NIL) (DATE (@@@LAST_APRIL@@@
          @@@WEDNESDAY@@@) NIL)))
    (NP ((NOUN (JET PLANE) (2174460))))))
((3 10)
   ((VP ((VERB (CRASHED SLAMMED) NUCLI)))
    (PP ((PREP NIL NIL) (DATE (@@@LAST_APRIL@@@
          @@@WEDNESDAY@@@) NIL)))
    (NP ((NOUN (JET PLANE) (2859872))))))
((3 12)
   ((VP ((VERB (CRASHED SLAMMED) NUCLI)))
    (PP ((PREP NIL NIL) (DATE (@@@LAST_APRIL@@@
          @@@WEDNESDAY@@@) NIL)))
    (NP ((NOUN (JET PLANE) (9457))))))
```

Figure 4. General patterns with 3 matches obtained from specific patterns shown in Figure 3. Note that the three last patterns match the same groups but with different synset numbers.

```
Algorithm ELA (specific patterns)
  Initialize $covered\_groups$ to NIL
  Set $general\_patterns\_set$ to the null set
  While not remaining information from specific patterns to be covered or
        not remaining specific patterns to be used as seed in cycle do
    $gen\_pat$:= cycle(num. matches, max. relaxation, set of spec. patterns)
    If $gen\_pat$ was accepted by the expert in the cycle function
      Then
        Add $gen\_pat$ to $general\_patterns\_set$
        Mark the groups of specific patterns covered by $gen\_pat$
    EndIf
  EndWhile
EndAlgorithm
```

Figure 5. ELA algorithm.

4 EXPERIMENTS

The set of experiments presented here are extracted from the results obtained on training ELA on MUC7 dry-run texts. These texts are separated in two sets composed of 100 texts from the New York Times News Service (one set for training and the other for testing). Not all the news describe the crash of a flight but all of them mention at least one crash. The task for the scenario template was to find out information about aircraft crashes or accidents, such as the location and the date of the accident, the number of victims or the aircraft involved in.

Although ESSENCE does not need a corpus with answer keys about the information to extract, MUC style competitions deliver them. We will use the answer keys not for learning but only to automatically validate the patterns generated, releasing the expert from this task and obtaining in this way results directly comparable with other systems. Validation will be done with the known measures of *Recall*, *Precision* and the mixture of them *R&P* (also known as *F* with β value set to one) [5]. In short, Recall measures the coverage of the set of IE pattern and Precision measures the quality of the IE patterns obtained. Both values are expressed as percentages. A 100% of Recall indicates that all information that had to be extracted were actually extracted. A 100% of Precision indicates that all information extracted was right.

Specifically, the ESSENCE methodology was used to extract the site and date of a flight crash jointly with the departure site, the destination site and the airline of the flight, and the manufacturer and the kind of aircraft that crashed.

In the experiments, we used a modified version of MARMOT as the syntactic analyzer with a general module for date detection and lists of companies and aircraft.

The user defined two types of keywords, one for crash info (that includes the slots `crash-site` and `crash-date`) and another for flight info (`aircraft, airline, manufacturer, departure` and `destination`).

Keywords used for the slots of crash information (place and date) were selected from 6 set of words that usually describes a flight crash. The words selected were CRASH for expressing the crash of a flight, FALL describing the fall of an aircraft, DISAPPEAR for expressing the disappearing of a flight from radar screens, EXPLODE for describing a flight accident by explosion, PLUNGE for expressing a crash into water, and KILL expressing an accident were people died.

This set of words was expanded with their synonyms and hyponyms from WordNet, giving the following complete set of keywords:

BUMP, CLASH, COLLIDE, CRASH, HIT, JAR, KNOCK, RAM, SHOCK, SLAM, STRIKE, DESCEND, DOWN, FALL, LAND, DISAPPEAR, LOSE, BLEW, BOMB, EXPLODE, FIRE, HIT, STRIKE, DIE, KILL, PERISH, DIVE, NOSEDIVE, PLUMMET, PLUNGE.

In the same way, keywords for the `destination` and `departure` slots included words making reference to LEAVE, GO, RETURN, TRAVEL, LAND, FLY and APPROACH. This set of words was completed with synonyms and hyponyms as in the previous case. `Airline` and `manufacturer` slots had defined the keywords used for crash information plus new ones that makes reference to seller-buyer relations, as BUY, ORDER, OWN, RENT, BORROW and DELIVER, also completed with the help of WordNet. Finally, keywords for the `aircraft` slot included all the previous keywords.

From this set of keywords and the set of acceptable synsets defining the kind of information for each slot, the ESSENCE methodology was applied over 100 texts that compose the training set.

The set of patterns obtained was automatically tested in both the training set and the test set in order to validate it. Results in recall, precision and R&P for each slot to be filled in the IE task are shown in table 1.

Table 1. Results for the Aircraft Crash domain.

	Training set			Test set		
Concept	**R**	**P**	**R&P**	**R**	**P**	**R&P**
Crash info						
Crash Site	67.6	68.0	**67.8**	59.4	51.2	**55.0**
Crash Date	78.3	62.0	**69.2**	75.4	82.6	**78.8**
Flight Info						
Aircraft	69.0	100.0	**81.6**	65.0	100.0	**78.8**
Airline	55.7	57.9	**56.8**	65.2	55.2	**59.9**
Manufacturer	56.2	53.6	**54.9**	39.5	62.3	**48.4**
Departure	52.1	87.7	**65.3**	57.6	51.5	**54.4**
Destination	54.3	94.8	**69.1**	60.7	72.9	**66.3**

Results presented show an average level of P&R of 66.4% in training and 63.1% in testing, that is reasonable high compared with the performance of other systems in similar tasks. Note that in some slots the results obtained for the training set are better than for the test set. This effect appears because IE patterns are acquired using non-supervised techniques and because of the small number of cases in both sets.

Nevertheless, some of these results can be improved. In particular, the Airline and Manufacturer values can be greatly improved by allowing the extraction of information from *modifiers* of the headwords. Preliminary results show that in this case, the values for these slots can be raised to near 80%.

5 CONCLUSIONS AND FUTURE WORK

In this paper we have presented a new methodology, named ESSENCE, for acquiring IE patterns to build IE systems. The main advantage of this methodology is that it reduces the effort of the expert in the process of developing an IE system, therefore decreasing the cost of production. This is achieved by centering the effort of the expert on the definition of the task and on the validation and typification of patterns, while tedious tasks have been automatized by the use of Machine Learning techniques, and linguistic resources and tools.

The linguistic components are domain independent, is the case of WordNet and the syntactic analyzer. The independence of the linguistic tools from the IE task also ensures the easy portability of the methodology to build new IE systems.

Moreover, the use of general tools allow the methodology to be applied to other languages. For instance, Catalan and Spanish languages have been participants in the EuroWordNet project that elaborates a multilingual version of the initial English WordNet. This lexical resource along with available NLP tools developed in our research group for these languages (see http://www.lsi.upc.es/∼acquilex/nlrg.html), will allow us porting the ESSENCE methodology to them.

ACKNOWLEDGEMENTS

Work partially supported by Catalan Government (1999SGR-150).

REFERENCES

[1] A. Bagga, J.Y. Chai, and A.W. Biermann, 'The role of WordNet in the creation of a trainable MESSAGE UNDERSTANDING system', in *Proceedings of the Ninth Annual Conference on Innovative Applications of Artificial Intelligence (AAAI-97)*. AAAI Press/The MIT Press, (1997).
[2] J. Cowie and W. Lehnert, 'Information extraction', *Communications of the ACM*, **39**(1), 80–91, (1996).
[3] S.B. Huffman, 'Learning information extraction patterns from examples', in *IJCAI-95 WSP on New Approaches to Learning for NLP*, 1995.
[4] J. Kim and D. Moldovan, 'Acquisition of linguistic patterns for knowledge-based information extraction', *IEEE Transactions on Knowledge and Data Engineering*, **7**(5), 713–724, (October 1995).
[5] W. Lehnert and B. Sundheim, 'A performance evaluation of text analysis technologies', *AI Magazine*, pp. 81–94, (1991).
[6] G.A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller, 'Introduction to WordNet: An on-line lexical database', Technical report, Cognitive Science Laboratory, Princeton University, (1993).
[7] *Proceedings of the Third, Fourth, Fifth and Sixth Message Understanding Conference*. Morgan Kaufmann, 1991, 1993, 1995, 1997.
[8] *Information Extraction*, ed., Maria Teresa Pazienza, Lecture Notes in Artificial Intelligence, Springer-Verlag, Rome, 1997.
[9] E. Riloff, 'Automatically constructing a dictionary for information extraction tasks', in *Proceedings of the Eleventh National Conference on Artificial Intelligence*, (1993).
[10] E. Riloff and R. Jones, 'Learning dictionaries for information extraction by multi-level bootstrapping', in *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, Orlando, (July 1999).
[11] E. Riloff and J. Shoen, 'Automatically acquiring conceptual patterns without and annotated corpus', in *Proceedings of the Third Workshop on Very Large Corpora*, pp. 148–161, (1995).
[12] S. Soderland, D. Fisher, J. Aseltine, and W. Lehnert, 'CRYSTAL: Inducing a conceptual dictionary', in *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, (1995).