Learning Text Extraction Rules, without Ignoring Stop Words

João Cordeiro[1] and Pavel Brazdil[2]

[1]Department of Informatics, University of Beira Interior, Rua Marquês d'Ávila Bolama, Covilhã, Portugal
[1]Artificial Intelligence and Computer Science Laboratory (LIACC), U. Porto, Portugal

[2]Faculty of Economics of University of Porto, Porto, Portugal
[2]Artificial Intelligence and Computer Science Laboratory (LIACC), U. Porto, Portugal

Abstract. Information Extraction (IE) from text /web documents has become an important application area of AI. As the number of web sites and documents has grown dramatically, the users need an easy, fast and flexible ways of generating systems that can carry out specific IE tasks. This can be achieved with the help of Machine Learning (ML) techniques. We have developed a system that exploits this strategy. After training the system is capable of identifying certain relevant elements in the text and extracting the corresponding information. As input, system takes a collection of text documents (in a certain domain), that have been previously annotated by a user. This is used to generate extraction rules. We describe a set of experiments that have been oriented towards the domain of announcements (in Portuguese) concerning house/flat sales. We show that quite good results overall can be achieved using this methodology. In previous work some authors argue that stop words should really be eliminated before training. We have decided to re-examine this assumption and present evidence that these can be quite useful in some sub-tasks.

1 Introduction

Thanks to Internet, an enormous amount of information is available to us nowadays. To be able to exploit this we need automatic tools to help to identify the relevant information. This is why Information Extraction (IE) from text /web has become a very important research area.

In the earlier days of Information Extraction research, the IE systems (often referred to as *wrappers*), were created manually and were oriented towards specific tasks. Broadly speaking, these systems implement a set of patterns or rules that can be used to process particular documents and extract the elements of interest [1]. Typically, the output is represented as entries in a table, where each line corresponds to one particular case and the columns represent different elements of interest.

In recent years the interest has turned to systems that could be easily adapted to new tasks. It has been shown that the required extraction patterns (or rules) can be

Cordeiro J. and Brazdil P. (2004).
Learning Text Extraction Rules, without Ignoring Stop Words.
In *Proceedings of the 4th International Workshop on Pattern Recognition in Information Systems*, pages 128-138
DOI: 10.5220/0002681601280138
Copyright © SciTePress

obtained with the help of inductive or Machine Learning (ML) techniques [2], [3]. Our work described here exploits the ML approach.

Our main aim is to consider a collection of text documents in a certain domain – for instance, announcements concerning houses/flats on sale - and extract certain relevant information. So for instance we may be interested in details concerning *price*, *type*[1] and *location* of different houses/flats.

Our system requires a set of annotated documents in which the elements of interest are duly identified. The annotated documents are used to generate a training set, which in turn is used to generate extraction rules. These rules can then be applied to new, unseen documents.

In the next section (2) we present an overview of the system. Section 3 gives more details on how the training data is obtained from a set of annotated documents. In that section we discuss also the issue of selecting relevant attributes. Section 4 discusses how the induced extraction rules are generated and used to carry out IE tasks. Section 5 describes the results. Section 7 is dedicated to relation to other relevant work. Finally, Section 6 concludes this paper.

2 Overview of the System

In this section we present a brief overview of the system. Basically, it comprises the following three phases. First it is necessary to prepare the data needed later. This phase is carried out by the user. Once this has been done, we can proceed to use a chosen ML to identify relevant documents generate text extraction rules. Finally, the extraction rules can be applied to new contexts. More details are given below.

Phase 1: Data Preparation

This phase comprises two steps, both of which need to be carried out by the user.

Phase 1.1: Consider certain set of documents and identify (label) those that are of interest, for a particular IE task. If our aim is to extract information from advertisements concerning house / flat sales, we need a sample of documents which includes both some relevant documents (those on sales) and some irrelevant ones. This set needs to be labelled (classified)[2].

Phase 1.2 Identify the relevant elements in the relevant documents. Identify attributes and generate training instances.

More details concerning this are given in Section 3.

Phase 2: Generating Text Extraction Rules

Phase 2.1: The documents obtained in Phase 1.1 are used to train a system to identify the relevant ones. In this phase we have used an enhanced Naïve Bayes [4].

Phase 2.2: The documents obtained in Phase 1.2 are used to train a system to identify/extract the relevant elements. Each element type is handled separately.

[1] The item "type" is used to describe how many rooms the house / flat has.

[2] Here we have used documents that appeared at the *News Group* "pt.mercado.imobiliario".

That is, a separate training is done to extract e.g. the element *price*. This is repeated for the other element types.

Section 4 gives more details concerning how this is done.

Phase 3: Processing New Documents

Phase 3.1: The system generated in Phase 2.1 is used to classify new documents. Those that are classified as *relevant* are passed to the next phase.

Phase 3.2: Different text extraction rules are applied to the documents identified above to extract the relevant information.

Section 5 describes the results obtained with our system on a new set of documents (documents that were not used in training).

3 Generating the Training Data

The issue of generating appropriate training data is a non-trivial issue [5], [6]. In this section we describe how this is done.

Each particular element type is handled separately. Let us assume, for instance, that we are concerned with the element type *price* (representing prices of houses / flats).

The aim of Phase 1.2 is to identify the relevant elements (i.e. price in our case) in each document. This is done using XML tags which are supplied by the user. An example showing such tags is given in Fig. 1. The tags <preço>[3] and </preço> mark the beginning and the end of the element price, which in this case is "117 500".

```
---------------------------------------------------------------------
  "... uso de apenas seis meses. O motivo da venda está
relacionado com trabalho no estrangeiro e o preço pedido é:
<preço>117 500</preço> euros (negociável).
  Contacto: Armando Vieira 934592123 ..."
---------------------------------------------------------------------
```

Fig. 1. Occurrence of the price element

The next problem is how we can generate a training example for the chosen Machine Learning (ML) system. The basic idea is to consider the words in a certain neighbourhood of the given element (i.e. *price*). Here we refer to the neighbourhoods as *contexts*. We distinguish between the *left context* (the one preceding the element) and the *right context* (the one succeeding the element). Each context has a certain size (e.g. 3 words on the left). In Fig. 1 both contexts have the same size and are identified by underlining.

As the example shows certain words appear in the each context. In our example the left context includes the words "*preço*", "*pedido*" and "*é*". These words are then considered as *candidate attributes*, together with all the other words that appear in the other annotated examples[4].

[3] The tag word shown represents the Portuguese word for "*price*".

[4] All words have been stripped off the punctuation marks and converted to lower case.

Later we will discuss the issue of stop words and usage of generalized concepts that are often used in IE systems. For now, let us say that each context is represented as a *bag-of-words*. That is, we do not care about in which position a particular word appears. All we care about is whether a particular word is present (or absent) in the given context. This is represented by values true/false.

Obviously, if we consider many annotated examples, the resulting set of candidate attributes can be quite large. Therefore we need some ways of selecting the relevant attributes from this set.

3.1 Selecting Informative Attributes

Many different methods exist that can be used to select a subset of attributes from a given set. Here we use the *Information Gain (IG)* measure [5] to calculate the IG value for all possible attributes. All attributes can thus be ranked and the most informative ones selected. If we use this measure in conjunction with the attributes originating in the left context of the element *price*, we obtain the result shown in Fig. 2.

Attribute (word)	Information Gain (IG)
preço	0.200
valor	0.170
venda	0.124
por	0.070
assunto	0.065
de	0.005
o	0.003
a	0.002
…	…

Fig 2. Ranking of attributes appearing in different left contexts of the *price* element.

Having ordered the attributes, we can select the best ones, that is, for instance those that satisfy IG > ε (where ε is pre-fixed value). If, for instance, the limit were 0.010 then the first 5 attributes in Fig. 2 would be chosen and the others dropped.

3.2 Note on Stop Words

Stop words are by definition common words, like conjunctions (e.g. “and”, “or” etc.), prepositions (“with”, “without”, “from” etc.) or other similar categories. One of generally accepted beliefs in Information Retrieval is that these words are quite irrelevant and have no effect on the final results [7], [4]. As a consequence, many systems use a pre-defined list of stop words for the particular language in question. The list is used to remove potential word attributes from consideration. A question arises whether this strategy is right.

We verified that, in our domain, some stop words may represent important pieces of information. One example can be seen in Fig. 2. The Information Gain (IG) value

of the attribute "*por*" is not negligible. We can explain why this is so. This preposition is commonly being used in Portuguese to indicate the price ("*por 1100*" means "*for 1100*").

Other examples of stop words can be seen Fig. 5, which shows examples of some extraction rules generated. We note that some stop words are present there. One of them is the preposition "*em*" in one of the rules used for extraction of "*local*" (location). We note again that this makes sense because it is usual to use the word "*em*"[5] before the name of a place (as in "*em Lisboa*", "*em Sintra*" etc).

We have verified that the performance figures (see Section 5) would decrease, if the stop words were simply eliminated. So we do not do that, but rather rely on the IG measure to determine which attributes should be retained or dropped.

3.3 Characterizing the Focus Element

Apart for left and right contexts, we also consider the textual expression appearing between the two tags, which is used in the extraction of the relevant information.

Here we refer to it as ***f*** (the focus element). Let us consider again Fig.1 shown earlier. The focus element in this example is "117 500".

The system tries to generalise the string found, using a pre-defined generalisation rules provided by the user. So, for instance, *"117 500"* would be generalized into "*NUMBER*". Similarly, "*100 mil*"[6] would be generalized into "*NUMBER mil*" etc.

A similar strategy would be used in other extraction tasks. So when dealing with extraction of *local* (location), the string *"Porto"* would be generalized into "*LOCAL*". Similarly, for instance, "*Rua S. Catarina no Porto*" would be substituted by "*LOCAL no LOCAL*" etc. Here again the system uses domain-specific generalization rules (provided by the user). Besides, it uses also the list of many existing locations in Portugal (towns, villages, streets), available at one of the existing web sites. The list of locations is used to recognize the corresponding names in the text.

Let us refer to a particular generalized focus elements as ***f****. In general a particular focus element ***f*** may give rise to different generalizations ***f****.

The generalisations encountered in different training examples are stored for further use. Here we will refer to this set as ***F****. Fig. 3 shows a set of generalized elements relative to the element *local*.

[5] "em" means "in" as "in Lisbon".

[6] The word "mil" means "thousand", in Portuguese.

```
-----------------------------------
   “quinta do s.joão”,
   “s.joão do LOCAL”,
   “LOCAL LOCAL”,
   “praia da LOCAL”,
   “quinta do LOCAL”,
   “sta LOCAL”,
   “LOCAL”,
   “LOCAL do bosque”,
   “stª eulália”,
   “LOCAL do LOCAL”,
   “LOCAL da LOCAL”,
   “s.b. LOCAL”,
   “praceta quinta de s. joão”,
   “stºovideo”,
   “vn LOCAL”,
   “s. LOCAL”
-----------------------------------
```

Fig.3. Generalized elements used in the extraction of *local*

3.4 Examples of Training Instances

In this section we present some examples of actual training instances, complementing the explanation given above. First, let us examine an example of a source text shown in Fig. 4.

```
-------------------------------------------------------------
No Cacem, na freguesia do Cacem, concelho de Sintra, vendo
apartamento t3, junto da estação da CP, com quartos 15,19 m2;
15,66 m2; 18,56 m2 e sala
<¬preço>31,10</¬preço> m2, ainda está em estuque nunca foi
pintada, é espectacular; belíssima vista desafogada, muito
soalheira.

Boa oportunidade!
PREÇO: <preço>128.440,00</preço> euros
mais informações tel. 219136000, tlm. 917621828
-------------------------------------------------------------
```

Fig 4. Example of source text for the extraction of *local.*

The next figure presents some training instances generated, relative to the *price* element. One is positive (the value of the class “Extract” is T) and the other negative (the corresponding value is F).

preço ∈ l	*valor* ∈ l	*vendo* ∈ l	*por* ∈ l	*euro* ∈ r	***f****	**Extract**
T	F	F	F	T	NUM	**T**
F	F	F	F	F	NUM	**F**

Fig 5. Examples of training instances generated

The first line contains the word attributes. It includes, for instance, the attribute "preço ∈ L", among others. This attribute is true (T), if the word "*preço*" appears in the left context (l).

The actual attribute values relative to the positive instance are shown on line 2. As the word "*preço*" is present in the left context, the value of the corresponding attribute is T. Also, we note that the word "*euros*" appears in the right context (r) and so the value of the corresponding attribute is also T.

The value of the ***f**** attribute (generalized focus) is "NUM", meaning the element is a number.

After the positive instances have been generated, the system proceeds to generate negative instances, using a *near-miss* strategy [8]. This amounts to searching for patterns in the text that are similar to the positive examples, but are not classified as such. Concretely, we search for occurrences of ***f**** attribute elsewhere in the text. One such situation is identified by the tags **<¬preço>** and **</¬preço>** in Fig. 4. This corresponding text string gives rise to the negative instance shown in Fig. 5.

4 Generating Extraction Rules

The set of instances obtained from the annotated text documents are submitted to system C5, or more precisely, to C5rules [9], to generate the extraction rules.

For every relevant element there is a separate set of training instances (each including both positive and negative ones). Each set is used to generate the extraction rules.

Fig. 6 shows examples of some rules generated. The rules in part (a) are relative to the extraction of *price* (*preço*), while the rules in part (b) are oriented to *location* (*local*). The symbols *l* and *r* refer to the left and right contexts.

The condition f*∈ F* is interpreted as follows. The condition is true if the generalized focus element (f*) is one of the items in F* (stored there during training).

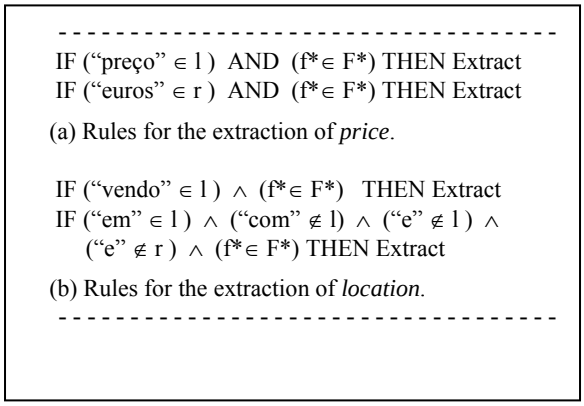

Fig 6. Examples of extraction rules generated

4.1 Applying the Extraction Rules

The extraction rules are applied to new text as follows. The text is considered as a string of words. The system moves through this string and tries to identify places which satisfy the condition of the focus element (i.e. f*∈ F*). Let us assume that our aim is to extract the location using the rules shown in Fig.6 and that the text being considered is:

“*assunto: vendo no Porto, apartamento t3 como novo*”

The condition f*∈ F* is true, as the word “Porto” is generalized into “LOCAL” and this is one of the items of F* (see Fig. 3). As “vendo” appears in the left context, the rule

IF (“vendo” ∈ l) ∧ (f*∈ F*) THEN Extract

is satisfied and consequently the extraction takes place.

5 Results

We have evaluated our system on 200 documents, all from the domain described earlier (house / flat sales). The documents were divided into two groups, referred to as A and B. Both contained various occurrences of the elements of interest. Fig. 7 gives the details.

	Group A	**Group B**
preço	62	92
tipo	138	211
local	187	220

Fig 7. Numbers of elements in each group

We have carried out two evaluations. In one, the documents in Group A were used to train the system, while those in B were used to evaluate it. In the second one, the two groups were interchanged (i.e. B was used for training and A for testing).

Here we have used precision, recall and F-measure that are usually used in IE to evaluate the performance [10]:

precision = TP / (TP + FP)
recall = TP / (TP + FN)
Fmeasure = 2*precision*recall / (precision+ recall)

Here TP, FP and FN represent the true positives, false positives and false negatives respectively.

The results obtained for each element type and each experiment are presented in Fig. 8.

Element	Train	Test	Precision	Recall	Fmeasure
preço	A	B	.788	.891	.837
	B	A	.727	.903	.806
tipo	A	B	1.000	.877	.934
	B	A	1.000	.609	.757
local	A	B	.651	.805	.720
	B	A	.783	.658	.715

Fig 8. Results for each element type

The mean performance in terms of F-measure on all three element types was 79.5%. In other words, the results were quite satisfactory.

As the results were achieved using symmetric contexts of size 3, a question arises whether this choice was right. We have decided to consider different sizes and examined its effects on performance. The results of our experiments are shown graphically in Fig. 9. The results indicate that contextual information is important, but up to a certain limit (3-4 words to the left and right). In other words, if it is extended beyond this limit, the performance may either stay the same, or even decrease for some extraction tasks (extraction of preço).

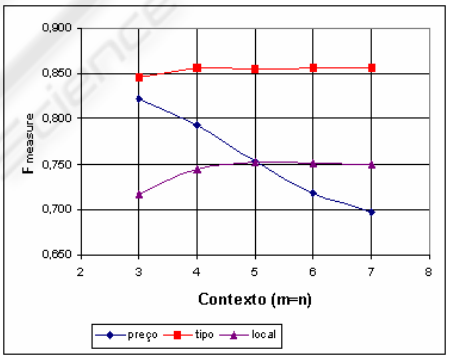

Fig 9. Effect of varying the contexts sizes

6 Relation to Other Work

Information Extraction from text has become a very active area of research. Some approaches began to explore learning approaches. This includes WIEN system of [2] the recent system of [3], among others.

The WIEN system is a wrapper induction system that tries to identify symbolic patterns present in the neighbourhood of the target information. It uses a collection of annotated training examples as input and induces the least general wrapper capable of extracting a given relevant element.

The system is oriented towards HTML documents and uses a denominated HLRT language, were H stands for the string that ends the "header", L and R the strings preceding and following the target element and T the string that begins a "tail" section of a web page. Therefore if, for example, the system tries to induce a wrapper to extract a person's name in some type of web pages, we may have T="Web Page</H1>", L="<B><I>" and R="</B></I>".

WIEN system works well if the information is in a semi-structured form, for example a HTML table. The information must not be very complex, otherwise the wrapper cannot be learned. This is due to the fact that WIEN takes in account formatting features. Another weak point of that system is that the system re-invents many induction procedures, instead of exploiting the existing ones.

More recently [3] have described a system that seems to be close to our own work. They use a two-stage classification process. First, a Naïve Bayes classifier is used to determine whether the given sentence is relevant. In the second stage the system attempts to identify the words belonging to the relevant element. This is a somewhat similar to the method used in our system. However, our first phase is oriented towards documents, not to individual sentences. We could try to extend our system to identify relevant portions of documents. There is another difference in the two approaches. Sitter & Daelemans use both words and part-of-speech (POS) tags that occur in left and right neighborhood. We could again try to extend our system with this facet. On the other hand, it appears that they do not use information about the target text to be extracted (our focus attribute), as we do.

7 Conclusions and Future Work

We have presented a description of a system that can be trained to extract certain relevant elements from text documents in a specified domain. The overall results were on the whole quite good (F measure of the order of 0.8).

The results indicate that contextual information is important, but up to a certain limit. Exceeding this does not bring further improvements.

We have carried out analysis regards the usefulness of word attributes. While it is generally believed that eliminating stop words is a good thing, we have shown that some may be useful in some tasks. We have suggested exploiting an existing measure to determine which stop words should be retained or dropped.

We believe that it would be relatively easy to adapt the system to other domains / tasks, but of course, this should be subject to further investigation in future.

There is scope for further improvements in future. We believe introduction of both syntactic (e.g. POS tags) and semantic information lead to improvements in the

overall results. As for the latter, we could contemplate just simpler extensions, helping, for instance, to determine the correct word sense. These could substitute the word attributes used here.

Alternatively, we could try to categorize the given words (or groups of words) into semantic categories. We could continue these extensions by trying to establish relationships between different categories (e.g. relate an object of an action with some agent and the action itself). This could give rise to new attributes (or predicates) that could be exploited in learning.

From a practical standpoint it would be useful to exploit, in the first place, semantic networks like *WordNet* for the Portuguese language. It would also be useful to make use of existing ontologies developed by others. These could be exploited to obtain generalizations not only for the focus element, but also in the right and left contexts.

Despite the fact that many improvements could be done, our work shows that even a relatively simple system could already be useful to carry out rather complex extraction tasks.

Acknowledgements

The authors wish to acknowledge the support provided by FCT (Fundação para a Ciência e Tecnologia) under so called Pluriannual Programme attributed to LIACC, and the funding received from POSI/POCTI.

References

1. Shadbolt N., *Caught up in the web*, Invited talk at the 13th Int. Conf. on Knowledge Engineering and Knowledge Manangement (EKWA02) (2002)
2. Kushmerick N., *Wrapper induction: efficiency and expressiveness*, Elsevier (2000), 15-68
3. Sitter A., W. Daelemans, Information Extraction via Double Classification, in Proceedings of the Int. Workshop on Adaptive Text Extraction and Mining (C.Ciravegna and N.Kushmerick, eds.), associated with ECML/PKDD-2003 Conf., Dubrovnik, Croatia, (2003)
4. Mladenic D., M. Grobelnik, Feature selection for unbalanced class distribution and Naive Bayes, in Machine Learning: Proceedings of the Sixtheenth International Conference (ICML'99), Morgan Kaufmann (1999)
5. Mitchell T. M., Machine Learning, McGraw-Hill (1997)
6. Witten Ian, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann (2000)
7. Craven M., D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam and S. Slatery, Learning to construct knowledge bases from the World Wide Web, Elsevier (2000), 69-113
8. Winston P. H., Artificial Intelligence, Addison-Wesley (1992)
9. Quinlan J. R., C5.0 Data Mining Tool, www.rulequest.com (1997)
10. Meadow T. Charles, B. R. Boyce, D.H. Kraft, Text Information Retrieval Systems, 2nd ed., Academic Press (2000)