Discovery of Language Resources on the Web: Information Extraction from Heterogeneous Documents

Viktor Pekar and Richard Evans
School of Humanities, Languages, and Social Sciences,
University of Wolverhampton, Stafford Street, Wolverhampton,
WV1 1SB, UK

Abstract

The present article is concerned with the problem of automatic database population via information extraction (IE) from web pages obtained from heterogeneous sources, such as those retrieved by a domain crawler. Specifically, we address the task of filling single multi-field templates from individual documents, a common scenario that involves free-format documents with the same communicative goal such as job adverts, CVs, or meeting/seminar announcements. We discuss challenges that arise in this scenario and propose solutions to them at different levels of the processing of web page content. Our main focus is on the issue of information extraction, which we address with a two-step machine learning approach that first aims to determine segments of a page that are likely to contain relevant facts and then delimits specific natural language expressions with which to fill template fields. We also present a range of techniques for the enrichment of web pages with semantic annotations, such as recognition of named entities, domain terminology and coreference resolution, and examine their effect on the information extraction method. We evaluate the developed IE system on the task of automatically populating a database with information on language resources available on the web.

Correspondence:
Viktor Pekar, HLSS, University of Wolverhampton, Stafford Street, Wolverhampton, WV1 1SB, UK.
E-mail:
v.pekar@wlv.ac.uk

1 Introduction

The World-Wide Web has become a popular vehicle for disseminating and obtaining research and educational resources. In Natural Language Processing (NLP), the specialist community has accumulated a large amount of language resources developed over many years, including software [part-of-speech (PoS) taggers, parsers, various corpus analysis tools], and data (evaluation corpora and datasets, frequency lists, glossaries, gazetteers). Most of these resources are freely available, which helps other researchers save effort on developing them and allows for direct comparisons with previous work. However, finding these resources on the web is not straightforward.

Traditional keyword search is not able to detect complex patterns in data of the kind that one is typically looking for when trying to locate a certain language resource. For example, current web search engines would not perform well on a query like '*list all freely available newspaper corpora in German with a size of over 1 million words*'. To actually find an answer to this query using conventional search engines one would have to try various query formulations and spend a lot of time scanning

Literary and Linguistic Computing, Vol. 22, No. 3, 2007.

doi:10.1093/llc/fqm010 Advance Access Published on 20 April 2007

through numerous retrieved pages. An alternative solution is to look up web link collections on the topic, such as the ACL/NLP Universe (ACL/NLP Universe, 2004), the Linguist List (Linguist List, 2004), or the Language Technology World (LT World, 2004). Unfortunately, as they are compiled and maintained manually, these collections quickly fall out of date and have limited coverage. A comparison of the links mentioned on the LT World website (208 links in total) with those on the Linguist List (122 links in total) shows that there is a maximum overlap between them of less than 11%, which suggests that neither can be taken to be a comprehensive reference on the topic.

It is obvious that one cannot search complex data patterns from the ocean of the web in a realistic time frame without technologies that take over at least some of the work on the intelligent processing of documents. Information Extraction (IE) is an area of research that aims to perform such intelligent analysis of the contents of documents. The goal of an IE system is to extract text fragments instantiating predefined semantic entities that can be mapped to fields in a database and later easily manipulated using database queries. An example of the IE task is the extraction of facts related to business transactions (the object of a transaction, participants, money involved, etc.) from business newswire texts.

The purpose of our work on the BiRD project (BiRD, 2007) is to design an Information Extraction system which will mine the web for pages on language resources, extract relevant facts from them, and produce a publicly accessible database containing this information. The specific problem we address in this article is information extraction from heterogeneous documents, such as web pages discovered at diverse websites by a domain crawler. Although this appears to be a very common IE application, previous work primarily concerned documents that have highly consistent structure such as pages from a specific web portal. In this article, we present our experience of developing a IE system for web pages obtained from the unrestricted web. The main contributions of this research are an investigation into the integration of techniques for semantic annotation of free-format web pages into a single IE system and an extension of the double classification approach to IE to operate on this type of document.

The article is organized as follows. In Section 2, we describe the difficulties for IE presented by pages from the unrestricted web. In Section 3, we give an overview of the system and describe its components developed in order to prepare documents for subsequent processing in the Information Extraction stage. Section 4 is devoted to the information extraction component, describing the double classification method it implements, and proposing a new technique for the flexible delineation of template fillers among candidates output by the method. Section 5 describes the settings for the experimental evaluation of the system, while Section 6 presents results and discussion emerging from the experiments. Finally, in Section 7 conclusions are drawn.

2 Heterogeneous Documents

Most of the related work on IE has been concerned with filling multi-field templates from consistently structured web pages and from complete grammatical sentences. Reliable performance has been achieved by systems that operate on web pages with regular layout, such as those providing a web interface to a database (e.g. Doorenbos *et al.*, 1997; Kushmerick *et al.*, 1997; Cohen *et al.*, 2002). These techniques rely on the analysis of the HTML structure of pages to learn how to locate relevant strings of text in them. Considerable progress has also been made in developing methods for extracting information from grammatical text (e.g. Soderland 1999; Roth and Yih, 2002; Zelenko *et al.*, 2003; Cullota *et al.*, 2006; Shinyama and Sekine, 2006). The cues these methods use are typically the output of different language processing steps such as PoS tagging and full or partial syntactic parsing. Unfortunately, in the context of an unrestricted web application, one cannot make the assumption that relevant entities and relations between them are expressed either through consistent page layout or specific linguistic constructs.

IE approaches that do operate on the unrestricted web have mostly been concerned with 'fact extraction', i.e. the filling of single-field templates

through the discovery of binary relations between entities (Cimiano *et al.*, 2004; Etzioni *et al.*, 2004; Shinzato and Torisawa, 2004). This work has been extended to perform multi-field IE by (Mann and Yarowsky, 2005), who used these techniques first to extract several isolated facts about an entity, possibly from different documents, and then to aggregate them into one multi-field template. However, their method extracts binary relations based on their multiple instantiations found on the web, rather than within a single document, which makes them unsuitable for the one template per document scenario.

The challenges posed by 'free-range' web pages for an IE system can be summarized as follows:

2.1 Document layout

As numerous studies have shown (Doorenbos *et al.*, 1997; Kushmerick *et al.*, 1997; Etzioni *et al.*, 2004; Schneider, 2005), the layout of a document can constitute very valuable evidence for information extraction. However, not only is there hardly any consistency in the layout of web pages retrieved by a domain crawler, but also one can seldom rely on even general formatting cues occurring on the page.

2.2 Discourse structure

Different documents about the same kind of event or entity tend to contain the same kind of information about them (such as the previous employment, contact details, or names of referees in a CV). This information may even be arranged into similar sections, but the linear order in which it is presented and its place in the overall discourse structure of the document can vary considerably from one document to another.

2.3 Free and semi-structured text

Web pages may contain grammatical text (natural language sentences) interspersed with graphical means of presenting information (tables, itemization lists, attribute-value lists). Correspondingly, fragments to be extracted need to be sought in both types of page content.

2.4 Vocabulary

Although the documents may be describing very similar resources, they may do so using very different styles of writing, which in turn influences the choice of vocabulary used. For example, the description of commercial software will be very different from the description of software that is distributed for free.

2.5 Format of extractable information

Because of the abundance of presentation styles that can be found in a crawler's output, the format in which extractable information appears is highly variable so that the boundaries of some entities may not be easily delimited. For example, the licensing information for a software product can be given as a number (the price), a word ('*shareware*'), a phrase ('*free of charge*'), or a sentence ('*The software is distributed free of charge for academic institutions*').

3 Document Preprocessing

To enrich web page content with annotation that is potentially useful for IE, we employ a range of techniques for page layout recognition and normalization, language identification, linguistic processing, named entity and terminology recognition, and coreference resolution. While most of them have been previously used in different NLP applications, our specific goal is to adapt them to the task at hand, ensuring their interoperability in order to maximize both the effectiveness and the efficiency of the system as a whole.

The preparation of web pages for the IE task is carried out by means of a series of processing units that implement these functionalities (see Fig. 1 for an overview of the system). After a web page is supplied to the system, it is passed through the processing units sequentially, with each unit placing XML mark-up into the page. The preprocessing procedure is pipelined in such a way that each unit benefits from the annotations produced by the previous one. In this way, at the end of the process, the page has received normalized structural, linguistic, and semantic annotations, which are later used for training and application of the IE unit.

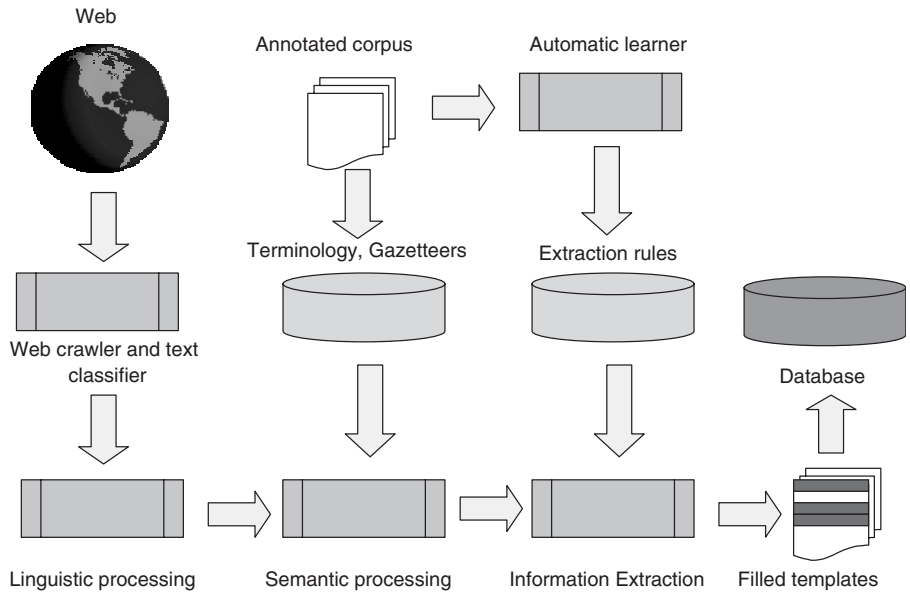

Fig. 1 The overview of the IE system

3.1 Collection of documents

To obtain relevant documents from the web, our system incorporates a web crawler, whose operation is based on the assumption that pages describing a specific type of entity can be best discovered on link collections on the topic (the so-called 'hubs'), rather than through key word search. Starting with a few seed hubs, the crawler seeks out more hubs on the topic using link analysis and selectively downloads pages mentioned in the discovered hubs. The algorithm performs these steps iteratively, at each step recomputing relevance scores for hubs and pages and enlarging both sets of URLs. The algorithm terminates when a certain stopping criterion has been reached (e.g. after a fixed number of iterations or search engine queries has been made).

Downloaded pages are filtered for size and duplicates among them are detected and discarded. After that irrelevant pages are filtered out using the Rainbow text classification toolkit,[1] which has been trained on a set of manually labeled documents.

3.2 Linguistic processing

A web page often contains descriptions of the same entity in more than one language. To enable further linguistic processing, non-English segments of pages are identified by means of a character-based trigram model of the language identity of the text (Grefenstette, 1995). XML tags are then placed around these non-English segments.

PoS tagging and phrase chunking of the English text in the documents are carried out using the LT CHUNKER (Mikheev, 1996), which operates over XML documents and places linguistic information as another XML layer. The Porter stemmer is used to derive the stems of the words.

3.3 Acquisition of domain terminology and gazetteers

This stage is responsible for the acquisition of the most important terms and named entities (person, location, and organization names, names of conferences, etc.) in the domain and their attribution to a set of semantic classes. The acquisition of the resources is carried out using

a domain corpus, which can be any corpus representative of the domain, possibly obtained automatically from the web.

The terminology acquisition unit employs methods similar to those developed by Juteson and Katz (1996) that search for the most frequent words and word sequences in the corpus matching patterns of PoS tags indicative of domain terms, such as (*Adj|Noun*)***Noun*. The unit also searches for expanded acronyms and abbreviations and uses them to delineate multi-word terms and identify their variants.

In order to recognize the semantic types of terms, the terminology acquisition unit applies an additional set of rules firing on certain constituents of the terms, their orthography and PoS tags. For example, if the last noun in an extracted sequence matching (*Adj|Noun*)***Noun* is a word denoting organization (e.g. 'department'), the term is ascribed to the semantic class ORGANIZATION. To ensure their quality, the created terminology and gazetteer lists are revised by an expert.

3.4 Recognition of page layout

To recognize page formatting that may be useful for the IE step, we introduce XML tags to mark up attribute-value lists and the so-called personal data blocks. Attribute-value lists are recognized by detecting several consecutive lines which are significantly shorter than the average line on the page and contain a colon, a tabulation character, or multiple white spaces. The text preceding the separating element is taken to be the name of the attribute and the text following it as the value for that attribute, and both strings are marked up accordingly.

Personal data blocks are several short lines containing a person's name, postal address, telephone and fax number, email address, and URL of their home web page. Personal data blocks are identified using mark-up rules based on HTML tags and generic named entity tags supplied by the ANNIE system (Section 3.5).

After this stage, all web pages are processed using HTML *Tidy*,[2] which verifies and corrects their HTML structure.

3.5 Named entity and terminology recognition

Named Entity Recognition (NER) is performed in two steps. In the first step, common named entities (person names, locations, and dates) are recognized by means of the ANNIE system, a part of the GATE architecture (Cunningham *et al.*, 2002). The output from ANNIE is customized to a specific application domain by applying a number of postprocessing procedures. Domain-specific terms and NEs are recognized using domain gazetteers and terminology lists (Section 3.3).

The NER techniques used so far produce NE and terminology annotations with high precision, but they have poor coverage, especially in the case of domain-specific items that are often missing from the gazetteers. At the second step, we make use of the available NE annotations and layout cues to leverage mark-up of NEs and terms that remain unmarked to this point.

For this purpose we employ two methods, both of which are general-purpose. The first one is a list extraction technique akin to the one used in Etzioni *et al*, (2004). It consists of (1) locating itemization lists in the page, that contain a sufficient number of NEs and terms marked-up in previous stages, the NEs are used as seed data (Fig. 2); (2) inducing a list-specific 'wrapper' from the seeds and the regular layout and punctuation of list items; and (3) applying the wrapper to mark up remaining terms and NEs in the list.

The second technique is similar to the first, but is applied to grammatical text to identify enumerations. It finds sentences which contain enumerated expressions, each of which is separated by the same punctuation symbol, and possibly linked by a conjunction such as 'and', 'or', or 'as well as' at the end. e.g.

. . .<APPLICATION> *summarization* </APPLICATION>, <APPLICATION> *information retrieval*</APPLICATION>, *or online foreign language tutoring*. . .

Assuming that enumerations comprise expressions with similar semantics, all of the expressions delimitable by means of commas and conjunctions are marked-up by the same tag as the seeds.

```
Candace Kamm, <ORGANIZATION>AT&D</ORGANIZATION>
Lin-Shan Lee, <ORGANIZATION>Taiwan University</ORGANIZATION>
<PERSON>Susann Luperfoy</PERSON>, Akamai Technologies
<PERSON>Patti Price</PERSON>, SRI International
<PERSON>Owen Rambow</PERSON>, <ORGANIZATION>AT&D</ORGANIZATION>
```

Fig. 2 An itemization list with seed named entities marked-up

3.6 Coreference resolution

In order to detect coreference chains involving NEs of interest, we use the light-weight techniques for NE coreference resolution introduced in Bontcheva *et al.* (2002), which detect coreference based on orthographic cues, such as those capturing different variants of a person's name (e.g. *John Brown, J.Brown, and Mr. Brown*) and regular ways to create an acronym for an organization (e.g. *UN* for *United Nations*).

Common noun phrases are introduced into coreferential chains using the same precompiled vocabulary for each type of NE as was used for the recognition of the semantic class of the automatically acquired terms (Section 3.3): noun phrases that appear in the context of a term and that match the semantic label of the term are taken to be coreferent with the term. Prior to processing the document by means of the IE module, all references in a coreferential chain are substituted for the normalized variant of the entity.

4 Information Extraction

Previous research has developed a range of methods based on machine learning techniques for IE tasks that require the completion of one template per document. These provide the flexibility to be deployed for documents with inconsistent layout or discourse structure (e.g. Freitag, 1998; Califf and Mooney, 1998; McCallum *et al.*, 2000; Chieu and Ng, 2002). In this work, we opt for the double classification approach to IE, proposed by De Sitter and Daelemans (2003).

The method is inspired by the observation that when humans need to extract facts from a document they scan it quickly and only read closely those parts that look most relevant. In a similar manner, the double classification method uses two automatic classifiers: the first classifier identifies document fragments that are likely to contain template fillers, while the second classifies tokens inside the promising fragments to more precisely pinpoint the filler. The study by De Sitter and Daelemans (2003) has shown that this is a promising approach to pursue. It has demonstrated that the method is both more efficient and more effective than methods that extract template fillers by examining only the immediate context of each token in the text.

In this article, we examine the idea that the effectiveness of the method can be increased by optimizing the classification problems for the two classifiers relative to the domain in which the method is applied. We then propose a new method for the identification of multi-token fillers in the output of the double classification method.

4.1 Double classification

The IE procedure performed by the double classification method can be formally described as follows. Suppose we have a corpus manually annotated in terms of a predefined IE template, i.e. certain text tokens (words and punctuation) have a tag signifying that they instantiate a field of the template. The corpus is randomly divided into a training set of documents D_{TR} and a test set of documents D_{TS}. The first step is to split documents in D_{TR} and D_{TS} into sets of document fragments (say, sentences or paragraphs) F_{TR} and F_{TS}, respectively.

At each classification stage, n binary classifiers are built, one for each template field s_i. At the fragment

classification stage, classifier C_1 is learned from F_{TR}. Positive instances are fragments that contain at least one token annotated as s_i. Features for representing an instance are all tokens found inside the fragment. C_1 is evaluated on F_{TS}. Its output is F_{out}, the set of fragments that it has labeled as positive.

At the second stage, classifier C_2 is learned only from those fragments in F_{TR} that contain tokens annotated as a filler for s_i. Positive instances here are tokens annotated as s_i, negative ones are those that do not have any tags or have tags other than s_i. Features are tags and tokens appearing within a certain window around the token. C_2 is evaluated on tokens contained in F_{out}. The output from C_2 is a list of tokens W_{out}, which it has labeled as positive instances of s_i.

In computing evaluation metrics for the method as a whole, true positives are those tokens in W_{out} that have been manually annotated as positive, false positives are tokens in W_{out} that have not been manually annotated as positive, and false negatives are tokens that have been manually annotated as positive, but did not make it into either F_{out} or W_{out}.

4.2 Adjusting the task difficulty

We hypothesize that the effectiveness of the method greatly depends on how the difficulty of the entire IE task (i.e. the search space in which template fillers should be found) is distributed between the two classifiers. We would like to find a balance that will avoid two kinds of situations leading to poor overall results:

(1) C_1 achieves high precision, but low recall. The search space for C_2 is greatly shrunk, but many relevant tokens are missing from it.
(2) C_1 achieves high recall, but low precision. Most of the relevant tokens do appear at the second stage, but there are also very many irrelevant ones and the search space for C_2 is too large.

In this article, we examine whether or not the factors described in Sections 4.2.1–4.2.3 can help in the discovery of this optimal balance.

4.2.1 *Thresholding*

The most suitable balance is not necessarily the most balanced precision/recall ratio at C_1. It may depend on the nature of the documents at hand, namely on how indicative of template fillers larger contexts of tokens (e.g. paragraphs) are in comparison to their local contexts (e.g. neighboring tokens). If, for example, it is easy to recognize fillers in their local contexts, but larger contexts are difficult to distinguish as relevant or irrelevant, one should be cautious about classifications at C_1 and aim to maximize C_1's recall in order to increase the number of promising tokens passed on to C_2.

Thresholding is a technique that allows one to boost either precision or recall by looking at the confidence score of the classifier, such as the class membership probability output by Naïve Bayes classifiers. Various thresholding techniques exist (e.g. Sebastiani, 2002). In this study, we prioritize recall by determining the average confidence score for false negatives on held-out data and during proper testing we retrieve all instances whose classification was made with a confidence score above that threshold. To increase precision, we find the average confidence score for false positives and during testing treat all instances below that threshold as negative even if the classifier assigns them the positive label.

4.2.2 *Fragment size*

The size of the fragments may have a strong effect on how many relevant and irrelevant tokens will be passed from C_1 to C_2. If a document is split into many smaller fragments, such as lines, classifying them will be more difficult, but this will allow for the greatest reduction of the search space for C_2. If instead one chooses larger fragments like paragraphs, relevant ones will be found with greater ease, but C_2 will suffer more from the unbalanced data problem. Again, the best solution is not necessarily to simply choose average sized fragments; the usefulness of the fragments depends on the specific characteristics of the documents.

It should be noted that thresholding and fragment size are orthogonal factors, so that their combination may maximize the desired effect and

at the same time compensate for one another's drawbacks.

4.2.3 *Selection of instances*

As in De Sitter and Daelemans' study, we consider whether or not the problem of too many negative instances can be alleviated by performing instance selection. In order to build a more effective classification model, we train it on data from which some negative instances have been removed so that a certain desired proportion between negative and positive instances is achieved. After the model is learned from the balanced training data, it is evaluated on the unbalanced test data.

In this study, we look at how instance selection interacts with the other two parameters. In particular, we wish to find out whether the problem of increased search space resulting from maximized recall or from using larger fragments can be remedied by performing instance selection.

4.3 Identifying the best filler

The double classification method tries to find all occurrences of a filler in the document. Obviously, this task is difficult and seldom error-free. In many situations, however, extracting all field instantiations is not necessary, since the template field has to be filled with one single filler. One possibility to perform this is to choose, out of all candidates, the one that the token-level classifier extracted with the greatest confidence. However, this approach will not be very helpful in the case when fillers consist of multiple tokens, as it may easily select one part of the filler, but miss out others.

We propose a new method to identify the best filler for a template field, which operates on the output of the token-level classifier, as shown in Fig. 3. In addition to the classifier confidence score, it incorporates information about whether the positively labeled tokens appear sequentially, the count of these sequences, and the use of general surface constraints on the appearance of the filler.

The algorithm (Fig. 4) consists of two major steps. The first step (lines 1–7) is to extract N, a set of all possible token ngrams from C_2's output and compute the initial score for each n. The ngrams are uninterrupted sequences of tokens labeled as positive instances of a template field. Those ngrams that consist of tokens bearing no content such as stopwords and punctuation are eliminated (line 4). Semantically empty tokens are also removed from the beginnings and ends of the ngrams (line 5). The initial score for n is computed as the sum of the weights of its occurrences, where the weight of each occurrence n_{occ} is the average classifier score C of its constituents (line 6).

```
...
18   authored   NIL      0.5
19   by         AUTHOR   0.5
20   John       AUTHOR   0.75
21   Doe        AUTHOR   0.5
22   .          NIL      0.5
...
44   writes     NIL      0.6
45   John       AUTHOR   0.35
46   .          NIL      0.45
...
72   John       NIL      0.3
73   Doe        AUTHOR   0.7
74   ,          NIL      0.9
...
```

Fig. 3 Example of the output from C_2

To illustrate with an example, consider the hypothetical output from the classifier in Fig. 3 (the first column shows the number of the token in the document, the second the token itself, the third the label assigned by the classifier, and the last the classifier confidence score). The algorithm will first extract the ngrams '*by John Doe*', '*John*', and '*Doe*'. Stripping stopwords and punctuation at the beginning and end of each of them, three ngrams will be obtained: *John Doe, John*, and *Doe*. Their initial scores will be computed as follows:

$$score\ (John\ Doe) = (0.75 + 0.5)/2 = 0.625$$
$$score\ (John) = 0.35$$
$$score\ (Doe) = 0.7$$

The second step (lines 8–13) is to add further weight to those ngrams, whose subsequences exist in N. Thus, the final score for *John Doe* will be

Data: a list of positively classified document tokens T, each attached with a classifier confidence score C
Result: a list of token ngrams ranked according to their relevance as template fillers

Extract a set of uninterrupted sequences S from T;
Create a set of token ngrams N by extracting all subsequences from each $s \in S$;
for *each* n *in* N **do**
  discard n, if it contains only punctuation/stopwords;
  remove punctuation/stopwords at the beginning and end of n;
  $score(n) = \frac{1}{length(n)} \sum_{i \in n} C(i)$;
end
for *each* n *in* N **do**
  in N, discover N' such that n' is a subsequence of n;
  for *each* n **do**
    $score(n)$ += $score(n') \times \frac{length(n')}{length(n)}$;
  end
end
Rank N according to *score*;

Fig. 4 The algorithm for identifying the best single filler per template field

increased by the initial scores of its subsequences *John* and *Doe* appearing as distinct ngrams, each weighted by the proportion of its length to the length of the greater ngram, i.e. by 0.35*0.5 + 0.7*0.5 = 0.525. In this way, the algorithm aims to further take into account those cases when only a part of a relevant ngram has been labeled positively by the classifier.

As it is reliant upon the frequency of the ngram, the method may have an important interaction with the particular thresholding and fragmentation methods used. Specifically, we hypothesize that the best filler identification method works best with double classification settings that achieve the greatest recall while maintaining high precision.

5 Evaluation Settings

5.1 Experimental task

The documents we would like to extract information from are web pages describing NLP resources including software (PoS taggers, parsers, various corpus tools) and data (evaluation corpora and datasets, frequency lists, gazetteers). The IE template consists of the following fields: NAME, CREATOR, AREA (application area), TGTLANG (target language), PLATFORM, PROG-LANG (programming language), and EMAIL (contact email). All the fields take single fillers, except TGTLANG and PLATFORM. Some slots are mandatory (e.g. NAME), while others are not (e.g. TGTLANG). Table 1 summarizes the experimental task. It should be emphasized that although some of the fields are filled by a closed class of words (e.g. PLATFORM), the IE method is a machine learning procedure that extracts fillers by examining only the context of tokens in the documents.

5.2 Data

To perform the experimental evaluation, we needed a gold standard that would contain web pages on the relevant topic, but taken from diverse sources and having heterogeneous formatting, vocabulary, and document structure characteristics. For this

Table 1 Semantic entities to be extracted, their description, and examples of corresponding fillers

Semantic entities	Description	Number of fillers	Example fillers
NAME	The name of the resource	Single	CLAWS PoS-tagger
AREA	The task/area of application	Multiple	PoS tagging
CREATOR	Person or institution who created the resource	Single	UCREL
LICENCE	Licensing information/price	Single	commercial
PLATFORM	Operating system	Multiple	UNIX
PROGLANG	The programming language	Single	–
TGTLANG	The target natural language	Multiple	English
URL	The URL of the original page	Single	http://www.comp.lancs.ac.uk/ucrel/claws/
CONTACT	The contact details of the creators/distributors	Single	http://www.comp.lancs.ac.uk/computing/...

purpose, 100 web pages were downloaded from the link collection on the topic at the *Language Technology World* web site.[3] Preprocessing was carried out in accordance with the procedures described in Section 3. The pages were then manually annotated in terms of the IE template previously described by two annotators. The agreement between the annotators measured 69%, when exact string matches are considered and 77% when partial matches are also considered (the weight for a match was the Normalized Edit Distance between the two partially matching strings).

5.3 Classification method

At C_1 each fragment was represented as a feature vector, where features correspond to the tokens found within it. All words were stemmed and stopwords and words appearing in less than five different fragments in the entire corpus were discarded.

At C_2, each token t is represented by the following features:

- `token`: the string corresponding to t;
- `tags`: all tags (layout, PoS, phrase tags) appended on t;
- `token_before`: the token directly before t;
- `tags_before`: the tags on the token before t;
- `token_after`: the token directly after t;
- `tags_after`: the tags on the token after t;
- `token_window`: the tokens appearing within the window of 2 around t; and
- `tags_window`: all the tags appended on the tokens within the context window.

In the experiments, we used the WEKA implementation of the multinomial Naïve Bayes learner (Witten and Frank, 2000). To assess the accuracy of classifications, we used 10-fold cross-validation, computing precision, recall, and *F*-measure for each field and then averaging the results.

6 Results and Discussion

6.1 Recognition of document layout, terminology and named entities

We first examined the influence of layout, terminology, and NE mark-up on the accuracy of information extraction. Fixing all the other parameters of the double classification method (using paragraphs as document fragments and performing no thresholding or instance selection), we looked at its performance when (1) neither layout nor NE and terminology recognition is performed; (2) when the documents are enriched with layout tags; and (3) when NE and terminology annotations are also added to the document. Table 2 presents the results of these experiments. The best results among the three methods are shown in bold.

We found that introducing layout tags into the documents prior to passing them to the IE module slightly increases the quality of classifications at both the fragment and token levels. The average improvement amounted to around +1%

Table 2 The effectiveness (*F*-measure) of the IE method resulting from (1) using neither layout nor NE tags, (2) with layout tags included, and (3) also including NE tags as features

	Layout − NEs and Terms −		Layout + NEs and Terms −		Layout + NEs and Terms +	
	Frags	**Tokens**	**Frags**	**Tokens**	**Frags**	**Tokens**
AREA	0.430	0.127	**0.433**	**0.131**	**0.450**	**0.140**
CONTACT EMAIL	0.440	0.476	**0.452**	**0.494**	0.436	0.472
CREATOR	0.408	0.303	**0.422**	**0.312**	**0.444**	0.293
LICENCE	0.322	0.209	**0.332**	0.209	**0.374**	**0.228**
NAME	0.637	0.244	**0.662**	**0.259**	**0.744**	**0.363**
PLATFORM	0.579	0.389	0.576	0.386	**0.638**	**0.394**
PROGLANG	0.383	0.189	0.384	0.187	**0.394**	**0.226**
TGTLANG	0.430	0.127	**0.433**	**0.131**	**0.450**	**0.140**
AVERAGE	0.384	0.226	**0.394**	**0.233**	**0.420**	**0.254**

at both levels. Some fields seem to profit more from the layout tags (NAME (+2.5% and +1.5%), TGTLANG (+3% and +1.5%)), while others exhibit no change in performance (e.g. PLATFORM and PROGLANG).

When further adding NE and term tags, the average figures rise further, and more perceptibly: by +2.6% at the fragment level and +2.1% at the token level. Again, certain fields profit more from them than others. The greatest improvements are observed for NAME (+8.2% and +10.4%), PLATFORM (+6.2% and +2%), and TGTLANG (+3.2% and +5.8%). Slight negative changes in performance were observed for EMAIL (−2.2% at the token level) and CREATOR (−1.9%). The different effect of the tags on the performance with respect to different fields can be most obviously explained by the relative usefulness of the tags for indicating the field fillers. For example, it seems that NAME benefits more from both the layout and NE tags because the name of the resource is typically emphasized in a certain way and also often contains domain terminology and NEs. Thus, the results generally suggest that both layout and NE tags are useful contributors to the accuracy of the IE method.

We now describe experiments on the optimization of the two classification problems for the IE method.

6.2 Fragmentation method

We experimented with four types of document fragments: Sections (*Sec*), Paragraphs (*Par*), Sentences (*Sent*), and Lines (*Lin*). Table 3 characterizes each type of fragment in terms of the average number of tokens contained in a fragment and the average number of fragments in a document.

Table 3 The average size of fragments and the number of fragments per document for the four fragmentation methods

	Sec	**Par**	**Sent**	**Lin**
Tokens per fragment	68.5	18.8	9.4	7.3
Fragments per document	15	54.2	101.6	142.1

Table 4 The accuracy of the fragment- and token-level classifiers resulting from each fragmentation method

	Fragment			**Token**		
	P	***R***	***F***	***P***	***R***	***F***
SEC	0.590	0.922	**0.720**	0.158	0.428	0.231
PAR	0.611	0.818	0.848	0.700	0.412	**0.555**
SENT	0.447	0.692	0.543	0.163	0.903	0.276
LINES	0.502	0.878	0.639	0.166	0.440	0.241

Table 4 describes the effectiveness of classifications at both levels resulting from the use of each fragmentation method (the best results across fragmentation methods are shown in bold); Table 5 characterizes the search space for each fragmentation method as the corresponding proportion of positive and negative instances at the token level.

We see that larger fragments do indeed result in an easier classification task: the highest effectiveness at the first stage is achieved for the *Sec* and *Par*

methods (0.72 and 0.94, respectively). Looking at the second stage, we notice that the proportion of positive instances for C_2 increases as the fragment size decreases (Table 5). This accounts for the fact that notwithstanding good performance at the first stage, the *Sec* method is often the worst when these fragments are taken as the source from which fillers are to be extracted. Although *Lin* has the greatest positive/negative ratio, it performs poorly compared other methods, obviously because of inaccurate classifications at the initial stage. *Par* exhibited the best overall performance at the second stage, outdoing *Sent* by a large margin.

6.3 Thresholding

We looked further at how maximizing recall or precision interacts with different fragment sizes. We would like to see whether thresholding can help to compensate for the weaknesses in a particular fragmentation method. Thus, we expect that low recall of small fragments which greatly reduces the search space for C_2 can be improved by increasing recall at C_1. This will increase the search space for C_2, but the increase might be smaller than the one resulting from simply using larger fragments. Table 6 describes the results of these runs at C_2. The figures shown in bold indicate improved performance in comparison with the runs without thresholding.

As will have been noted from Table 5, boosting recall at C_1 does increase C_2's search space somewhat, but for *Sent* and *Lin* it is still smaller than for *Sec* and *Par* without thresholding. This leads to an improvement in performance: both precision and recall rates frequently rose for *Sent*. In a similar fashion, boosting precision at C_1 reduces the search space for C_2, which often improves the accuracy for larger fragments such as *Sect*.

Table 5 Search space at C_2: the proportion of positive and negative training instances for different classification methods and thresholding settings

	No thresholding	Boosted *P*	Boosted *R*
SEC	0.02	0.03	0.01
PAR	0.05	0.06	0.04
SENT	0.08	0.08	0.07
LINES	0.09	0.1	0.08

Table 6 The effect of boosting precision versus recall at C_1 on the accuracy of C_2

	Boosted *P*			Boosted *R*		
	P	*R*	*F*	*P*	*R*	*F*
SEC	**0.208**	0.418	0.278	0.158	0.428	0.231
PAR	0.306	**0.854**	0.451	0.249	0.845	0.385
SENT	**0.166**	0.903	**0.280**	0.256	**0.948**	**0.403**
LINE	0.150	**0.376**	0.214	0.114	0.420	0.179

6.4 Instance selection

We examined instance selection techniques as an alternative way to reduce the unbalanced data problem at C_2. We looked at whether they are especially effective for situations when high recall at C_1 is achieved and/or whether they lead to further improvements in performance. Table 7 describes the results for different settings with instance selection applied. We see that instance selection invariably improves recall (results superior to the corresponding ones in Table 4 and 6 are presented in bold). However, this is sometimes achieved at the cost of a considerable decrease in precision (e.g. for *Lin* from 11.4%, Table 6, to 3.8%). We believe that the explanation for the increased recall and often decreased precision is the fact that the model is induced from data that contains a greater proportion of positive instances. This causes the

Table 7 The effect of instance selection on C_2

	No thresholding			Boosted *P*			Boosted *R*		
	P	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*
SEC	0.131	**0.889**	0.228	**0.287**	**0.755**	**0.416**	0.131	**0.889**	0.228
PARA	0.371	**0.859**	0.518	**0.384**	**0.859**	**0.531**	**0.278**	**0.923**	**0.427**
SENT	**0.270**	**0.942**	**0.420**	**0.276**	**0.941**	**0.427**	0.256	0.948	0.403
LINE	0.078	**0.951**	0.144	**0.194**	**0.852**	**0.316**	0.038	**0.983**	0.073

classification of a larger proportion of test instances as positive. When precision is boosted, however, additionally applying instance selection increased both precision and recall, yielding the best results overall thus far.

The use of binary classifiers gives one the opportunity to adjust the classification problems for each template field separately. Table 8 compares the results achieved for each field, using the most optimal setting for that field against the typical settings of the double classification method, i.e. using sentence fragments without performing any thresholding or instance selection. Performance scores improving on the standard settings appear in bold. The results indicate that fine-tuning the classification problem for each field separately offers a significant improvement over the traditional approach in terms of precision (by 20%) and *F*-measure (by 26%).

6.5 Identifying the best fillers

We evaluated the best filler identification algorithm against the performance of manually constructed IE rules that trigger the extraction of a particular field filler based on a variety of orthographic, linguistic, and page formatting cues. As a gold standard, we used the same data as in the previous experiments: a database was prepared by filling each template field for each document with the most frequent unique filler tagged by annotators in that document. The evaluation of both IE methods consisted of 10-fold cross-validation, at each fold both methods were evaluated on the same set of documents. The hand-crafted rules were prepared by two domain experts; the construction of the rules took four person-weeks in total.

We examined the effect of varying the parameters of the double classification method (the fragment size, thresholding, and instance selection) on the

Table 8 Comparison of accuracy using the best settings for each field against the typical parameter settings: *P* boosted in each case

	Settings		Best configuration			Typical		
	Frag	**Inst. sel.**	***P***	***R***	***F***	***P***	***R***	***F***
NAME	para	yes	**0.47**	**0.88**	**0.61**	0.277	0.811	0.413
AREA	para	yes	**0.31**	0.67	**0.418**	0.177	0.81	0.291
CREATOR	para	yes	**0.21**	0.95	**0.341**	0.073	0.964	0.136
PLATFORM	para	yes	**0.65**	0.92	**0.758**	0.288	1	0.447
PROGLANG	para	yes	**1**	0.6	**0.75**	0.115	1	0.206
TGTLANG	line	no	**0.24**	0.45	**0.317**	0.041	0.761	0.078
EMAIL	para	no	**0.36**	0.98	**0.524**	0.173	0.975	0.294
AVERAGE	–	–	**0.46**	**0.78**	**0.53**	0.163	0.903	0.276

Table 9 The F-scores achieved by the best filler identification algorithm and hand-crafted rules

	Settings		One-best filler selection	Hand-crafted rules
	Frag. method	**Inst. sel.**		
NAME*	paragraph	no	**0.527**	0.424
AREA*	sentence	yes	**0.705**	0.211
CREATOR	sentence	no	**0.639**	0.402
PLATFORM	sentence	yes	**1**	0.472
PROGLANG*	sentence	yes	**1**	0.443
TGTLANG	paragraph	no	**0.849**	0.16
EMAIL	paragraph	no	**0.276**	0.108
AVERAGE	–	–	**0.713**	0.317

*indicates boosted P, for the rest of the fields, thresholding is irrelevant.

performance of the best filler identification algorithm. Table 9 describes the results achieved with the most optimal parameter settings for each field. Performance scores improving on the standard settings are shown in bold. We found that the performance of the proposed algorithm was consistently superior to that of the hand-crafted rules, and often by a considerable margin (e.g. 68% for TGTLANG).

7 Conclusion

In this article, we discussed the problem of information extraction from highly heterogeneous documents, such as web pages located at diverse websites. The article explored our experience developing an IE system for this task, discussed problems that arise in this scenario and proposed their solutions at different levels of text processing: language identification, linguistic preprocessing, document layout recognition, recognition of named entities, coreference resolution and information extraction proper.

Our specific focus was the double classification approach to information extraction. The method is known to provide a convenient means to perform information extraction tasks where there is one template to be filled on the basis of an entire document. In this article, we reported a study of the impact of the proposed methods for document layout recognition and recognition of named entities and domain terms on the performance of the IE method. The results of the evaluation suggest that both layout, NE, and terminology tags are useful contributors to the accuracy of IE.

We further presented an investigation into a number of parameters of the IE method in order to optimize its two classification subproblems and eventually improve its overall performance. These experiments have shown that finding appropriate settings for the three factors influencing the distribution of the task difficulty between the two classifiers helps to improve performance. In particular, doing so improved *F*-measure by 26% in comparison with a method performing fragmentation of documents at sentence boundaries without applying thresholding or instance selection as was done in the original study presented in De Sitter and Daelemans (2003).

The double classification method aims to extract all tokens instantiating template fields, which is a very difficult and error-prone task. However, what is often needed instead is the accurate extraction of one single filler which may consist of a single token or a sequence of tokens. We have presented a new method for the identification of such fillers in the output of the double classification method. The proposed method takes advantage of the evidence for the best filler in the form of the relative position of tokens labeled as positive by the second classifier, the frequency of the token sequences, and the frequency of their component tokens. Our evaluation shows that the method, coupled with the double classification approach, performs consistently better than extraction rules constructed by hand.

Acknowledgements

This work was carried out within the research project 'An Automatic System for Resource Databases for Researchers' (ESRC grant RES-000-23-0010).

References

ACL/NLP Universe. (2004). *http://tangra.si.umich.edu/clair/universerk/html/u/db/acl/* (accessed 27 October 2004).

BiRD. (2007). http://clg.wlv.ac.uk/projects/bird/. (accessed 6 March 2007).

Bontcheva, K., Dimitrov, M., Maynard, D., Tablan, V., and Cunningham, H. (2002). *Shallow Methods for Named Entity Coreference Resolution, Chanes de Références et Résolveurs d'Anaphores, Workshop TALN 2002*, Nancy, France.

Califf, M. E. and Mooney, R. J. (1998). *Relational Learning of Pattern-Match Rules for Information Extraction, Working Notes of AAAI Spring Symposium on Applying Machine Learning to Discourse Processing*, Menlo Park, CA.

Chieu H. L. and Ng H. T. (2002). *A Maximum Entropy Approach to Information Extraction from*

Semi-Structured and Free Text, Proceedings of AAAI-02, Edmonton, Alberta.

Cimiano P., Pivk A., Schmidt-Thieme L., and Staab S. (2004). *Learning Taxonomic Relations from Heterogeneous sources, Proceedings of the ECAI 2004 Ontology Learning and Population Workshop, Valencia, Spain*.

Cohen, W., Hurst, M., and Jensen, L. S. (2002). *A Flexible Learning System for Wrapping Tables and Lists in HTML Documents, Proceedings of the Eleventh International World Wide Web Conference WWW-2002*, Honolulu, Hawaii.

Cullota, A., McCallum, A., and Belz, J. (2006). *Integrating Probabilistic Extraction Models and Data Mining to Discover Relations and Patterns in Text, Proceedings of HLT-NAACL'06*, New York, NY.

Cunningham, H., Maynard, D., Bontcheva, K., and Tablan V. (2002). *GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications, Proceedings of ACL-02*, Philadelphia, PA.

De Sitter, A. and Daelemans, W. (2003). *Information Extraction via Double Classification, Proceedings of the ECML/PKDD 2003, Workshop on Adaptive Text Extraction and Mining*, Cavtat-Dubrovnik, Croatia.

Doorenbos, R., Etzioni, O., and Weld, D. (1997). *A Scalable Comparison-Shopping Agent for the World-Wide Web, Proceedings of First International Conference on Autonomous Agents*, Marina Del Rey, CA.

Etzioni, O., Cafarella, M., Downey, D. *et al*. (2004). *Web-Scale Information Extraction in KnowItAll, Proceedings of the Thirteenth International World Wide Web Conference WWW-2004*, Manhattan, NY.

Freitag, D. (1998). *Information Extraction from HTML: Application of a General Learning Approach, Proceedings of AAAI-98*, Madison, Wisconsin.

Greffenstette, G. (1995). *Comparing Two Language Identification Schemes, Proceedings of the 3rd International Conference on the Statistical Analysis of Textual Data (JADT 95),* Rome, Italy.

Justeson, J. S. and Katz, S. L. (1996). Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, **3**(2): 259–289.

Kushmerick, N., Weld, D., and Doorenbos, R. (1997). *Wrapper Induction for Information Extraction, Proceedings of IJCAI-97*, Nagoya, Japan.

Linguist List (2004). http://linguistlist.org/ (accessed 27 October 2004).

LT World. (2004). http://lt-world.org/ (accessed 27 October 2004).

Mann, G. and Yarowsky D. (2005). *Multi-Field Information Extraction and Cross-Document Fusion, Proceedings of ACL-05*, Ann Arbor, MI.

McCallum, A., Freitag, D., and Pereira, F. (2000). *Maximum Entropy Markov Models for Information Extraction and Segmentation, Proceedings of the 17th International Conference on Machine Learning*, San Francisco, CA.

Mikheev, A. (1996). *LT CHUNK V 2.1. Language Technology Group*, University of Edinburgh, UK.

Roth, D. and Yih, W.-T. (2002). *Probabilistic Reasoning for Entity and Relation Recognition, Proceedings of the 19th International Conference on Computational Linguistics*, Taipei, Taiwan.

Schneider, K. M. (2005). *Information Extraction from Calls for Papers with Conditional Random Fields and Layout Features, Proceedings of AICS-05, Belfast, UK*.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, **34**(1): 1–47.

Shinyama, Y. and Sekine, S. (2006). *Preemptive Information Extraction using Unrestricted Relation Discovery, Proceedings of HLT-NAACL'06*. New York, NY.

Shinzato, K. and Torisawa, K. (2004). *Extracting Hyponyms of Pre-Specified Hypernyms from Itemizations and Headings in Web Documents, Proceedings of COLING'04*, Geneva, Switzerland.

Soderland, S. (1999). Learning Information Extraction Rules for Semi-Structured and Free Text. Machine Learning, **34**(1–3): 233–272.

Witten I. and Frank, E. (2000). *Data Mining - Practical Machine Learning Tools and Techniques with Java Implementations*. San Francisco: Morgan-Kaufman.

Zelenko D., Aone, C. and Richardella, A. (2003). Kernel Methods for Relation Extraction. *Journal of Machine Learning Research*. **3**: 1083–1106.

Notes

1 www-2.cs.cmu.edu/ mccallum/bow/rainbow/
2 http://tidy.sourceforge.net
3 http://www.lt-world.org