Information Extraction Framework

Hassan A. Sleiman and Rafael Corchuelo

Abstract. The literature provides many techniques to infer rules that can be used to configure web information extractors. Unfortunately, these techniques have been developed independently, which makes it very difficult to compare the results: there is not even a collection of datasets on which these techniques can be assessed. Furthermore, there is not a common infrastructure to implement these techniques, which makes implementing them costly. In this paper, we propose a framework that helps software engineers implement their techniques and compare the results. Having such a framework allows comparing techniques side by side and our experiments prove that it helps reduce development costs.

Keywords: Information Extraction Framework Architecture.

1 Introduction

The Web contains a huge amount of information and is a still growing data container. This unlimited repository aroused enterprises' interests in exploiting web information, so new applications that consume and analyse this information have emerged. Applications such as businesses intelligence and other marketing tools need data from the Web to help users making decisions and offer best service. Unfortunately, the information in the Web is embedded in HTML tags and in other contents that in many cases are not interesting. Information extraction is used in these cases to obtain the information in which user is interested and to discard the other [6].

Information extraction from the Web is the task that extracts relevant information from web pages, where relevant is relative to the use case and the user's intentions. During the last decades, many proposals on information extractors have been introduced, but the web has changed and many of these proposals are not useful anymore.

Hassan A. Sleiman · Rafael Corchuelo
University of Sevilla
e-mail: {hassansleiman,corchu}@us.es

J.M.C. Rodríguez et al. (Eds.): Trends in PAAMS, AISC 157, pp. 149–156.
springerlink.com

According to a recent report [4], developing and maintaining information extractors is still a tedious process because of the lack of development support tools. Comparing information extractors are usually compared by their concepts, such as the surveys [2] and [11]. Empirical comparisons are still tedious since they require the development of other proposals and need to be performed under coherent conditions.

Relying on a framework in the domain of information extractors has many benefits. Using a framework in developing proposals reduces development and testing costs. If the framework is accompanied by a collection of datasets, then it shall also help compare different techniques homogeneously.

In this proposal, we present an overview of the framework architecture which has been validated by developing several techniques. The time needed to develop some techniques using our framework is compared to the time that was necessary to develop the same techniques without the framework to show the costs reduction. Furthermore, cross validation is used to test proposals to obtain comparable precision and recall between the different techniques.

This paper is organised as follows: First, Section 2 classifies and lists related work briefly and then, Section 3 describes the architecture of the framework. The framework is then used to develop a set of proposals and the experimental results are reported in Section 4. We conclude our work in Section 5.

2 Related Work

The high number of proposals on information extraction makes us classify them according to some criteria to detect the approaches for which our framework shall adapt. Information extractors can be used to extract and structure information from free text web pages, such as news and blogs, or from result and detail web pages, such as web pages with one or more result records and detail web pages with information about a certain product. Our work focuses on the second type of information extractors used for semi-structured web pages.

Information extractors for semi-structured web pages can be classified into two groups: a heuristic based group and a rule based group. The heuristic based group contains proposals that are based on predefined heuristics. Although these heuristics can be seen as rules, the difference between this group and the rule based group is that they are totally independent from the web page on which they are applied. These heuristics are not modifiable and are not inferred by an automatic technique. Techniques like Stavies [13], Alvarez et al. [1], and ViPER [14] are heuristic based.

The rule based group contains information extractors that are configurable by means of rules. Beyond handcrafted information extraction rules, there are many proposals in the literature to learn them in a supervised and in an unsupervised manner. Supervised techniques require user intervention to learn these rules. Generally, it needs the user to annotate a set of web pages by selecting and assigning a type for the relevant information in a set of web pages used then in the learning process. Techniques like Softmealy [7], WIEN [9], Stalker [12], and DEByE [10] are supervised techniques. This latter group also contains a number of unsupervised techniques, which do not requiere the user to provide annotations. Input web pages are analysed to detect repeating patterns or templates used at the server side to generate these pages. Techniques like FiVaTech [8], RoadRunner [5], DEPTA [16], and DeLa [15] belong to this group.

3 Architecture of Our Framework

The framework is composed of six packages. These packages can be used during the rule learning process for the rule based information extractors or in testing extraction results for all the types of information extractors described in the previous section. These packages are Dataset, Annotator, Learners, Tokeniser, Cross Validator, and Utilities, which are explained in the following subsections:

3.1 Datasets

This package, contains all the information annotated by user during the annotation process or extracted by an information extractor during an extraction process, see Figure 1. Package classes are described below:

- Dataset: This is a map-like structure that associates a set of annotations to a number of web pages. These annotations represent the relevant data in this web page and are represented by a Resultset.
- Resultset: A class that contains the annotations that mark the relevant information in a web page. Each annotation has its description besides the relations between these annotations.
- Locators are pointers to each annotated fragment in a web page. They are of two types: TreeLocators which contain an XPath that points to annotation's node or TextLocators which points to the beginning offset and the length of annotation in the web page.
- Views can be created for a web page: a TextView offers working with the text contained inside a web page and a TreeView which can be used to work with the HTML tree and its nodes.

3.2 Annotator

A tool that helps users download and annotate web pages to create Datasets. First, the user shall select an ontology which is used to assign a type and a relation between the annotations. Then, he or she navigate to web pages and add them to the created Dataset. Once added, contents from this web page can be selected, dragged and dropped into the ontology. This allows the creation of individuals of a certain class and assigning properties to them, saving their locators too. The tool also checks and warns for possible errors during the annotation process. Datasets can be then loaded and modified in the tool, or used in the framework for learning and testing tasks.

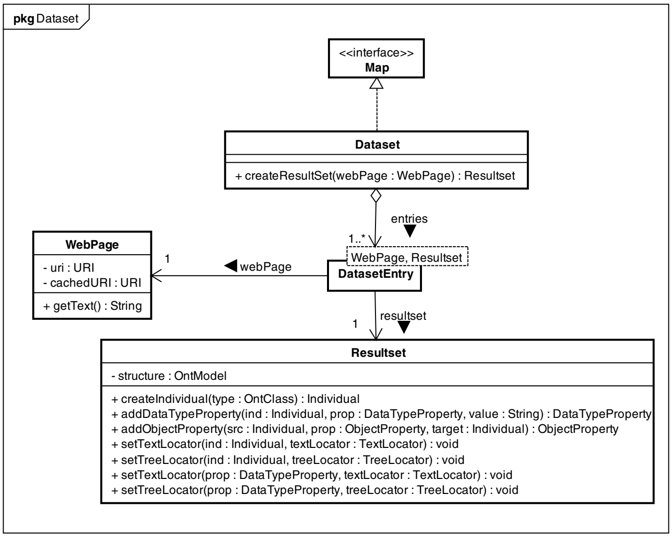
pkg Dataset
<<interface>>
Map
Dataset
+ createResultSet(webPage : WebPage) : Resultset
entries
1..*
WebPage, Resultset
DatasetEntry
webPage
1
WebPage
- uri : URI
- cachedURI : URI
+ getText() : String
resultset
1
Resultset
- structure : OntModel
+ createIndividual(type : OntClass) : Individual
+ addDataTypeProperty(ind : Individual, prop : DataTypeProperty, value : String) : DataTypeProperty
+ addObjectProperty(src : Individual, prop : ObjectProperty, target : Individual) : ObjectProperty
+ setTextLocator(ind : Individual, textLocator : TextLocator) : void
+ setTreeLocator(ind : Individual, treeLocator : TreeLocator) : void
+ setTextLocator(prop : DataTypeProperty, textLocator : TextLocator) : void
+ setTreeLocator(prop : DataTypeProperty, treeLocator : TreeLocator) : void

Fig. 1 Dataset Package

3.3 Learners

The Learners package provides an interface implemented by different techniques and other other classes used during rule learning process. Package classes are described briefly next:

- Learner: It is an interface that provides a number of template methods software engineers must provide to implement their techniques.
- SkeletonLearner: This class transforms a set of annotations into a Transducer in which each state identifies a type of data to be extracted. The transitions between them maintain the separators found in the input web page, which can later be used by the learning algorithms to learn transition rules.
- Transducer: A class that models state machines that can be learnt incrementally and executed to extract information from a web page. Thanks to the SkeletonLearner class, software engineers need only focus on learning the transitions.

3.4 Tokeniser

This is a configurable tokeniser that allows users to define a hierarchy between types of tokens, generalising and specialising them during the learning process, see Figure 2. The main classes of this package are described briefly here:

- TokeniserConfig: A class that helps read the XML configuration file where classes and hierarchy are declared and create the token classes. It maintains the defined structure to be used during the learning process.

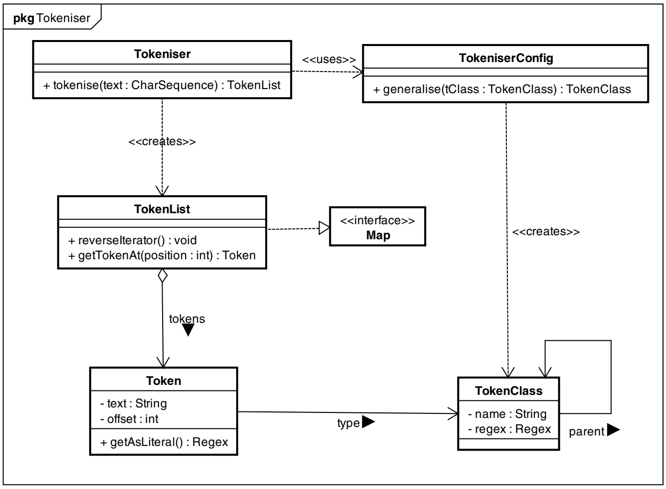

Fig. 2 Tokeniser

- TokenClass: This class refers to a type of token. Token classes are defined in the tokeniser's configuration file where a name and a regular expression for each TokenClass is defined.
- Tokeniser: A class that tokenises input text by searching for matchings of the defined token classes on this text and returning a TokenList.
- TokenList: It is a map that saves the tokens according to their position. It allows to sort and search for specific tokens very efficiently.
- Token: This class is created by the tokeniser by assigning a TokenClass to it when it matches the regular expression of a TokenClass. It can be converted into a regular expression in case of specialisation.

3.5 Cross Validator

Our framework provides a cross validation package to evaluate information extractors with the following classes:

- TestUtilites: This class is used to compare extraction results with an annotated dataset. It calculates precision, recall, F-measure, accuracy, specificity and sensitivity for each type of data in these datasets.
- CrossValidation: Implementes a *K* cross-validator, where k is typically 10.
- Stats: This class collects information during the cross validation process. At the end of the cross validation process, this class calculates statistical information about each one of the parameters, e.g., Precision and Recall.

3.6 Utilities

This package includes a set of classes used in more than one extraction rules learning proposal. Some of these utility classes are:

- StringAligner: This class implementes a string alignment algorithm inspired by FiVaTech [8] that aligns a set of input sequences of tokens and returns a unique aligned sequence.
- PatternDetector: It uses a FiVaTech [8] similar algorithm to detect pattern in an input sequence. The result is a regular expression that represents fixed,repeated and optional elements.
- PatriciaTree: It creates a Patricia tree from a set of token sequences and builds a regular expression from this tree. This is used in proposals such as DeLa [15] and IEPAD [3].

4 Experimental Results

To validate our framework, we have implemented a number of proposals in the literature. We provide a toolkit with the following learners NLR, SM and FT which are inspired by [9], [7] and [8] respectively. The time necessary for their development using our framework is compared to the development time that was necessary without using our framework. We have also compared their performance regarding precision and recall on a homogenous collection of datasets.

Table 1 shows the time that was necessary to develop and test the techniques in the first column. The second and the third columns show the time that was necessary to develop these techniques without using the framework and using it. The costs reduction is clear since the framework allowed reusing components and the same datasets were used in every implementation. The last column shows the reduced time percentage, the arithmetic mean of the reduction percentage is 57.51%.

Table 1 Comparing implementation times for NLR, SM, and FT

Technique	Without Framework	Using Framework	Reduced time percentage
NLR	145hrs	32hrs	77.94%
SM	123hrs	87hrs	29.27%
FT	176hrs	61hrs	65.34%

Table 2 reports the results of applying these techniques in practice on several datasets compared side by side. The first column contains the used datasets. Other columns contain precision (P) and recall (R), besides the time that was necessary to learn extraction rules by each technique for each dataset. Each dataset contains 30 web pages, and the results regarding precision and recall were calculated using 10-fold cross validation.

Table 2 Comparing precision and recall of NLR, SM, and FT techniques

Dataset	NLR			SM			FT		
	P	R	T(s)	P	R	T(s)	P	R	T(s)
doctor.webmd.com	0.62	0.62	989.78	0.98	0.95	11.14	0.83	0.61	4.29
extapps.ama-assn.org	0.61	0.61	384.84	0.79	0.38	4.46	0.70	0.58	3.65
www.dentists.com	1.00	1.00	18.82	0.64	0.62	2.32	1.00	0.30	1.84
www.drscore.com	0.80	0.06	14.25	1.00	0.86	4.87	0.81	0.05	3.31
www.steadyhealth.com	0.75	0.72	265.75	1.00	0.96	11.68	1.00	0.78	6.21
classiccarsforsale.co.uk	0.49	0.38	39.14	1.00	0.80	11.89	0.96	0.23	7.62
internetautoguide.com	0.30	0.21	85.23	0.30	0.21	11.68	0.91	0.67	5.87
www.autotrader.com	0.88	0.70	130.15	1.00	0.93	18.34	0.88	0.22	10.59
www.carmax.com	0.84	0.81	39.82	0.99	0.90	8.85	0.89	0.80	5.67
www.carzone.ie	0.84	0.81	37.85	0.99	0.67	6.90	0.98	0.66	4.64

Note that these techniques can obtain better precision and recall by adding more web pages to these datasets, but this is not our case since we are just obtaining comparable results which allows us selecting the extraction rules learning algorithm that best fits the web site we are interested in.

5 Conclusions

In this paper we have described our information extraction framework. We also reported our first experimental results which confirms the fact that using the framework reduces costs and allows side by side comparison providing comparable results. Development costs were reduced 57.51%. Future proposals that use our framework and our datasets can compare their result with the obtained results here without having to implement these techniques again neither annotate the same web pages.

Acknowledgements. Supported by the European Commission (FEDER), the Spanish and the Andalusian R&D&I programmes (grants grants TIN2010-21744-C02-01, TIN2007-64119, P07-TIC-2602, P08-TIC-4100, TIN2008-04718-E, and TIN2010-09988-E).

References

[1] Álvarez, M., et al.: Extracting lists of data records from semi-structured web pages. Data Knowl. Eng. 64(2) (2008)

[2] Chang, C.-H., et al.: A survey of web information extraction systems. IEEE Trans. Knowl. Data Eng. 18(10) (2006)

[3] Chang, C.-H., Lui, S.-C.: IEPAD: information extraction based on pattern discovery. In: WWW (2001)

[4] Chiticariu, L., et al.: Enterprise information extraction: recent developments and open challenges. In: SIGMOD Conference (2010)

[5] Crescenzi, V., et al.: Roadrunner: Towards automatic data extraction from large web sites. In: VLDB (2001)

[6] de Viana, I.F., Hernandez, I., Jiménez, P., Rivero, C.R., Sleiman, H.A.: Integrating Deep-Web Information Sources. In: Demazeau, Y., Dignum, F., Corchado, J.M., Bajo, J., Corchuelo, R., Corchado, E., Fernández-Riverola, F., Julián, V.J., Pawlewski, P., Campbell, A. (eds.) Trends in PAAMS. AISC, vol. 71, pp. 311–320. Springer, Heidelberg (2010)

[7] Hsu, C.-N., Dung, M.-T.: Generating finite-state transducers for semi-structured data extraction from the web. Inf. Syst. 23(8) (1998)

[8] Kayed, M., Chang, C.-H.: FiVaTech: Page-level web data extraction from template pages. IEEE Trans. Knowl. Data Eng. (2010)

[9] Kushmerick, N.: et al. Wrapper induction: Efficiency and expressiveness. Artif. Intell. 118(1-2) (2000)

[10] Laender, A.H.F., et al.: DEByE - data extraction by example. Data Knowl. Eng. 40(2) (2002)

[11] Muslea, I., et al.: Extraction patterns for information extraction tasks: A survey. In: AAAI-1999 Workshop on Machine Learning for IE (1999)

[12] Muslea, I., et al.: Hierarchical wrapper induction for semistructured information sources. Autonomous Agents and Multi-Agent Systems 4(1/2) (2001)

[13] Papadakis, N., et al.: Stavies: A system for information extraction from unknown web data sources through automatic web wrapper generation using clustering techniques. IEEE Trans. Knowl. Data Eng. 17(12) (2005)

[14] Simon, K., Lausen, G.: ViPER: augmenting automatic information extraction with visual perceptions. In: International Conference on Information and Knowledge Management (2005)

[15] Wang, J., Lochovsky, F.H.: Data extraction and label assignment for web databases. In: WWW (2003)

[16] Zhai, Y., Liu, B.: Structured data extraction from the Web based on partial tree alignment. IEEE Trans. Knowl. Data Eng. 18(12) (2006)