Knowledge-Based Systems 93 (2016) 84–108

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

ARIEX: Automated ranking of information extractors

Patricia Jiménez [a,*], Rafael Corchuelo [a], Hassan A. Sleiman [b]

[a] Universidad de Sevilla, ETSI Informática, Avda. de la Reina Mercedes, s/n. Sevilla E-41012, Spain
[b] Commissariat à l'Énergie Atomique et aux Énergies Alternatives LIST, LADIS, Digiteo Labs Saclay, 91191 CEDEX, Gif Sur Yvette, France

ARTICLE INFO

Article history:
Received 16 April 2015
Revised 3 November 2015
Accepted 5 November 2015
Available online 30 November 2015

Keywords:
Web documents
Information extraction
Ranking method
Automatisation

ABSTRACT

Information extractors are used to transform the user-friendly information in a web document into structured information that can be used to feed a knowledge-based system. Researchers are interested in ranking them to find out which one performs the best. Unfortunately, many rankings in the literature are deficient. There are a number of formal methods to rank information extractors, but they also have many problems and have not reached widespread popularity. In this article, we present ARIEX, which is an automated method to rank web information extraction proposals. It does not have any of the problems that we have identified in the literature. Our proposal shall definitely help authors make sure that they have advanced the state of the art not only conceptually, but from an empirical point of view; it shall also help practitioners make informed decisions on which proposal is the most adequate for a particular problem.

1. Introduction

A web information extractor works on user-friendly web documents that have been typically gathered using a crawler [1]. They analyse the documents and extract the information that they provide in a structured format that can be used to feed knowledge-based systems. The information is commonly structured into attributes and records, to which we collectively refer to as slots.

Fig. 1 illustrates what web information extraction is about. It shows an excerpt of a sample web document that provides a listing of records regarding phones. The document is rendered in a friendly format that a person can easily understand. The problem is that the information in this document is not structured, which means that it is not easy to use it in an automated process. Information extractors are devised to help in this task, since they can transform the web document on the left into the structured information on the right. Formally speaking, an information extractor can be modelled as a function that maps DOM nodes or text fragments onto slots that assign a meaning to them, e.g., *Phone, model, seller*, or *price*. The definition is simple because the problem is simple to formulate; what makes it a challenging research field is that devising a machine learner that can learn such a mapping as effectively and efficiently as possible is not trivial at all. This has made it quite an active research field; for instance, as of the time of writing this article, our library reports on roughly 4 190 proposals that have been published in the last decade.

The existing proposals can be classified as rule-based, which require a rule set that specifies how to extract the information of interest, or heuristic-based, which have built-in extraction rules that are based on heuristics. Depending on the kind of document on which they work, they can be further classified as free-text or semi-structured. In the literature, there are many proposals to learn rule sets. Some of them are supervised, that is, they require the user to provide an annotated learning set from which rules that map the information of interest onto appropriate user-defined slots are learnt automatically; contrarily, others are unsupervised, which means that they can learn the rule sets from learning sets that are not annotated, but require the user to interpret them and handcraft mappings that assign the information extracted by each rule onto the appropriate user-defined slot. Many proposals are closed, chiefly in the field of semi-structured information extractors, which means that they are intended to work with documents from a given source; a few ones, chiefly in the field of free-text information extractors, are open, which means that they are intended to work with documents on a given topic, independently from the site from which they are downloaded.

Currently, there are many proposals on web information extraction in the literature. Laender et al. [2], Chang et al. [3], Kushmerick and Thomas [4], Turmo et al. [5], Sarawagi [6], Sleiman and Corchuelo [7], and Ferrara et al. [8] have published some comprehensive surveys on this topic. Unfortunately, heuristic-based proposals have not been surveyed so far; the reader might be interested in consulting references [9–16] for further information. Etzioni et al. [17] provided additional details on open information extractors.

* Corresponding author. Tel.: +34 954552770.
E-mail addresses: patriciajimenez@us.es (P. Jiménez), corchu@us.es (R. Corchuelo), hassan.sleiman@cea.fr (H.A. Sleiman).

http://dx.doi.org/10.1016/j.knosys.2015.11.004
0950-7051/

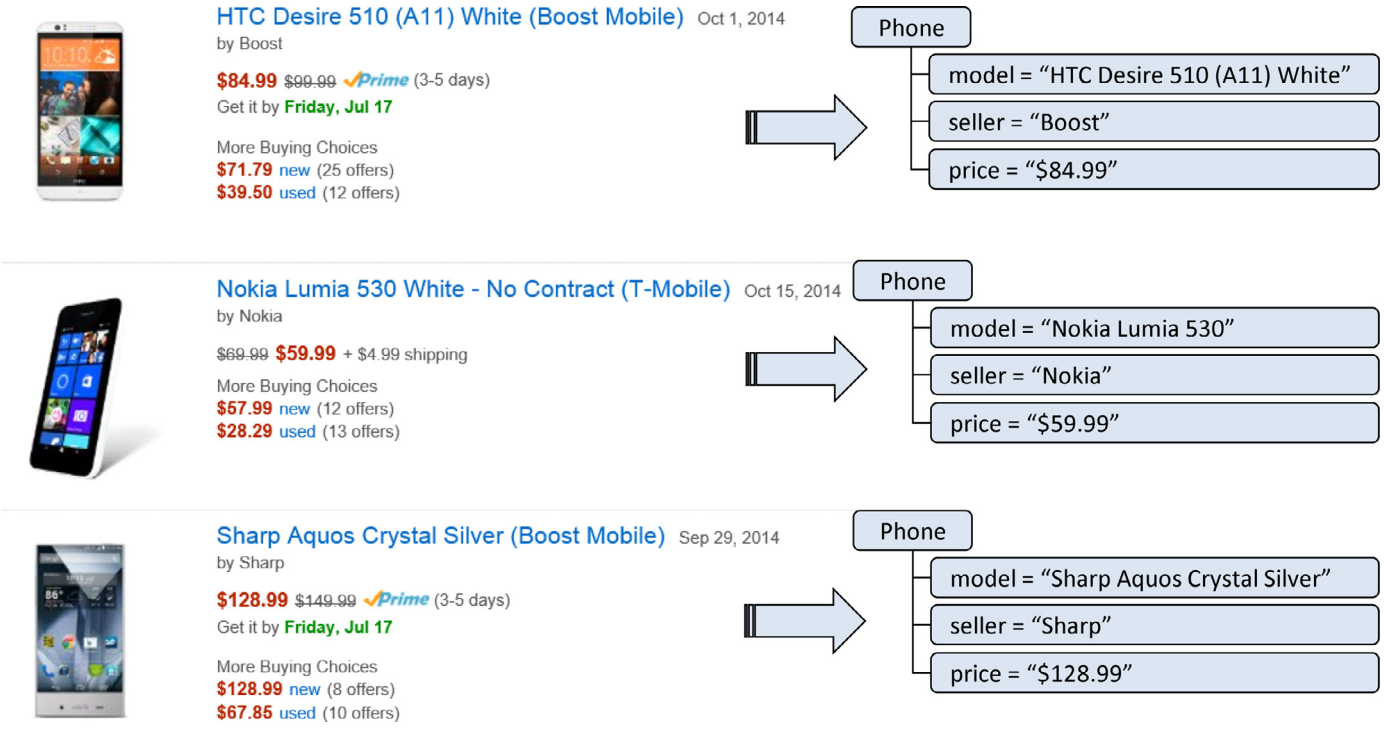

Fig. 1. An illustration of information extraction.

The authors of new proposals must obviously compare them to others so that they can prove that they have introduced conceptual innovations that advance the state of the art. But this is not enough: it is also necessary to rank them regarding their performance; in other words, it is necessary to evaluate them regarding some effectiveness and efficiency measures and then compare the results so as to compute a ranking in which the best-performing proposals are at the top. Practitioners are obviously very interested in such rankings since they lay the foundation to make informed decisions regarding which proposal should be used to solve a given problem.

In our opinion, a good ranking method must have the following key features: it must be automated, so that researchers can bias the conclusions as little as possible, open, so that it can easily accommodate new performance measures, and agnostic, so that it can be applied to as many different kinds of proposals as possible. Furthermore, it must also address the following key questions: how to set up the experimental environment, how to create appropriate evaluation splits, how to compute the experimental data, how to cook them (regarding how to purge them, compute derived measures, or normalise them), how to compute the rankings, and how to report on the results.

We have surveyed many proposals on web information extraction that use an informal method to rank them [2–8]. Unfortunately, our conclusion is that they provide a foundation and some guidelines, but do not have the key features or address the key questions that we have identified above regarding a good ranking method. The informal methods were not intended to be reused, but to help the authors of a proposal support the idea that it outperforms others; as a conclusion, they are not automated, open, or agnostic, but ad-hoc; furthermore, they do not usually disclose many important details regarding the experimental environment; it is not commonly clear how the evaluation splits are created; neither is it clear how the matchings required to compute most effectiveness measures are counted; the experimental data are not cooked; and the resulting rankings are not generally statistically sound. As a conclusion, the stringency level varies from paper to paper, which makes the results available in the literature difficult to reuse when a new proposal needs to be compared to them. Unfortunately, there are very few formal methods in the literature [18–23]. They are generally supported by software tools that aid in computing the experimental data, but they are not actually automated; neither are they open, since they commit to a particular set of performance measures and everything in the method revolves around them; they all originated in a community that was interested in supervised free-text proposals, so they have not paid attention to other kinds of proposals; they report on several alternatives to create evaluation splits, but do not assess the pros and cons or commit to a specific method; they gather experimental data and compute precision- and recall-related measures, but it is not clear how they compute the matches; they do not provide a method to cook the experimental data; and the resulting rankings must be handcrafted, although they pay attention to ensuring that the results are statistically sound.

In this article, we report on ARIEX (Automated Ranking of Information EXtractors), which is a method to evaluate, compare, and then rank web information extraction proposals. It overcomes the problems that we have found in the literature since it reduces the bias that a researcher can introduce in the results because it is automated; it can easily accommodate new performance measures as they are devised and proven to be adequate in our context because it is open; it does not commit to a particular kind of extractor, but has been designed to rank as many proposals as possible because it is agnostic; it provides a clear guideline regarding how the experimental environment must be set up, with a special emphasis on selecting the most appropriate set of performance measures so that the conclusions are global and unbiased; it provides a method to compute as many evaluation splits as possible out of the datasets available; it provides a method to compute the experimental data that takes into account how matchings are computed and does not neglect unsupervised or heuristic-based proposals; it provides a new statistically-sound method to purge the experimental data, it also takes into account derived measures, and provides a normalisation method that makes it open; and it provides a statistically sound method to compute per-measure rankings and then combine them all taking into