A Transducer Model for Web Information Extraction

Hassan A. Sleiman, Inma Hernández, Gretel Fernández, Rafael Corchuelo
University of Seville
Departamento LSI, ETSI Informática
Avd. Reina Mercedes S/N, Sevilla 41012, Spain
Email: {hassansleiman,inmahernandez,gretel,corchu}@us.es

***Abstract*—In recent years, many authors have paid attention to web information extractors. They usually build on an algorithm that interprets extraction rules that are inferred from examples. Several rule learning techniques are based on transducers, but none of them proposed a transducer generic model for web information extraction. In this paper, we propose a new transducer model that is specifically tailored to web information extraction. The model has proven quite flexible since we have adapted three techniques in the literature to infer state transitions, and the results prove that it can achieve high precision and recall rates.**

***Index Terms*—Web Information Extraction, Inferring Transducers.**

I. Introduction

Web information provides added value in many automated business processes. Unfortunately, there are many web sites that do not provide a programmatic interface, which implies that such business processes need to extract and structure the information they require from HTML pages. During the last decades, this has motivated many researchers to work on information extractors [2]. A few rely on heuristics to identify repetitive data [1][11], but most of them are algorithms that a user can configure by means of site-specific extraction rules. The literature provides a variety of techniques to infer these rules, which range from regular expressions to context-free grammars, horn clauses, tree templates, or transducers, to mention a few. Unfortunately, none of them is universally applicable, which makes this quite an active research area [4].

Our focus is on information extractors whose extraction rules are transducers. From a theoretical point of view, a transducer is a finite-state machine with an input tape and an output tape, contrarily to regular finite automata, which have a single input tape. In our context, the input tape consists of an input HTML page, which is represented as a sequence of characters, and the output tape is a sequence of slots that hold part of the information in the input tape in a structured format. Information to be extracted from web pages depends on the context. The benefit of using transducers instead of other types of rules is that they can easily deal with repeating, alternating or optional input patterns.

There are a number of techniques in the literature to infer transducers from sample web pages [5][10][8][3]. They all are supervised techniques, i.e., they require a user to annotate several web pages to create a training dataset. The annotation process, c.f. Figure 1, consists of tagging the relevant information fragments to be extracted, e.g., using special tags, an external offsets file, or an external XPaths file. The goal of the learning process is to learn a transducer that transforms pages, that are similar in structure to the pages in the training dataset, into sequences of structured slots that represent part of the information in the input pages.

Unfortunately, there does not seem to be a common transducer model in the literature of information extraction. The techniques available have been developed in isolation, which has resulted in several transducer models in current use. In this paper, we present a new transducer model that unifies existing models. We have devised a general algorithm that allows to infer the skeleton of our transducers from a training dataset, and we have adapted three techniques in the literature to learn state transitions. Obtained results show that our model can be used by several rule learning techniques and it can achieve high effectiveness in the domain of information extraction.

This paper is organised as follows: First, Section II surveys briefly some proposals on information extraction that used transducers. Section III presents our transducers model for information extraction using formal language. Section IV describes how transducers are created and how different rule learning algorithms can be adapted for this purpose. Experimental results are reported in Section V to validate our model. We conclude our work in Section VI.

II. Related Work

Information extractors are used to extract and structure information from unstructured web pages, such as news and blogs, or from semi-structured 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 proposals for information extraction from semi-structured web pages.

The literature has provided several proposals on information extraction that are based on transducers. Furthermore, another proposals can be adapted easily by reusing their learning algorithms to learn transition conditions of a transducer. We survey briefly these proposals:

- Softmealy [5] constructs a transducer in which each state indicates an attribute to extract and the order in which the annotations appear in the input web pages. Separating

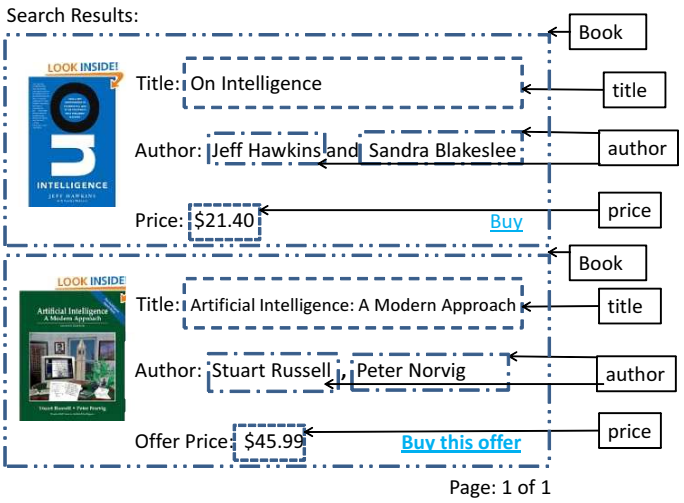

Fig. 1. Annotations performed by user.

text for each transition is obtained from input web pages and then aligned and generalised to learn the transition conditions. It uses a hierarchical tokenisation and an alignment algorithm.

- WIEN [9] identifies six learning classes. These classes work on learning the maximum common prefix and suffix for each type of annotation and then interpret them as regular expressions. Some of these classes extract flat attributes, while others extract nested data. In this work we use LR learning class which detects the largest common prefix and suffix tokens.
- FiVaTech [6] works on learning a template tree that defines the information present on a web page. It uses a clustering technique to label HTML tree nodes and then applies alignment and pattern detection techniques over the sequences of nodes. Then, FiVaTech works on detecting the data scheme used in these pages by detecting tuple types and the order in which information appears.

Another proposal on transducers for information extraction is the one proposed in [3]. This technique described how traditional transducers can be considered as information extractors, but does not propose any rule learning technique. The difference with our model is that ours is tailored specifically to information extraction, and labelling of extracted information, and allows adapting several rule learning techniques for this purpose.

III. Transducer Model

A. Formal Definition

A transducer is a tuple of the form (S, T, i, f, M), where S denotes a finite set of states, T a finite set of transitions, $i \in S$ the initial state, $f \in S$ the final state, and M is a map that associates a user-defined label with every state. Transitions are tuples of the form $(a, L_1 \mid R_1 \rightarrow L_2 \mid R_2, b)$, where $a \in S$ is the source state, $b \in S$ is the target state, and $L_1 \mid R_1 \rightarrow L_2 \mid R_2$ is a condition in which L_1, R_1, L_2, and R_2 are regular expressions. If $a = i$, then $L_1 = \hat{}$, which is a regular expression that denotes the beginning of a string; if $b = f$, then $R_2 = \$$, which is a regular expression that denotes the end of a string.

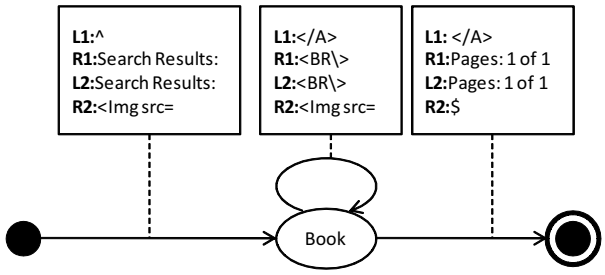

Fig. 2. An example of a transducer to extract Books learnt by a specific rule transition technique from annotations in Figure 1.

Transducers run on configurations of the form (W, R, c, p), where W is the sequence of characters in an input web page, R is a sequence of slots, c denotes the current state, and p denotes the current offset in W. Information extraction takes place at every state, excluding i and f; map M is used to assign a label to every piece of information that is extracted in a state. By slot we mean a piece of text that is tagged with a user-defined label; note that a slot may refer to an attribute, e.g., title, author, or price, or to a record, which is a piece a text to which another transducer must be apply in order to extract its attributes, e.g., Book.

At run time, every transducer starts in a configuration of the form $(W, \langle\rangle, i, 0)$. Given an arbitrary configuration of the form (W, R, c, p) a transition of the form $(c, L_1 \mid R_1 \rightarrow L_2 \mid R_1, c')$ may take place as long as the following conditions hold:

- L_1 matches W at position $pos_{L_1} \geq p$); we denote the length of this match as len_{L_1};
- R_1 matches W at position $pos_{R_1} = pos_{L_1} + len_{L_1}$; we denote the length of this match as len_{R_1};
- L_2 matches W at position $pos_{L_2} \geq pos_{L_1} + len_{L_1}$; we denote the length of this match as len_{L_2}.
- R_2 matches W at position $pos_{R_2} \geq pos_{L_2} + len_{L_2}$; we denote the length of this match as len_{R_2}.

In case the previous transition takes place, the configuration is transformed into (W, R', c', p'), where $R' = R \frown \{M(c) \mapsto W[p \,..\, pos_{L_1} + len_{L_1} - 1]\}$ if $c \neq i$, $R' = R$ if $c = i$, and $p' = pos_{L_2} + len_{L_2} - 1$; $\alpha \frown \beta$ represents the concatenation of sequences α and β, and $\alpha[i \,..\, j]$ the subsequence of α between indices i and j.

B. Disambiguation

Consider the following transducer (S, T, i, f, M) and the configuration (W, R, c, p) where c is the source state for n transitions where $n > 1$, these transitions are as follows:

- $T_1 : (c, L_1 | R_1 \rightarrow L_2 | R_2, c')$
- $T_2 : (c, L'_1 | R'_1 \rightarrow L'_2 | R'_2, c'')$

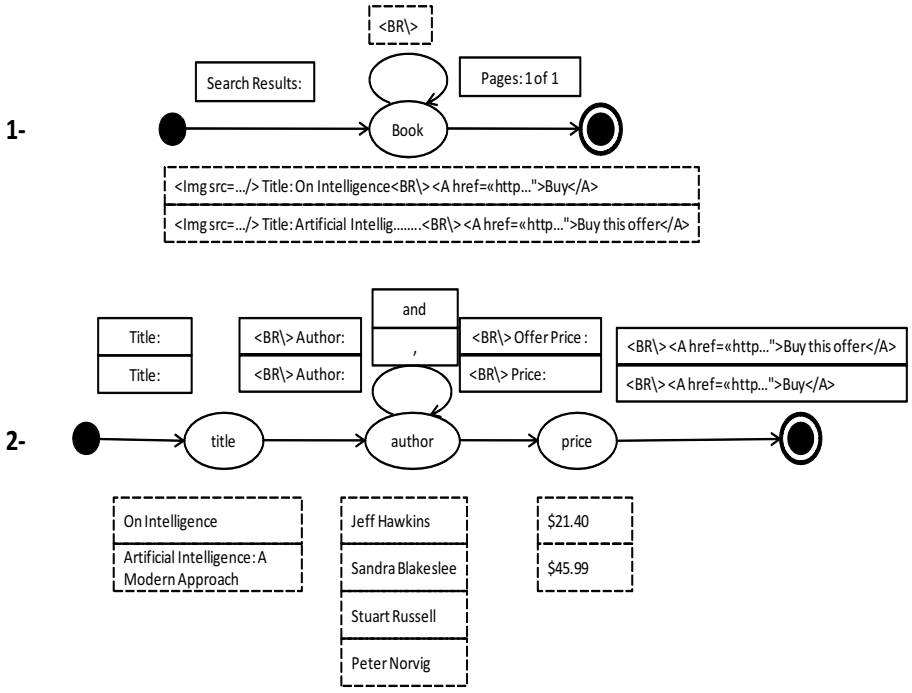

Fig. 3. Skeleton learnt from the annotated examples in Figure 1

.

- $T_n : (c, L_1^n | R_1^n \rightarrow L_2^n | R_2^n, c^n)$

When more than one transition conditions hold, an ambiguity should be resolved in order to select which transition should be performed. A set of heuristics were studied to select the next transition:

- Select the first transition that matches without checking the other ones.
- Assigning to each transition a weight and sorting them in a descendent way. This weight is obtained by calculating the number of cases in the training dataset where this transition took place.
- Sort the transitions according to the matching index of L_1—R_1 condition. If the matching index is equal in all cases, they are sorted by keeping the transitions to the final states at the end of these possible transitions.

While the first heuristic is arbitrary which causes results to vary depending on which transition is chosen at each moment, the second one gives priority to usual transitions while the third one considers the earliest matchings more important. According to our experimental results, the last heuristic performs better than the first two ones.

C. Transducer Example

Figure 2 shows a transducer learnt from the annotated examples in Figure 1. Its transition conditions learnt using a rule learning technique from literature. When the conditions of the transition from the initial state to the state labelled with $Book$ tag hold, then the transducer consumes characters until reaching the first Book and the current state changes to this state. It now tests which of the two transitions, whose source is the current state, hold. In case the conditions of the transition from Book to Book hold, a Book is extracted and the pointer now points at the start of the next book. In case the conditions of the transition whose target is the final state hold, a book is extracted and the transducer consumes until the end of the input text. In case there is an ambiguity and the two transitions whose source is the state "Book", one of the previous heuristics can be applied to select and perform a transition.

IV. Learning Transducers

A. Learning a Skeleton

A skeleton contains a transducer that does not have transition conditions and all the information from the training dataset that originated the states and transitions in the transducer. It is the learning technique responsibility to learn these conditions. The skeleton is learnt starting from a learning dataset and it describes the structure, type and order of relevant information on the web pages from the learning dataset.

A skeleton is a tuple of the form $\delta(T, \mu, \iota)$, where T is a transducer, μ is map in which for each transition t in the the transducer T, it saves a list of all the separating texts that originated this transition. ι is a map that saves for each state s in the transducer T, a list of all the annotations that originated this state.

Figure 3 shows two skeletons learnt starting from the annotated examples in Figure 1. The first contains an uncomplete transducer created to extract book records, but it needs a rule learning algorithm to learn its transition condition. ι in the

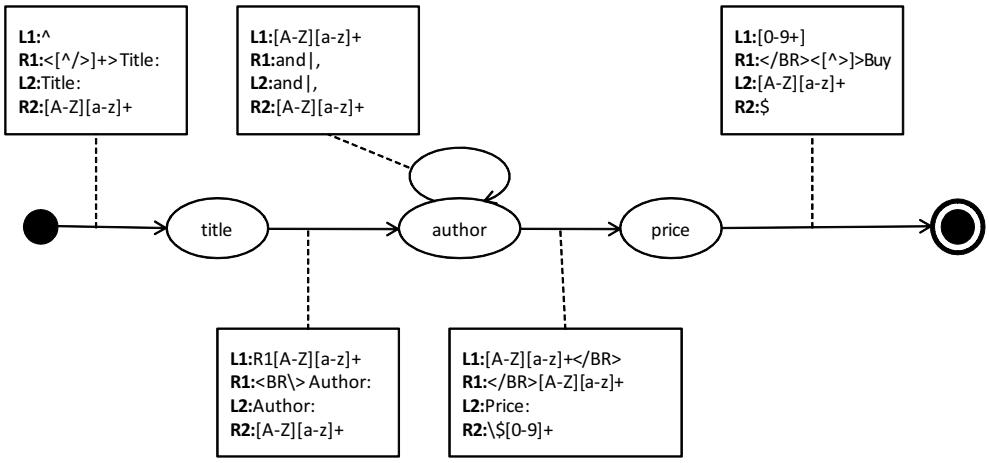

Fig. 4. Transducer learnt using SM learning technique.

first skeleton contains a map with a unique entry for the state with the label "Book" that contains a list of two elements which are the text fragments of the two annotated books. μ has 3 entries, one for each transition. Each entry in μ contains one text fragment since this transition appeared once in the annotated examples. The second skeleton in Figure 3 is learnt to extract Book attributes from text fragments extracted by the "Book" state in the first skeleton.

B. Learning Transition Conditions

Once a skeleton is learnt from a learning dataset, a learning algorithm is now necessary to learn transition conditions. Different learning techniques can be adapted for this purpose counting on the separators and annotations saved in a learnt skeleton. Consider the skeleton learnt in Figure 3, it is possible to learn transition conditions using techniques to infer regular expressions starting from a collection of strings. We adapted three techniques in the literature to prove the flexibility of our transducer model. These techniques are:

- NLR: It was inspired by WIEN [9] proposal, and concretely, from the LR class inside this WIEN proposal. The LR class searches for the longest common prefix and suffix for each annotation and converts it to a regular expression. In our case, L_1 condition was learnt by searching for the common tokens at the end of the individuals in the origin state. R_1 condition was learnt by searching for the common tokens at the beginning of all the separating texts for the transition. Common tokens at the end of these separators are used to learn L_2 and the common tokens at the beginning of the individual texts in the target state are used to learn R_2. Figure 2 shows a transducer to extract books learnt applying NLR technique.
- SM: This technique is inspired by SoftMealy [5]. It uses an alignment and generalising algorithms. It only considers the tokens at the beginning and at the end of the separators and of the individuals in each state that are limited by tokens of type WORD. Tokens of type WORD are tokens that are digits and characters, excluding spaces, punctuation and HTML tags. A multiple string alignment and a generalisation algorithms are applied on each set of the considered tokens to learn transition conditions.Figure 4 shows a transducer to extract attributes inside Books learnt applying our SM technique.
- FT: It is inspired by a multiple string alignment technique used in FiVaTech [6]. Separators in each transition are aligned using the multiple sequence alignment algorithm used in FiVaTech to create regular expressions and learn transition conditions.

V. Experimental Results

To perform our experiments, we first created a collection of datasets from current web sites and followed the guidelines from [7] to perform our empirical comparison.

Table I reports on the results of applying these techniques in practice on several datasets compared side by side. Each dataset contains 30 annotated web pages and a 10-folding cross validation was performed to calculate precision (P) and recall (R). The time column reports the time necessary to learn a transducer for each dataset. The experiments were run on a Windows Server 2008 (64-bits) machine that was equipped with a four core Intel Xeon 3.00 GHz CPU, 16 GB RAM, and JRE 1.6.0.

Sites to create the datasets were chosen arbitrarily and the web pages inside each dataset were chosen to cover all possible data variability, such as Books with one or more authors and Books with and without price. Note that these techniques can obtain better precision and recall by adding more web pages to these learning datasets, but this is not our study case since we are just checking if implemented techniques can achieve good results and to validate our transducers model. Better results may be obtained if each one of the used web sites are studied in-depth to choose the web pages to include in the web site's dataset.

TABLE I
COMPARING PRECISION AND RECALL OF NLR, SM AND FT TECHNIQUES USING TRANSDUCERS.

Dataset	NLR			SM			FT		
	P	R	Time	P	R	Time	P	R	Time
netlib	0.433	0.393	0.312m	0.853	0.520	0.544m	0.862	0.176	1.210m
albanianfilmdatabase.com	0.874	0.245	0.700m	0.874	0.304	0.120m	0.962	0.371	0.610m
disneymovieslist.com	0.731	0.731	0.900m	0.989	0.460	0.150m	1.000	0.000	0.930m
imdb.org	0.753	0.855	2.300m	0.985	0.845	0.630m	1.000	0.124	2.310m
citwf.com	0.915	0.915	4.800m	0.981	0.878	0.130m	0.992	0.892	4.810m
betterworldbooks.com	0.993	0.915	0.530m	0.877	0.844	0.411m	0.920	0.514	0.747m
manybooks.net	0.974	0.824	0.480m	0.746	0.067	0.550m	0.770	0.536	0.311m

VI. CONCLUSIONS

We have proposed a new powerful model of transducers for web information extraction. To validate our model, we have adapted several rule learning algorithms which shows the feasibility and flexibility of our model. The model has high expressiveness since it supports disjunction, multi-order attributes, optional and multi-valued attributes. The experimental results demonstrate the high effectiveness that can be achieved on current web sites using our model which is almost 100% in some cases.

ACKNOWLEDGMENT

This paper was supported by the European Commission (FEDER), the Spanish and the Andalusian R&D&I programmes (grants TIN2007-64119, P07-TIC-2602, P08-TIC-4100, TIN2008-04718-E, TIN2010-21744, TIN2010-09809-E, TIN2010-10811-E, and TIN2010-09988-E).

REFERENCES

[1] Manuel Álvarez, Alberto Pan, Juan Raposo, Fernando Bellas, and Fidel Cacheda. Extracting lists of data records from semi-structured web pages. *Data Knowl. Eng.*, 64(2):491–509, 2008.

[2] Chia-Hui Chang, Mohammed Kayed, Moheb R. Girgis, and Khaled F. Shaalan. A survey of web information extraction systems. *IEEE Trans. Knowl. Data Eng.*, 18(10):1411–1428, 2006.

[3] Boris Chidlovskii. Wrapping web information providers by transducer induction. In *European Conference on Machine Learning*, pages 61–72, 2001.

[4] AnHai Doan, Jeffrey F. Naughton, Raghu Ramakrishnan, Akanksha Baid, Xiaoyong Chai, Fei Chen 0002, Pedro DeRose, Byron J. Gao, Warren Shen, Ting Chen, Ba-Quy Vuong, Eric Chu, Chaitanya Gokhale, and Jiansheng Huang. Information extraction challenges in managing unstructured data. *SIGMOD Record*, 37(4):14–20, 2008.

[5] Chun-Nan Hsu and Ming-Tzung Dung. Generating finite-state transducers for semi-structured data extraction from the web. *Inf. Syst.*, 23(8):521–538, 1998.

[6] Mohammed Kayed and Chia-Hui Chang. FiVaTech: Page-level web data extraction from template pages. *IEEE Trans. Knowl. Data Eng.*, 2010.

[7] Barbara Kitchenham, Shari Lawrence Pfleeger, Lesley Pickard, Peter Jones, David C. Hoaglin, Jarrett Rosenberg, and Khaled El Emam. Preliminary guidelines for empirical research in software engineering. *IEEE Trans. Software Eng.*, 28(8):721–734, 2002.

[8] Raymond Kosala, Hendrik Blockeel, Maurice Bruynooghe, and Jan Van den Bussche. Information extraction from structured documents using k-testable tree automaton inference. *Data Knowl. Eng.*, 58(2):129–158, 2006.

[9] Nicholas Kushmerick. Wrapper induction: Efficiency and expressiveness. *Artif. Intell.*, 118(1-2):15–68, 2000.

[10] Ion Muslea, Steven Minton, and Craig A. Knoblock. Hierarchical wrapper induction for semistructured information sources. *Autonomous Agents and Multi-Agent Systems*, 4(1/2):93–114, 2001.

[11] Nikolaos Papadakis, Dimitrios Skoutas, Konstantinos Raftopoulos, and Theodora A. Varvarigou. 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):1638–1652, 2005.