Towards Domain-Independent Information Extraction from Web Tables*

Wolfgang Gatterbauer, Paul Bohunsky, Marcus Herzog,
Bernhard Krüpl, and Bernhard Pollak
Database and Artificial Intelligence Group
Vienna University of Technology, Austria
{gatter,bohunsky,herzog,kruepl,pollak}@dbai.tuwien.ac.at

ABSTRACT

Traditionally, information extraction from web tables has focused on small, more or less homogeneous corpora, often based on assumptions about the use of `<table>` tags. A multitude of different HTML implementations of web tables make these approaches difficult to scale. In this paper, we approach the problem of domain-independent information extraction from web tables by shifting our attention from the tree-based representation of web pages to a variation of the two-dimensional visual box model used by web browsers to display the information on the screen. The thereby obtained topological and style information allows us to fill the gap created by missing domain-specific knowledge about content and table templates. We believe that, in a future step, this approach can become the basis for a new way of large-scale knowledge acquisition from the current "Visual Web."

Categories and Subject Descriptors: H.2.8 [Database Management]: Database Applications – *Data Mining*; H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval

General Terms: Algorithms, Experimentation, Theory

Keywords: Information extraction, Web mining, Web tables, Web page representation, Visual analysis

1. INTRODUCTION

The Web is an enormous source of information contained in billions of individual pages. Information extraction (IE) tries to process this information and make it available to structured queries. Most often, information extraction systems are targeted towards specific domains of interest and involve either manual or semi-automatic learning of the target examples involved. In contrast, the goal of automatic information extraction is to discover relations between data items of interest and similar data items on a large scale and independently of their domain without any training [2, 14]. One principal idea is that an extraction system makes a single data-driven pass over its entire corpus and extracts a large set of relational tuples without requiring any human input [4]. Traditionally, such domain-independent information extraction systems aim to find relations in unstructured plain text by applying natural language processing techniques [10]. In contrast, we approach the problem of large-scale and domain-independent information extraction from web tables. Tables are interesting because they present information in a condensed, rather simple, and well structured way. At the same time, domain-independent large-scale attempts at information extraction have been difficult partly because the prevailing approach of the IE research community has been to analyze either the plain text content or the tree structure of web pages for relevant tabular extraction patterns. Tables, however, only explicitly reveal the true nature of their data structure in a two-dimensional (2-D) context, whereas the source code only "encodes" the visual information in an implicit and rather difficult to analyze format (Fig. 1).

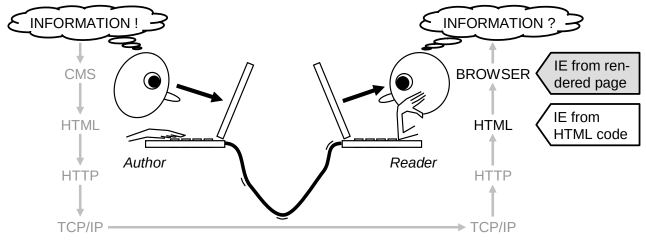

Figure 1: Information extraction (IE) from rendered web pages emulates the process by which internet users encode and decode information in the current "Visual Web."

We therefore deviate quite drastically from the prevailing approach of web information extraction and cast the problem into a formalism that moves the focus from a tree-structured to a 2-D pattern recognition problem using a variation of the CSS2 visual box model. Following this approach, we show that extracting information from web tables is possible without reliance on "heavy" linguistic techniques tuned to the domain of interest and, thereby, differ from previous table extraction systems, which were dominantly domain-specific (e.g. extraction of catalogues with product information).

*This research was supported in part by a DOC scholarship from the Austrian Academy of Sciences and by the FIT-IT programme of the Austrian Federal Ministry for Transport, Innovation and Technology.

Overall, the main contributions of this paper can be summarized as follows: (1) We generally classify visually structured data on the Web. (2) We cast the problem of information extraction from the Web – in particular from web tables – into a formalism that moves away from the commonly favored tree-based representation of web pages. (3) We motivate an output format of information extraction from tables that allows for multiple interpretations of the data extracted in a later probabilistic information integration step. (4) We outline a universal ground truthing methodology that allows to ground truth web tables independent of their implementation. (5) We introduce a challenging test set of web tables that was compiled by 63 students and ground truthed by us using our methodology. (6) We present first experimental results which indicate that domain-independent information extraction from tables can be performed without the need of heavy linguistic methods, relying only on visual characteristics of the tables.

After a brief review of related work (Section 2), we characterize the phenomenon "web table" and related visually structured data structures (Section 3). We then lay out the formal setup of our method (Section 4), before we describe our chosen approach in more detail (Section 5). In Section 6 we outline our method to ground truth web tables and apply it to an objectively chosen test collection of web tables. We use this test set to evaluate our system (Section 7) and conclude after a brief discussion of next steps (Section 8).

2. RELATED WORK

Web Table Analysis. Web table analysis has become a widely researched topic on its own over the years. Penn et al. [25] define genuine uses of HTML tables as document entities where the 2-D grid is semantically significant and describe a couple of heuristics to distinguish genuine from non-genuine leaf `<table>` tables on web pages. Yalin Wang and Hu [33] train a classifier on content features of individual cells and non-text layout features from the HTML source to perform the same task of table location. Chen et al. [7] employ heuristic rules to filter out non-genuine tables from their test set and make assumptions about cell content similarity for table recognition and interpretation. Like most of the approaches we are aware of, their method relies on the hierarchical HTML tag structure of the documents, most notably that of `<table>` tags. Yang and Luk [36] describe how they extract attribute-value pairs from 1-D or 2-D tables, a notion similar to our view of 1-D and 2-D lists and tables. But for them, a 1-D table contains what they call "mixed-cell" content, which means that the attribute name (label) and value are in the same cell. We regard this data structure as nested, substructured list. Yoshida et al. [38] base their work on a general knowledge ontology and employ an expectation maximization algorithm to distinguish between attribute and value cells. They assume that tables do no contain any spanned cells. Tengli et al. [29] present an algorithm that extracts tables and differentiates between label and data cells.

All these approaches have in common that they assume that relevant tables only appear inside leaf tables, which are such `<table>` tags that do not contain other nested `<table>` tags. In contrast, Lerman et al. [22] mention that just a fraction of tables are actually created with `<table>` tags. In their algorithm, they leverage the list page-detail page structure present in some websites to find boundaries between records in what we would classify as a substructured 1-D list. They also mention that layout is important for table extraction, but go on to say that this means that records are separated by HTML tags. In contrast, we base our table extraction on positional information that is independent of the HTML tag structure and do not rely on particular HTML structures being present. Tijerino et al. [30] describe the automatic generation of ontologies from normalized tables, which is a structure they get after normalizing table-equivalent data. Pivk et al. [26] focus on understanding table-like structures only due to their structural dimension and transforming the most relevant table types into F-logic frames. Neither paper makes it clear to us how the described approaches locate tables on web pages in the first place.

We refer to two recent surveys on table analysis by Embley et al. [12] and by Zanibbi et al. [39], which cover a broad range of table processing literature and illustrate the general challenges of information extraction from tables despite their primary focus on non-HTML documents.

Visual Web Page Analysis. As far as we know, the idea of analyzing the visual representation of a web page for content analysis originated in the area of web page segmentation. Yang and Zhang [37] describe an approach which derives features directly from the layout of web pages. By using a "pseudo rendering process" they try to detect "visual similarities" of HTML content objects. We fully believe in their remark that HTML tags are not stable features for analyzing structures of HTML documents. Gu et al. [16] describe a top-down approach to segment a web page and detect its content structure by dividing and merging blocks. Kovacevic et al. [19] use visual information to build up a "M-tree", a concept similar to the DOM tree enhanced with screen coordinates. They then use further defined heuristics to recognize common page areas such as header, left and right menu, footer and center of a page. Cai et al. [6] describe a web page segmentation process that uses visual information from Internet Explorer. Their VIPS algorithm segments a DOM tree based on visual cues retrieved from the browser's rendition. Cosulschi et al. [9] describe an approach that uses positional information of DOM tree elements to calculate block correspondence between web pages.

Visual web page analysis can also be increasingly found in information extraction literature, specifically on record boundary detection. Zhao et al. [41], Zhai and Liu [40] and Simon and Lausen [28] describe different approaches for detecting repetitive patterns on web pages, which are predominantly source-code based and enhanced with visual cues. In contrast, Aumann et al. [3] describe a system that works only on a hierarchical structure of the visual representation (experiments are performed with PDF documents) and learns to recognize text fields such as author or title from manually tagged training sets of documents. Conversely, our approach does not attempt to find individual text fields, but rather, larger structures, does not require training sets and neither imposes a tree structure on web pages.

Visual Web Table Extraction. To our best knowledge, the idea of actually rendering or "executing" HTML code and using the results for detecting relational information in web tables was first mentioned by Cohen et al. [8]. The described approach, however, does not actually render web pages, but rather infers relative positional information of table nodes in an abstract table model with relative positional information deduced from the source code. In contrast, in our previous work [21] we describe a top-down web table location mechanism working exclusively on visual information obtained from the Mozilla web browser. The approach works on word bounding boxes after manipulation of the DOM tree and detects tables with the help of space density graphs and recursive application of the X-Y cut algorithm. This approach is later adapted in [20] to a bottom-up clustering algorithm starting with word bounding boxes as well. In [15], we describe a method that works on both word and element node bounding boxes, and that is able to locate concepts of a predefined seed knowledge in web tables.

The system of this paper builds upon this last method and extends it in the following ways: (1) the approach does not need pre-defined seed knowledge and is as such completely domain-independent; (2) the approach can recognize and transform the tabular information into a format, suitable for a subsequent information integration step.

3. VISUALLY STRUCTURED DATA ON THE WEB

Web information extraction, building upon structural features of the data on web pages, is probably the most intense studied research topic of web content mining [23]. Structured data formats are often used to represent information in a concise and unambiguous form, e.g. lists of products and services. Extracting such data in an automated fashion allows to provide value added services, e.g., comparative shopping [5], and meta-search [41]. As a result, a large amount of literature has been published on different approaches to structured data extraction, which is also known as wrapper generation. Such wrappers commonly try to learn certain regularities from example source code and then to find other instances of such patterns.

Another approach to structured information extraction, which we suggest in this paper, is to focus on the 2-D visual representation of web pages as intended by authors for readers in the current *Visual Web* [24], instead of the tree-based representation used to encode such information (Fig. 1). Explicit *Semantic Web* annotations are still rare and dynamic web technologies around Web 2.0 translate into an increasingly more complicated *code syntax*, but with more or less the same *visual syntax* used to express similar kind of human-understandable semantic relations. The source code carries the same amount of information, but in an implicit and difficult to analyze format. "Implicit" means that relations between individual items is not available without first fully analyzing and "executing" or rendering the code. And information actually is the result of combining individual data items together with relations between these data items (Fig. 2). Using and decoding the available visual information after rendering a web page allows us to draw additional conclusions, thus filling the gap between available data and domain-dependent semantic relations.

Such visual semantic relations can be expressed in two principal syntactic ways: (1) By *topology*, which concerns the spatial arrangement of the composing units of information. (2) By *typography* or *style*, which concerns metadata information such as font size, font weight or background color. Here, we provide a broad characterization of the first

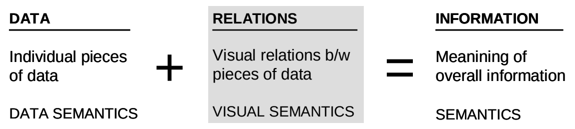
DATA
Individual pieces of data
DATA SEMANTICS
+
RELATIONS
Visual relations b/w pieces of data
VISUAL SEMANTICS
=
INFORMATION
Meaning of overall information
SEMANTICS

Figure 2: In *visually structured data*, meaning of information (semantics) results from meaning of the individual pieces of data (data semantics) and the visual relations between them (visual semantics).

subcategory: *visually structured information that predominantly derives its meaning from the spatial arrangement of its constituent data items.* In other words, the spatial relations between individual elements add some important meta information to the meaning of each data block, without which the information could not be understood to its full extent. Broadly, we find that web tables – the focus of our research – are, together with lists and some domain-specific aligned graphics, one of the three dominant topological data structures found on web pages. Our focus has been to classify the different phenomenons according to their intended purpose as visible to the human observer, not by their implementation, which vary for example for tables from `<table>` over `<div>` and `<li>` tables to tables in non-HTML format. All three can be found either as one dominant structure with atomic data content or with nested substructures as classified in Fig. 3. Below we give a set of definitions for these structures which helped us to develop human-like heuristics for our table extraction step, and which enable us to distinguish tables from similar visual structures as described in section 5.1. These definitions are best understood in connection with Fig. 3.

- **Tables.** We consider the description of tables as given by Vanoirbeek [31] a very useful definition of tables on the Web despite its original focus on printed documents: "In a global way, a table may be defined as a two-dimensional presentation of logical relationships between groups of data. Those connections are reflected by horizontal and vertical alignment of data in a grid." It does not include some unique properties of web pages, which we add for the purpose of defining web tables: *A web table is a two-dimensional presentation of logical relations between groups of data items. Those relations are reflected by different visual properties and by horizontal and vertical alignment of the data items in a visible or implied grid structure, which become observable after a web page is rendered.*

- **Lists.** Trying to find an appropriate definition for lists, we consulted a number of standard references like Encylopaedia Britannica and Merriam-Webster. However, we could not find one definition that describes the meaning we try to convey. As such, the following definition is the result of consulting a number of references and significant discussion in our group: *A list is a series of similar data items or data records. A list can be either one-dimensional or two-dimensional; in both variants, no hierarchical or other semantic relationships in between individual list items is implied except for a possible ordering of the items.*

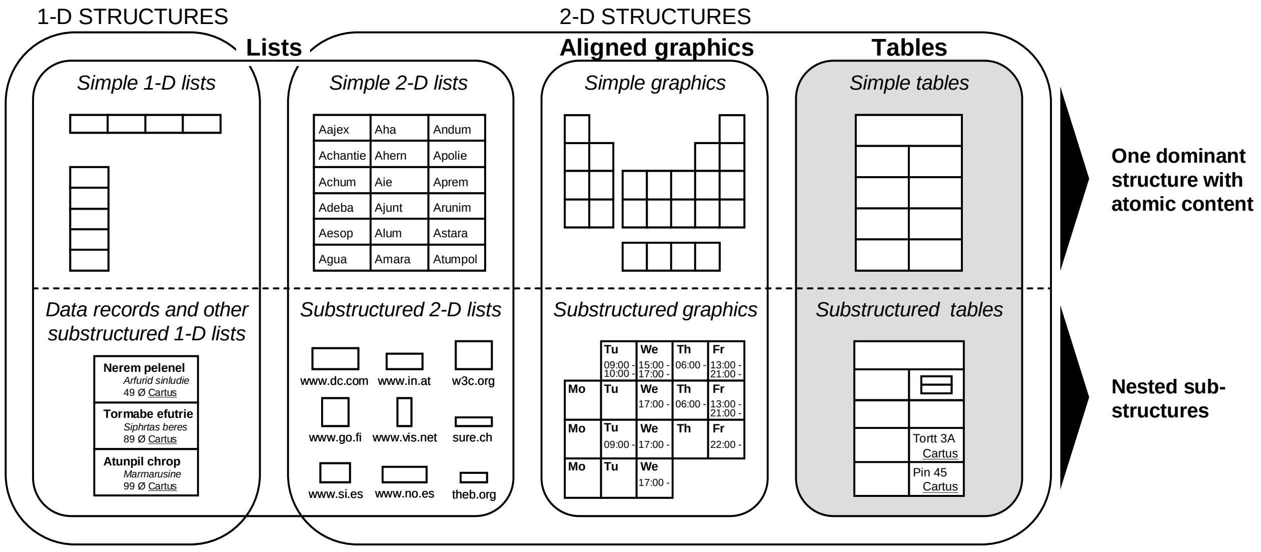

Figure 3: Tables are together with lists and some domain-specific aligned graphics one of the three dominant *spatially structured data* formats found on web pages. The constituent logical units of the dominant structure either have atomic content, or nested substructures such as lists of data records.

- **Aligned graphics.** Aligned graphics are such graphical depictions of relations between entities of a certain domain, which do not fall in either of the two other categories. They are generally domain-specific (like the period system for chemical elements), not necessarily bounded by a rectangle, and relative spatial positions do imply relationships between data items.

The structures described above often appear in nested forms. The *data records* of search engine results are an example of a list where the individual list items consist of repetitive substructures. The characterization looks at the "dominant" structure, which is the principal structure that can not be seen as a substructure of another table, list or other repetitive pattern. It is important to add that this "nestedness" does not necessarily imply nestedness in the source code. We only focus on the visual appearance of structures on rendered web pages, independent of their coding. Reality tends to be more complex than a model of reality, and some "real-world" structures cannot be unambiguously classified in the above scheme. However, we find the level of detail to be a good balance between classifying all available structures and applicability of the model. It serves well the task of developing heuristics to discern tables from other structures. But before, we will formalize our approach of IE from a visual representation of web pages.

4. FORMAL SETUP

In this section, we describe a topological representation of web pages and compare it to the commonly used tree representation. The choice of an appropriate web page description is an important decision as it determines the set of features available to discriminate between relevant and irrelevant information in the extraction process. We will use this web page representation and corresponding formalism in the next chapter to describe our method of extracting information from web tables.

DOM tree representation. Web information extraction and wrapper generation approaches commonly aim at detecting certain information patterns in the tag tree or the *DOM tree* of web pages. The DOM tree as an *attributed ordered rooted tree* is a special case of a directed acyclic graph $T = \langle \mathcal{N}, \mathcal{E}, r \rangle$ where $\mathcal{N} = \mathcal{N}_t \cup \mathcal{N}_e$ with $\mathcal{N}_t$ representing the set of *text nodes*, $\mathcal{N}_e$ the set of *element nodes*, $\mathcal{E} \subset \mathcal{N}_e \times \mathcal{N}$ the directed edges between the nodes, and $r \in \mathcal{N}_e$ the designated root node. Each text node $n_t \in \mathcal{N}_t$ has one non-empty string attribute s and each element node $n_e \in \mathcal{N}_e$ a set of attributes $\mathcal{A}$ with obligatory node name and optional further attribute name-value pairs, e.g. $n_t = \langle s \rangle = \langle$"Banff, Canada"$\rangle$ and $n_e = \langle \mathcal{A} \rangle = \langle \{ name: \text{"TD"}, class: \text{"left"}, bgcolor: \text{"green"} \} \rangle$. All nodes except the root node $n \in \mathcal{N} \backslash \{r\}$ have exactly one parent node $\texttt{parent}(n)$ with $\texttt{parent}(n) \leftarrow p \,|\, (p, n) \in \mathcal{E}$, which determines the hierarchy of the data structure. On a conceptual level, DOM tree-based IE works on three basic elements: (1) the text nodes $\mathcal{N}_t$ as *primitive content elements* containing the actual textual information of interest; (2) the element nodes $\mathcal{N}_e$ as *primitive structural elements* adding metadata to the content; and (3) the edges $\mathcal{E}$ between the nodes representing the actual *hierarchical structure* of the DOM tree. We work with analogous concepts in a different web page model (Fig. 4).

Visual box representation. When HTML documents are rendered by a browser, CSS (Cascading Style Sheets) represent the element nodes of the document by rectangular boxes and govern their layout according to the CSS2 box model and the CSS2 visual formatting model [34]. We refer to such rendered rectangles corresponding to element nodes in the DOM model of a web page as *visualized element nodes* or *VENs* and characterize them with the four coordinates of their position on the screen and a vector of attributes. We add the idea of treating individual words as rendered on the screen as separate entities [21] and call them *visualized words*. We can, therefore, represent a web page by a topological and typographical 2-D visual box model $V_{\mathbf{r}} = \langle \mathcal{V}, \mathcal{X}_e \rangle$

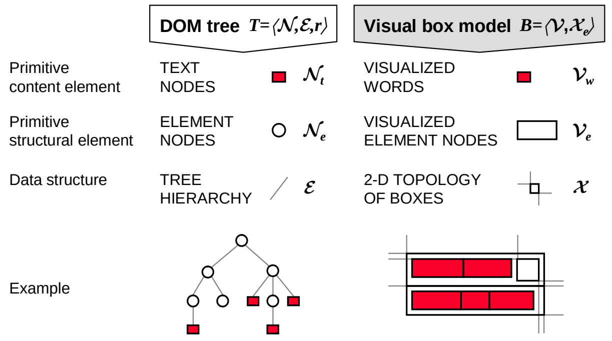

Figure 4: IE from the visual box model of web pages works on three different conceptual elements than IE from the DOM tree.

where $\mathcal{V} = \mathcal{V}_w \cup \mathcal{V}_e$ is the set of primitive data elements with $\mathcal{V}_w$ representing the set of visualized words and $\mathcal{V}_e$ the set of VENs. The parameter $\mathbf{r}$ describes a list of rendering conditions like a chosen rendering algorithm and screen width. Each VEN $e \in \mathcal{V}_e$ consists of the screen coordinates vector of its four borders $\mathbf{x} = \langle x_1, y_1, x_2, y_2 \rangle$ and a property-value attribute vector $\mathbf{a} = \langle a_1, ..., a_{|\mathbf{a}|} \rangle$ including its node name and its computed style attributes: $e = \langle \mathbf{x}^e, \mathbf{a}^e \rangle$. Each visualized word $w \in \mathcal{V}_w$ has additionally one non-empty string s associated that contains the textual content: $w = \langle \mathbf{x}^w, \mathbf{a}^w, s \rangle$. The data structure $\mathcal{X}_e$ is a minimum *double topological grid* [15] superimposed on all VENs and, as such, defined by $\mathcal{X}_e = \{\langle x_1, y_1, x_2, y_2 \rangle \mid x_1 \in \mathcal{X}_1, y_1 \in \mathcal{Y}_1, x_2 \in \mathcal{X}_2, y_2 \in \mathcal{Y}_2, x_1 < x_2, y_1 < y_2\}$ with $\mathcal{X}_1 = \{x \mid (\exists e \in \mathcal{V}_e \mid x_1^e = x)\}$ and analogous $\mathcal{Y}_1$, $\mathcal{X}_2$ and $\mathcal{Y}_2$. Each point $\mathbf{x} \in \mathcal{X}_e$ on this topological grid defines a *hyperbox* which is a rectangular area on the screen where each border is aligned with a corresponding border of at least one VEN. In an analogy to tree-based IE (Fig. 4), the visualized words form the *primitive content elements*, VENs the *primitive structural elements*, and the 2-D spatial topology the *data structure* in our web page representation.

The notion that a visualized element node e *contains* a visualized word w implies that the bounding box of e completely contains the bounding box of w: $(e\ \texttt{contains}\ w) \Leftrightarrow (x_1^e \le x_1^w) \wedge (y_1^e \le y_1^w) \wedge (x_2^e \ge x_2^w) \wedge (y_2^e \ge y_2^w)$. An ordered list of n visualized words $\mathbf{t} = \langle w_1, w_2, \ldots, w_n \rangle$ is said to be in *western reading order*, if its sequence as arranged on a screen is from left to right and between rows top down: $\forall w_i, w_j \in \mathbf{t} : i < j \Leftrightarrow (y_1^i < y_1^j) \vee \big((y_1^i = y_1^j) \wedge (x_1^i < x_1^j)\big)$. We define $\widehat{\mathcal{V}}_e$ as the set of *extended* visualized element nodes with each $\widehat{e} = \langle \mathbf{x}, \widehat{\mathbf{a}}, t \rangle \in \widehat{\mathcal{V}}_e$ corresponding to a VEN $e \in \mathcal{V}_e$ together with a string t and an attribute list $\widehat{\mathbf{a}}$, where t is the concatenation of the set of strings $\{s_i\}$ of all n words $\{w_i\}$ contained in e, in reading order and separated by space characters, and $\widehat{\mathbf{a}}$ is a merge function of the attribute list of e and all contained words w_i, $\widehat{\mathbf{a}} = m(\mathbf{a}^e, \mathbf{a}^{w_1}, ..., \mathbf{a}^{w_n})$. By a *frame* f, we mean a special hyperbox that can be completely tiled with a set $\widehat{\mathcal{V}}_e^f$ of extended VENs. "Completely tiled" refers to a situation where a hyperbox can be covered with VENs on the screen in such a way that the whole area is covered and no VEN overlaps another one (MECE = Mutually Exclusive, Collectively Exhaustive) except for the adjacency condition. We define a frame in such a way that it already contains the set of extended VENs: $f = \langle \mathbf{x}^f, \widehat{\mathcal{V}}_e^f \rangle$.

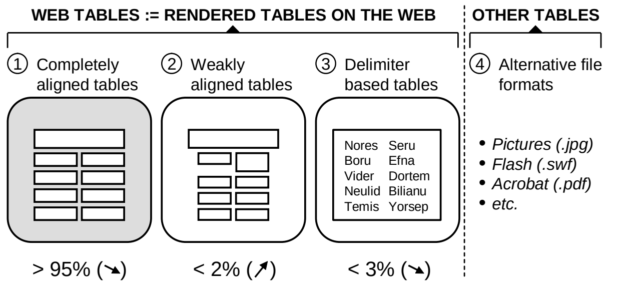

Figure 5: The table location step of VENTex focuses on the most dominant type of tables on the Web: web tables, whose logical cells together render as a completely tiled hyperbox or frame.

5. INFORMATION EXTRACTION FROM WEB TABLES

Our observation is that the majority of web tables topologically form a frame in the visual box model. Figure 5 compares the topology of this kind of completely tiled and "completely aligned tables" (1) with the other three types of tables on the Web: web tables, whose logical cells are formed by visualized element nodes but which are not completely aligned (2: "weakly aligned tables"); web tables whose logical cells are not contained in different visualized element nodes but rather as delimited words inside the same visualized element node or (3: "delimiter based tables"); and tables which have no equivalent HTML code and which, as such, are not "constructed" by rendering in a web browser, e.g. flash and pdf tables or pictures of tables (4). Relative occurrences and future changes are an educated and cautious estimate based on a number of table sets we have seen in the course of our studies. Depending on what kind of particular subset of web tables one focuses on (e.g. tables highly listed in search engines, product comparison pages, handpicked "mean" tables, older web pages, template-driven web pages from large websites, etc.), these numbers will slightly vary. However, the dominance of web tables formed as frames is universally prevailing, which is why we focus on this kind of tables. As the logical cells of tables we search for coincide with VENs on the screen, we refer to our approach as **VENTex** for **V**isualized **E**lement **N**odes **T**able **ex**traction.

The individual steps of table understanding have been defined and named in the table literature in a number of ways. We basically follow the naming and process description given by Hurst [18]. Hurst breaks down the task of table understanding into the following subtasks: (1) table location, the process of spotting tables in documents; (2) table recognition, the task of segmenting the original description of the table into a relative spatial description; (3) functional and (4) structural analysis; and finally (5) table interpretation, the extraction of meaningful and unambiguously structured information. However, we have to slightly accommodate the descriptions above to fit our specific approach. Also, we find a pragmatic division into three consecutive steps helpful:

- **Table location:** the task of identifying tables and their constituent logical cells on web pages. Our approach focuses on tables whose logical cells together form a frame (Fig. 5).

- **Table recognition:** the task of identifying the relative spatial relationships between individual logical cells of a table, which we call the topological structure of a table.
- **Table interpretation:** the task of extracting and saving information from tables in a structured format that preserves the meta information contained in all available visual relationships between the individual categories for a subsequent successful probabilistic semantic information integration step. We will argue that domain-independent table interpretation cannot result in unambiguously structured information because of existing inherent domain-specific ambiguities that can sometimes not even be resolved by humans.

Next, we describe our current implementation and its time complexity and speed in more detail. Our system analyzes any given web page for the existence of tabular data, recognizes relations as implied by their spatial arrangement, extracts a number of n-tuples together with hierarchical information about relations between their entries and saves them in an XML data format which is heavily influenced by the work of Wohlberg [35]. Because the first two steps of locating and recognizing a table are interdependent in our approach (a table frame cannot be reliably discriminated from non-table frames without first knowing its topology), we describe them as one single step of table extraction.

5.1 Table Extraction

The task of extracting web tables can be formulated as the the task of (1) finding all frames for a given web page; (2) discerning those which adhere to the definition of tables from section 3 and where a 2-D grid is semantically significant [25] from lists and other frames intended for non-relational layout purposes; (3) transfering the content into a topological grid description in which logical cells are flush with neighboring cells and their spatial relations are explicit. As explained in the section on related work, Yalin Wang and Hu [33] used a machine-learning based approach using features deduced from the HTML code to discern genuine from non-genuine tables. The difficulty of a HTML-centric approach is that important features of individual cells, like size and height, that are ideally suited to distinguish genuine from non-genuine `<table>` tables, or in our case, frames that represent tables from those that do not, are not explicitly available. Our approach can use such visual features, and the formal setup from section 4 allows us to describe a set of extraction rules and heuristics in an unambiguous way. Currently, we have defined over 20 such rules of which table 1 lists the most important 12.

We focus on a set of eight attributes for VENs (#1) and nine attributes for visualized words (#2). We only consider VENs with certain names, which we found form the dominant number of logical table cells on the Web (#3). Our approach focuses on tables which form a frame (#4) and eliminates duplicates of VENs which have the same coordinates (#5). Adjacency between neighboring VENs is loosely defined with a margin of 3 pixel (#6). The LocateFrames($\mathcal{V}'_e$) Algorithm 1 (#7) is a heuristics to locate all frames given a set of VENs. It builds upon the expansion algorithm Expand(*direction*, *frame*) from our previous work (Algorithm 1 from [15]) but works without any prior semantic word knowledge by expanding from all VENs except for those that were part of a previous expansion step. Two theoretical problems with this algorithm exist. In practical terms it works well. The expansion algorithm also implicitly considers a certain kind of spatial relationship which we consider as semantically ill-defined and, therefore, discriminatory against tables (#8). Another important algorithm (#9) is the cleaning algorithm CleanFrame(f). It deletes empty cells from the structure that convey just layout and no semantic meaning and as such just deletes cells whose disappearance does not change the visual semantic relations in a table. The result of this step is a structure with partial holes. We use a simple algorithm that works on the double topological grid to close the structure again. Figure 6 gives an intuition about the working of this grid transformation. The result of this step is again a set of completely tiled 2-D frames, which still do not necessarily represent tables. Rules #11 and #12 are two examples of several subsequent heuristics which try to discriminate tables based on layout characteristics.

Table 1: Twelve most important extraction rules and heuristics of VENTex.

#	Rules or heuristics	Explications
1	$\mathbf{a}^e_{prop} = \langle name,\ color,\ bgcolor,\ fsize,\ fstyle,\ fweight,\ ffamily,\ textalign \rangle$	Eight attributes of VENs are considered discriminatory enough
2	$\mathbf{a}^w_{prop} = \mathbf{a}^e_{prop} + \langle href \rangle$	Visualized words have an additional link attribute
3	$\mathcal{V}'_e \subset \mathcal{V}_e$ with $\mathcal{V}'_e = \{e \mid e \in \mathcal{V}_e, a^e_{name} \in \{\text{“TD”}, \text{“TH”}, \text{“DIV”}\}\}$	Only VENs with certain names are assumed possible logical table cells
4	Set of tables $\subseteq$ set of frames	Tables are assumed to form frames
5	PurgeCongruentVENs($\mathcal{V}'_e$)	Algorithm that eliminates congruent VENs
6	(a `x₁-adjacent` b) $\Leftrightarrow$ $(0 \leq x^1_1 - x^b_2 \leq 3\text{px})$	Adjacency between VENs is defined by their distance in each of the 4 directions
7	LocateFrames($\mathcal{V}'_e$)	Expansion Algorithm (see text)
8	$\nexists a, b \in \widehat{\mathcal{V}}^f_e \mid (x^a_1 < x^b_1 \wedge x^a_2 < x^b_2) \vee (y^a_1 < y^b_1 \wedge y^a_2 < y^b_2)$	A table cannot contain cells with overlapping projections
9	$\lvert \mathcal{X}^f_1 \rvert, \lvert \mathcal{X}^f_2 \rvert \geq 2,\ \lvert \mathcal{Y}^f_1 \rvert, \lvert \mathcal{Y}^f_2 \rvert \geq 3$	A table is 2-D and has at least 3 rows
10	CleanFrame(f)	Cleaning Algorithm (see text)
11	$\nexists e \in \widehat{\mathcal{V}}^f_e \mid Area^e > p_A \cdot Area^f$; $p_A = 0.4$	No single logical cell can cover more than 40% of a table
12	$\nexists e \in \widehat{\mathcal{V}}^f_e \mid \lvert\{w \mid e$ `contains` $w\}\rvert > p_w$; $p_w = 20$	No single logical cell can contain more than 20 words

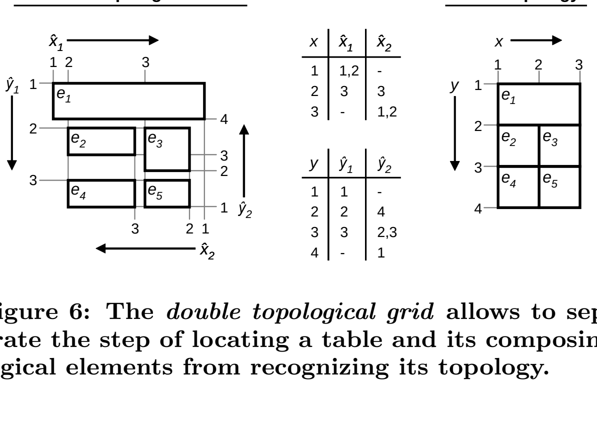

Figure 6: The *double topological grid* allows to separate the step of locating a table and its composing logical elements from recognizing its topology.

Algorithm 1 LocateFrames($\mathcal{V}_e$): locates all frames of a web page that are not part of another, bigger frame

```
Input: $\mathcal{V}_e$: set of visualized element nodes of web page
Return: $\mathcal{F}$: set of frames of web page
 1: $\mathcal{V}_e^* \leftarrow \mathcal{V}_e$
 2: $\mathcal{F} \leftarrow \{\}$
 3: for all $e \in \mathcal{V}_e^*$ do
 4:   $frame \leftarrow \langle \mathbf{x}^e, \{e\} \rangle$
 5:   $direction \leftarrow 0$
 6:   $fail \leftarrow 0$
 7:   repeat
 8:     repeat
 9:       $\langle frame, expandSuccess \rangle \leftarrow$ Expand($direction, frame$)
10:       if ($expandSuccess$ = true) $fail \leftarrow 0$ else $fail \leftarrow +1$
11:     until $expandSuccess$ = false
12:     $direction = (direction + 1)$ mod 4
13:   until $fail = 4$
14:   $\mathcal{V}_e^* \leftarrow \mathcal{V}_e^* - \mathcal{V}_e^{frame}$
15:   $\mathcal{F} \leftarrow \mathcal{F} \cup \mathcal{V}_e^{frame}$
16: end for
17: return $hList$
```

Algorithm 2 Clean(f): purges empty spacer columns and rows from a frame and candidate table f

```
Input: $\widehat{\mathcal{V}}_e^f$: set of extended VENs that form $f$
Return: $\widehat{\mathcal{V}}_e^f$: cleaned set
 1: construct double topological grid $\mathcal{X}$ for $\widehat{\mathcal{V}}_e^f$
 2: for all $x_1^* \in \mathcal{X}_1$ do
 3:   if all VENs with $x_1 = x_1^*$ are empty $\wedge$ have same $x_2^*$ then
 4:     delete these VENs from frame
 5:   end if
 6: end for
 7: for all $y_1^* \in \mathcal{Y}_1$ do
 8:   if all VENs with $y_1 = y_1^*$ are empty $\wedge$ have same $y_2^*$ then
 9:     delete these VENs from frame
10:   end if
11: end for
12: return $\widehat{\mathcal{V}}_e^f$
```

At the moment, these rules are still implemented in an ad-hoc fashion. On a first attempt, experimenting with a number of such heuristics enabled us to model the way humans recognize tables rather effectively, but still not perfectly. We think that considerable improvements can be made in a future step by casting the problem of optimizing these parameters into a machine-learning framework [33] which explicitly uses the visual features available in a 2-D topological problem formulation.

5.2 Table Interpretation

Two important decisions have to be made for the table interpretation step: (1) What is the appropriate output of a table interpretation, i.e. what kind of table model or table language should be used to describe information in tables? (2) How can the transition from the table topology into the model be automatized? Recent literature [13] suggests that the appropriate target format for table analysis is a representation based on the abstract table proposed by Xinxin Wang [32]. This table model is similar to the model of Vanoirbeek [31] in the sense that it is a multi-dimensional model of a table, separating the logical structure from the layout of a table. The distinctive characteristic of this model is that each entry does not have to be associated with labels of each of the dimensions (or categories) simultaneously, which allows to express composed tables. This logical model of data in tables was originally conceived to support the different stages of tabular composition, and we agree that it is a well suited format to further operate on the underlying information. We also think that it is possible for humans to transfer a given table to a Wang model most of the time, and that this process can be semi-automated for domain-specific tables. However, we think that the appropriate target format for automatic large-scale domain-independent information extraction and subsequent knowledge acquisition from web tables has to be more general as we will demonstrate with the help of Tables 2(a)-(b).

Table 2: Determining the number of dimensions of a table in the Wang model cannot be unambiguously made without an accompanying interpretation of the underlying domain.

(a)

		Price
A-Mart	Milk	1.19
	Sugar	0.59
	Bananas	1.99
B-Mart	Milk	1.29
	Sugar	0.59

(b)

		Price
A-Mart	#01124	1.19
	#01345	0.59
	#01347	1.99
B-Mart	pr78wh5	1.29
	pr62953	0.59

Each table shows the prices of products from two different stores A-Mart and B-Mart. Whereas Table 2(a) uses common names to specify the products, Table 2(b) uses product denominations which are shop-specific. Interpreting the tables along the thoughts of the Wang model, Table 2(a) is a three-dimensional table, having the categories shop, products, and prices. Table 2(b), however, is composed of two two-dimensional tables, each having the categories products and prices, with the name of each shop together with the product numbers building a labeled domain. Once, at a later stage, the shop-specific product names are mapped to a domain-specific naming scheme, Table 2(b) could be interpreted and queried like the other as three-dimensional table. Therefore, interpreting Tables 2(a)-(b) as either two-dimensional or three-dimensional at the moment of "parsing" and extracting the table jumps too early to a conclusion. As a domain-independent and automatic information extraction system should ideally make a single pass over its corpus guaranteeing scalability with the size of the corpus [4], such a content-specific and irrevocable decision is, in our opinion, better left to a later probabilistic information integration step. Therefore, we argue that the target format for domain-independent information extraction from web tables should not be the Wang model. Rather, it must be a format that allows to leave this final step to a later stage that analyzes the data space in connection with other information found on the Web. Our thinking is inspired by the work of Dalvi and Suciu [11] who propose to extend probabilistic databases by adding statistics on the global schema and probabilities to the view, thus allowing to compute probabilistic answers to queries instead of commonly used deterministic answers.

We believe that the appropriate format is a generalized n-tuple, which does not assign categories to each of the entries of the tuples. Therefore, these tuples cannot be seen as n-tuples in terms of the relational model of relational databases [1], but rather tuples whose relation between individual tuples and entries can be specified *a posteriori* of the extraction. Imagine the three-dimensional relational space spanned by the three entries of the three columns from Tables 2(a) -(b). The three entries of Table 2(a) are connected to both A-Mart and B-Mart, the three entries of Table 2(b), however, can be clearly separated. Imagine now the entry "Milk" replaced by the term "Super milk" everything else staying the same. The question of the number of dimensions of the table remains ambiguous, translating into only one single connections between A-Mart and B-Mart in the relational space. As such this representation also allows for a fuzzy, probabilistic notation of categories or dimensions. The end result of this extraction is an XML representation, which is heavily inspired by the work of Wohlberg [35]. Wohlberg's table model is almost identical to the Wang model. However, we do not use the same interpretation of the relations specified, we just use the file format and transfer all such nested structures in the form of nested sublabels.

We currently solve the issue of assigning a given table the according n-tupel relations by having first defined a number of most commonly found table types on the Web (similar to [38]) and then a number of discriminating heuristics which make heavy use of the style information contained in the attribute vector of the extended VENs. Currently these heuristics are combined in an ad-hoc fashion and work remains to be done to make this step more efficient.

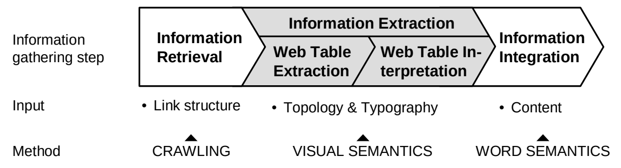

Figure 7: Domain-independent web table interpretation cannot resolve some domain-specific semantic ambiguities and therefore has to limit its scope of analysis to visual semantics.

5.3 Performance Analysis

Loading and rendering a page happens in $\mathcal{O}(n)$ time, where n is the number of element nodes as measure of a web page's size. It subsequently takes $\mathcal{O}(n\sqrt{n})$ time to construct the double topological grid. Whereas our previous expansion algorithm working with seed knowledge had complexity $\mathcal{O}(k\sqrt{n})$ where k is the number of appearances of a keyword on a page, we now have $\mathcal{O}(n\sqrt{n})$, which translates into a slight performance decline in practical terms.

Our current implementation extracts tables at an average speed of around 5 seconds per page. This puts our algorithm at a significant disadvantage to other approaches that work without rendering the page. We believe that, while this objection is true, future approaches will not be able to work on a broad scale without rendering web pages. E.g., identifying tabular structures from arbitrary web pages requires full rendering of the web page together with all linked style sheets and other sources in order to be able to work on the actual representation of data. Every image has its place holder and, as such, can contribute to the final spatial arrangement. Going from this assumption, major time improvements can be achieved by moving to faster rendering engines. But despite the slower performance due to rendering, we believe that we will witness a gradual shift in the IE community to rendering based information extraction in the time to come (Fig. 1).

6. WEB TABLE GROUND TRUTHING

During our research, we were trying to find available test data sets to evaluate the performance of our system. We did not find any useful and broad web table ground truth data set, which is why we started to build our own test data set. Now, while existing literature agrees that the ground truthing of interpreted tables in general is difficult because of several "truths" that might exist about what a table is and how it is interpreted [17], we also came across several problems in the steps prior to table interpretation.

One such obstacle is that it is not easy to create a permanent copy of available web pages as we explain in detail in [27]. In order to solve this task we built an open source Firefox extension named WebPageDump[1] which can handle the correct saving of most web pages. Another issue is that marking visual tables that can be built on any element node other than `<table>` tags is not trivial. Our method to ground truth web tables starts with making a local copy of the online web page and then generating different XML files that address each of the three steps of table understanding. Whereas the ground truthing of the table interpretation is limited by the model as explained in section 5.2, the other steps can be fully addressed. After saving local copies, we process the DOM tree of a web page and introduce unique identifiers for elements and words without changing the layout of the web page. Unique references to words are made possible by tokenizing text nodes and wrapping all words with a special `<x>` tag that does not change formatting properties in contrast to a `<span>` tag which sometimes inherits already assigned properties. This `<x>` tag enfolds every single word resulting in a differentiation on word-level in the subsequent XML documents, which allows this method to be applicable to all forms of web tables (Fig. 5). For delimiter-based web tables, we can combine single words to fictitious logical cells forming a logical entity. To be general on the grid level as well, we use the double topological grid structure, which allows us even to express such tables which are not aligned at the element node level (Fig. 6).

In order to objectively and transparently test our extraction system, we first wanted to create table ground truth from a really broad range of web tables. One important criterion was to eliminate our own influence on the web page selection and not to "make our scientific lives easier". So we asked students taking a class in web information extraction at Vienna University of Technology to provide us with a random selection of web tables, some of which we found were actual lists according to our definition from section 3. As result we received 493 web tables on 269 web pages provided by 63 students, which we ground truthed according to the aforementioned method. The test set of web pages together with the ground truth XML files will be available for download on the web page accompanying this publication[2].

7. EXPERIMENTAL RESULTS

We evaluated our system on the test set described in the previous section. We assume that a sample of web pages

[1]`http://www.dbai.tuwien.ac.at/user/pollak/webpagedump`
[2]`http://www.dbai.tuwien.ac.at/staff/gatter/ventex/`

with tables chosen by a number of people not involved in this research is sufficient to construct an unbiased and objective set of web tables on the Web.

Despite being a paper on tables, we think that a simple illustration can better convey the performance (recall r and precision p) of our system in its individual steps than a table. A table does show logical relationships between individual categories and its sublabels. However, it cannot convey comparisons within entries of the same category as illustratively as a graphic making an assertion. The recall of table extraction was 81%, precision 68%, recall of table interpretation was 57%, and precision at 48%.

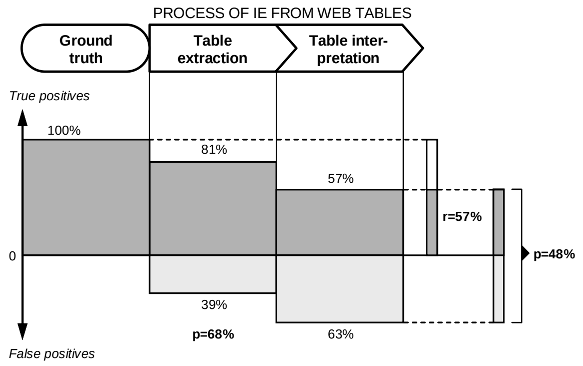

Figure 8: Experimental evaluation: True and false positives in each of the three table understanding steps as fraction of total HTML tables.

Due to the lack of standardized test sets, it is difficult to compare the results with existing solutions. Such a broad compilation of really diverse and sometimes even for humans difficult to understand web tables has – to our knowledge – not been tackled in the literature. The work of Wang and Hu [33], which we have already referenced several times, reports an approximate F-measure of 96% for table location after training a classifier on discriminating features from the source code of web pages. Our seemingly inferior result for table extraction can be set in context by observing: (1) their approach only works on leaf `<table>` tables, which we estimate to form less than 85% of current web tables; (2) The approach just locates tables, but does not recognize the actual relative topological relations between cells; and (3) our students wanted to, and did, challenge us. Information on web pages which are commonly more of interest to automated information extraction like those from price lists or comparison pages - which generally appeal to a larger audience and are highly ranked in search engine indexes - is commonly presented in a more “human interpretable” and, accordingly, “machine interpretable” form following the visual approach. Test runs on such web pages have led to far better results.

Still, we admit that our current results are far from being perfect test scores. And staying with our chosen test set, we think that any approach that intends to deal with such a variety of tables and varying implementation in the source code will need to use some kind of spatial reasoning on the rendered web page. We believe that there is no alternative than the visual path, but admit that details of implementation still leave space for future improvements.

8. FUTURE WORK

We are currently working to improve our approach in the following ways:

- **Table extraction.** We are still in the process of improving the quality of our table extraction step. We are experimenting in a number of ways to bring the defined heuristics close to the decision process that a human follows when spotting aligned information on a web page. We expect that a rule-based approach working on the correctly defined set of discriminating heuristics can bring the results close to human performance.

- **Table interpretation.** Whereas we think that the model we use is ideally fitted for our task, we are currently working on a limited base of table types. We are currently working on an improved table phenomenology, which incorporates a greater variety of tables found on web pages. Again, as we are working on distinctive visual characteristics, we also expect to be able to considerably improve our accuracy except for some very exotic cases, which would not play an important role for large-scale knowledge acquisition anyway.

- **Nested substructures.** Currently we do not consider any form of nested substructures whose dominant logical elements do not consist of atomic units (Fig. 3). We focus on locating dominant structures and regard their content as atomic units with words ordered in western reading order. Enlarging the scope of analysis also to such nested structures poses less a problem to the location and recognition, but rather the interpretation step. Our current model of tables (and other structured data) cannot be enlarged to this kind of nested data structures on the Web.

- **Other spatially and visually structured data.** So far, we have only focused on tables. We intend to enlarge the scale of this approach to other typologically and typographically structured data. An example is that of book list results on pages like Amazon. No spatial features are present, but distinctive typographical features like colors for attribute name and attribute value can be used for relational learning.

- **Information Integration.** Our described approach is only one step in a sequential series of three steps to large-scale knowledge acquisition from the Web (Fig. 7). One of our subtasks is developing the correct infrastructure that allows us to reason on the extracted information and attempt a probabilistic information integration from web tables.

9. CONCLUSIONS

In this paper, we formalized an unconventional and promising approach towards structured information extraction from the Web; in particular, from web tables. The approach uses a model of the visual representation of web pages as rendered by a web browser and, therefore, shifts the problem of information extraction from the lower level of code interpretation (HTML tag structure, CSS, JavaScript code, etc.) to the higher level of visual features (2-D topology and typography). We have also presented a model for representing web table structures along with algorithms to derive instances of the model given some arbitrary web pages. Our approach strives to perform well even without tuning for specific application domains such as the interpretation of product catalogues. We have shown this by providing a diverse test set of web tables that has been gathered by 63 students. Although our results are preliminary at the current state, we believe that applying a visual paradigm towards automatic information extraction from web tables is promising, especially given the rising complexity in the encoding of web pages on the source code level. Specifically, highly dynamic pages which tend to get more popular with the rise of Web 2.0 can not be processed without complex interpretation of the source code.

Acknowledgements. We like to thank Georg Gottlob for overall motivation of this research, Robert Baumgartner and his students for providing us with a challenging selection of web tables, and Tamir Hassan and the anonymous reviewers for helpful comments.

10. REFERENCES

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of databases.* Addison-Wesley, 1995.

[2] E. Agichtein and L. Gravano. *Snowball*: Extracting relations from large plain-text collections. In *Proc. 5th ACM DL*, pp. 85–94. ACM, June 2000.

[3] Y. Aumann, R. Feldman, Y. Liberzon, B. Rosenfeld, and J. Schler. Visual information extraction. *Knowledge and Information Systems*, 10(1):1–15, 2006.

[4] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. Open information extraction from the Web. In *Proc. 20th IJCAI*, pp. 2670–2676, Jan. 2007.

[5] M. Bilenko, S. Basu, and M. Sahami. Adaptive product normalization: Using online learning for record linkage in comparison shopping. In *Proc. 5th ICDM*, pp. 58–65. IEEE, Nov. 2005.

[6] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma. Extracting content structure for web pages based on visual representation. In *Proc. 5th APWeb*, pp. 406–417. Springer, Apr. 2003.

[7] H.-H. Chen, S.-C. Tsai, and J.-H. Tsai. Mining tables from large scale HTML texts. In *Proc. 18th COLING*, pp. 166–172. Morgan Kaufmann, Aug. 2000.

[8] W. W. Cohen, M. Hurst, and L. S. Jensen. A flexible learning system for wrapping tables and lists in HTML documents. In *Proc. 11th WWW*, pp. 232–241. ACM, May 2002.

[9] M. Cosulschi, N. Constantinescu, and M. Gabroveanu. Classification and comparison of information structures from a web page. *The Annals of the University of Craiova*, 31:109–121, 2004.

[10] A. Culotta, A. McCallum, and J. Betz. Integrating probabilistic extraction models and data mining to discover relations and patterns in text. In *Proc. HLT-NAACL*, pp. 296–303, New York, NY, June 2006.

[11] N. N. Dalvi and D. Suciu. Answering queries from statistics and probabilistic views. In *Proc. 31st VLDB*, pp. 805–816, Aug. 2005.

[12] D. W. Embley, M. Hurst, D. P. Lopresti, and G. Nagy. Table-processing paradigms: a research survey. *IJDAR*, 8(2-3):66–86, June 2006.

[13] D. W. Embley, D. P. Lopresti, and G. Nagy. Notes on contemporary table recognition. In *Proc. 7th Int. Workshop on Document Analysis Systems (DAS)*, pp. 164–175. Springer, Feb. 2006.

[14] O. Etzioni, M. J. Cafarella, D. Downey, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. Methods for domain-independent information extraction from the Web: An experimental comparison. In *Proc. 19th AAAI*, pp. 391–398. AAAI Press / MIT Press, July 2004.

[15] W. Gatterbauer and P. Bohunsky. Table extraction using spatial reasoning on the CSS2 visual box model. In *Proc. 21st AAAI*, pp. 1313–1318. AAAI Press, July 2006.

[16] X. Gu, J. Chen, W.-Y. Ma, and G. Chen. Visual based content understanding towards web adaptation. In *Proc. 2nd AH*, pp. 164–173. Springer, May 2002.

[17] J. Hu, R. S. Kashi, D. P. Lopresti, G. T. Wilfong, and G. Nagy. Why table ground-truthing is hard. In *Proc. 6th ICDAR*, pp. 129–133. IEEE, Sept. 2001.

[18] M. Hurst. Layout and language: Challenges for table understanding on the Web. In *Proc. 1st WDA at 6th ICDAR*, pp. 27–30, Sept. 2001.

[19] M. Kovacevic, M. Diligenti, M. Gori, and V. Milutinovic. Recognition of common areas in a web page using visual information: a possible application in a page classification. In *Proc. 2nd ICDM*, pp. 250–257. IEEE, Dec. 2002.

[20] B. Krüpl and M. Herzog. Visually guided bottom-up table detection and segmentation in web documents. In *Proc. 15th WWW*, pp. 933–934. ACM, May 2006.

[21] B. Krüpl, M. Herzog, and W. Gatterbauer. Using visual cues for extraction of tabular data from arbitrary HTML documents. In *Poster Proc. 14th WWW*, pp. 1000–1001. ACM, May 2005.

[22] K. Lerman, L. Getoor, S. Minton, and C. A. Knoblock. Using the structure of web sites for automatic segmentation of tables. In *Proc. SIGMOD*, pp. 119–130. ACM, June 2004.

[23] B. Liu and K. C.-C. Chang. Editorial: special issue on web content mining. *SIGKDD Explorations*, 6(2):1–4, 2004.

[24] B. Parsia and P. F. Patel-Schneider. Meaning and the Semantic Web. In *Proc. IRW at 15th WWW*, May 2006.

[25] G. Penn, J. Hu, H. Luo, and R. McDonald. Flexible web document analysis for delivery to narrow-bandwidth devices. In *Proc. 6th ICDAR*, pp. 1074–1078. IEEE, Sept. 2001.

[26] A. Pivk, P. Cimiano, and Y. Sure. From tables to frames. *Journal of Web Semantics*, 3(2-3):132–146, 2005.

[27] B. Pollak and W. Gatterbauer. Creating permanent test sets of web pages for information extraction research. In *Proc. 33rd SOFSEM: Theory and Practice of Computer Science*, vol. II, pp. 103–115, Jan. 2007.

[28] K. Simon and G. Lausen. ViPER: augmenting automatic information extraction with visual perceptions. In *Proc. 14th CIKM*, pp. 381–388. ACM, Nov. 2005.

[29] A. Tengli, Y. Yang, and N. L. Ma. Learning table extraction from examples. In *Proc. 20th COLING*, pp. 987–993. COLING, Aug. 2004.

[30] Y. A. Tijerino, D. W. Embley, D. W. Lonsdale, Y. Ding, and G. Nagy. Towards ontology generation from tables. *World Wide Web*, 8(3):261–285, 2005.

[31] C. Vanoirbeek. Formatting structured tables. In *Proc. of Electronic Publishing'92*, pp. 291–309. Cambridge University Press, Apr. 1992.

[32] X. Wang. *Tabular abstraction, editing, and formatting.* Ph.D. thesis, University of Waterloo, 1996.

[33] Y. Wang and J. Hu. A machine learning based approach for table detection on the Web. In *Proc. 11th WWW*, pp. 242–250. ACM, May 2002.

[34] H. Wium Lie, B. Bos, C. Lilley, and I. Jacobs. Cascading Style Sheets, level 2. Technical report, World Wide Web Consortium, 1998. See http://www.w3.org/TR/REC-CSS2.

[35] T. Wohlberg. Hypertables: Development of a structure description language for tables in XML. Master thesis, University of Hamburg, Germany, 1999. (Original title in German: Hypertables: Entwicklung einer Strukturbeschreibungssprache für Tabellen in XML).

[36] Y. Yang and W.-S. Luk. A framework for web table mining. In *Proc. 4th WIDM at 11th CIKM*, pp. 36–42. ACM, Nov. 2002.

[37] Y. Yang and H. Zhang. HTML page analysis based on visual cues. In *Proc. 6th ICDAR*, pp. 859–864. IEEE, Sept. 2001.

[38] M. Yoshida, K. Torisawa, and J. Tsujii. A method to integrate tables of the world wide web. In *Proc. 1st WDA at 6th ICDAR*, pp. 31–34, Sept. 2001.

[39] R. Zanibbi, D. Blostein, and J. R. Cordy. A survey of table recognition. *IJDAR*, 7(1):1–16, 2004.

[40] Y. Zhai and B. Liu. Web data extraction based on partial tree alignment. In *Proc. 14th WWW*, pp. 76–85. ACM, May 2005.

[41] H. Zhao, W. Meng, Z. Wu, V. Raghavan, and C. Yu. Fully automatic wrapper generation for search engines. In *Proc. 14th WWW*, pp. 66–75. ACM, May 2005.