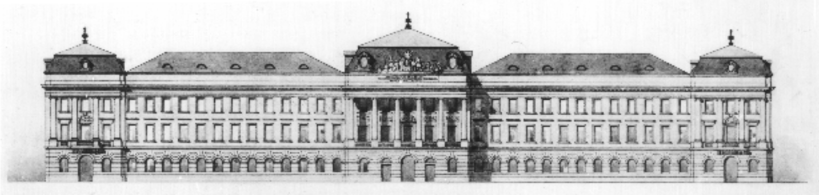
dbai

TECHNICAL

REPORT



INSTITUT FÜR INFORMATIONSSYSTEME

ABTEILUNG DATENBANKEN UND ARTIFICIAL INTELLIGENCE

Web Objects Identification for Web Automation: Objects and their Features

DBAI-TR-2013-80

Ruslan R. Fayzrakhmanov **Christoph Herzog**
Iraklis Kordomatis

Institut für Informationssysteme
Abteilung Datenbanken und
Artificial Intelligence
Technische Universität Wien
Favoritenstr. 9
A-1040 Vienna, Austria
Tel: +43-1-58801-18403
Fax: +43-1-58801-18493
sekret@dbai.tuwien.ac.at
www.dbai.tuwien.ac.at

DBAI TECHNICAL REPORT
2013

TECHNISCHE
UNIVERSITÄT
WIEN
Vienna University of Technology

DBAI TECHNICAL REPORT

DBAI TECHNICAL REPORT DBAI-TR-2013-80, 2013

Web Objects Identification for Web Automation: Objects and their Features

Ruslan R. Fayzrakhmanov [1] **Christoph Herzog** [2]
Iraklis Kordomatis [3]

Abstract. This report describes all features used in the web object identification problem considered within the scope of the TAMCROW project. We also give a short description of the Unified Ontological Model which underlie the web page analysis and feature computation. Moreover, the ATW dataset, which contains annotated web pages from the transportation search domain, is described in detail.

[1] Institute for Information Systems, Vienna University of Technology, Favoritenstrasse 9-11, 1040 Vienna, Austria. E-mail: fayzrakh@dbai.tuwien.ac.at

[2] Institute for Information Systems, Vienna University of Technology, Favoritenstrasse 9-11, 1040 Vienna, Austria. E-mail: cherzog@dbai.tuwien.ac.at

[3] Institute for Information Systems, Vienna University of Technology, Favoritenstrasse 9-11, 1040 Vienna, Austria. E-mail: kordomatis@dbai.tuwien.ac.at

Acknowledgements: This work is funded by the Austrian Forschungsförderungsgesellschaft FFG under grant 829614 (TAMCROW).

Contents

1 Introduction

Web pages consist of stereotypical elements. Objects such as menus, navigation bars, web forms and forums threads are common to a large number of web pages. Automatic recognition of these elements plays an important role in various areas of science and technology, either expressly or by implication, be it web GUI testing, web accessibility, web data extraction, or web form understanding for querying meta-search engines [13]. Different design goals and different authoring tools cause infinite variability in the representation of these figures at the technical level, represented mostly by X/HTML, CSS and Javascript [16]. Our goal is to tackle this recognition problem on the level that it is coded for: human visual perception [12, 15, 16]. We try to grasp the main spatial, visual and textual characteristics, which allows us to abstract from "decorative" design variations as well as from the technical interpretation. The first results in the direction of approaching a challenge of simple web object identification are presented in [13].

The rest of the report is organized as follows: In Section 2, we give a short description of the Unified Ontological Model (UOM) that underlies the methods of web page analysis, web information extraction and web page understanding introduced within the scope of the TAMCROW [2] and ABBA [1] projects. Section 3 describes main objects of a web page structure, and in Section 4 we list all features which are used in the challenge of web object identification. Section 5 presents main distance metrics for different features, which are used to detect a similarity of different objects. In Sections 6 and 7 we give a detail description of the ATW dataset which contains annotated web pages together with computed features and feature distances.

2 The Unified Ontological Model

The Unified Ontological Model (UOM) [10, 11, 16] is a domain-specific ontology developed within the scope of ABBA [1] and TAMCROW [2] projects and used in the tasks of web information extraction and web page understanding. The version of the UOM presented in this section is an implementation integrated into the WPPS framework [9, 10] intended for development of various methods of web page analysis.

The UOM consists of two main sub-models (cf. Figure 1): *physical* and *logical models*.

The physical model comprises the following: DOM^*, the Interface Model (IM), the Block-based Geometric Model (BGM), and the Visual Perception Model (VPM). It also can be easily extended with additional features and relations. DOM^* describes DOM trees and computed CSS attributes of a web page along with its frames' hierarchy and represents them as one single labelled ordered tree. This structure is convenient for reasoning and analysis that factor in a web page's segmentation established by the source code.

The IM represents information derived from the DOM^* and contains the following objects: ○ web forms and their elements (labels, buttons, text fields, etc.); ○ images; ○ hyper-links; ○ interactive elements (which have listeners attached); ○ structures such as lists and tables, which can be derived from the computed CSS attribute `display`.

The BGM [8] models a web page's layout, leveraging spatial features and relationships. Its main element is the block, which is used as a minimum bounding rectangle for visualised elements.

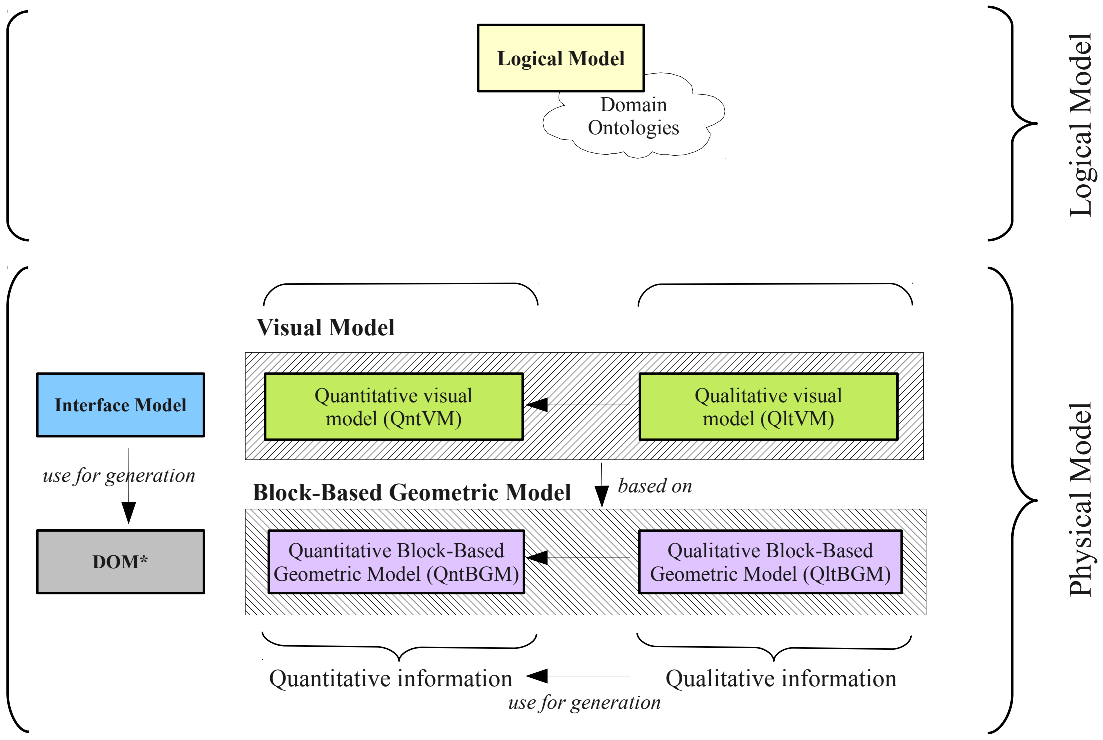

Logical Model
Domain Ontologies
Logical Model
Visual Model
Interface Model
Quantitative visual model (QntVM)
Qualitative visual model (QltVM)
use for generation
based on
Block-Based Geometric Model
DOM*
Quantitative Block-Based Geometric Model (QntBGM)
Qualitative Block-Based Geometric Model (QltBGM)
Quantitative information
Qualitative information
use for generation
Physical Model

Figure 1: The Unified Ontological Model

Blocks are used to approximate objects such as a web page, CSS box, an arbitrary set of elements and areas on a web page's canvas. The BGM consists of a structural BGM (SBGM), a quantitative (QntBGM) and qualitative (QltBGM) BGMs. The SBGM represents a structure of a web page's layout derived from the analysis of DOM^*.

QntBGM contains data such as position of a block, distance, width, height (in px), and direction (in degrees). QltBGM contains information about the layout represented using linguistic variables. The main features of a block in this sub-model are height and width. For representing spatial relationships and spatial configurations we use topological relations introduced in RCC8 algebra[7], distance, direction, and alignment. Based on our investigation and the fact that most of the web pages follow near-Manhattan layout, we have found rectangular cardinal directions [17] suitable for representing direction relationships. Furthermore, relations such as RCC8, alignment (except centring), and direction are expressed via two-dimensional interval relations (2DIRs) [6] that allows one to avoid inconsistency in the UOM. Thereby 2DIRs are fundamental relations in this case.

A visual perception model (VPM), which consists of structural VPM, quantitative and qualitative VPM, represents different perceptual information such as saliency (degree of uniqueness), emphasized (bold, italic, underlined) and various groupings that can be made based on the Gestalt theory [16].

The logical model provides an interpretation of the concepts in the physical model. It has

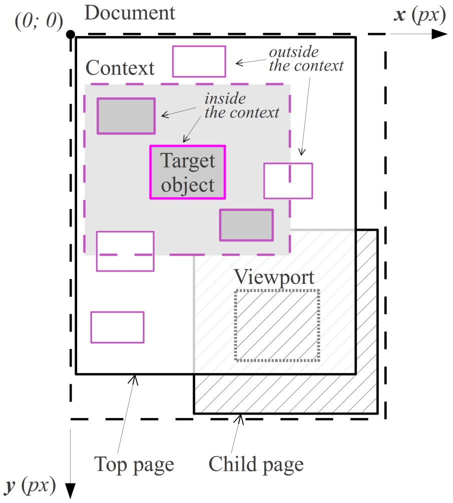


Figure 2: Structural objects of a web page

a vocabulary to represent different data structures such as sequence, tree, grid, etc. Applying external ontologies and referring to the linked data (e.g. DBpedia), it is possible to define web specific objects (e.g. navigation menu, main content, header) and domain specific objects (e.g. departure date, posting date, name) identified on a web page. The logical model is a contribution to the development of Semantic Web technology providing information about a web page's content that is accessible to computers in the Internet for further automatic analysis and reasoning.

3 Structural Elements

We define several main structural web page elements (objects) for the object identification problem (c.f. Figure 2): *document*, *page*, *selected* (*target*) *object*, and its *context*. These objects have their counterparts in the Unified Ontological Model (UOM) which is acquired in the process of web page analysis [9] and provides us with necessary information for computing features considered in this work. Every object is associated with some rectangular area that wraps certain set of other objects on a web page canvas. A geometric space of a web page is Euclidean space with pixels as a unit of measure. Origin of coordinates and direction of axes is depicted in Figure 2. The rectangular area $r \in R$ of a web page's structural objects is defined by its top-left (x^-, y^-) and bottom-right (x^+, y^+) extrime points, R is a type representing all the set of rectangles. A rectangle has its counterpart, *block*, in the Block-based Geometric Model (BGM) [8] of the UOM.

For every object, the containment function ρ is defined that maps set of objects to the power set of CSS boxes. This function reflects the spatial containment relationship, in particular P^{-1} relation from the Rectangular Connection Calculus (RCC) [18].

3.1 Document

A *document* is a web page rendered by the web browser's engine. It consists of a set of HTML, XHTML, or XML files forming a hierarchy of frames. The size of a document is defined as a minimal bounding rectangle of all the pages connected through frames.

3.2 Page

A *page* is a part of document, one of X/HTML or XML files (rendered by the web browser engine) which build either top level DOM window or DOM windows of the frames in the hierarchy of frames. In other words, it is a DOM tree together with computed CSS attributes. A page has its counterpart, *window*, in the Browser Object Model (BOM) [14]. The top-left corner of the top level page (p_t) is located in the origin of coordinates, i.e. $x^-_{p_t} = y^-_{p_t} = 0$.

3.3 Selected Object

We consider an object to be identified (hereafter referred to as *selected object*) as a certain rectangular area on a document, which corresponds to one or several CSS boxes and it is their minimal bounding rectangle. For the simplicity, we identify an object with a CSS box. Thus, given selected object $o \in O$, containment function $\rho_o : O \rightarrow B$ maps a set of selected objects O to the set of the CSS boxes B in a document. There are following possible types of selected objects considered in the TAMCROW project: `HtmlButton`, `HtmlCheckbox`, `HtmlFileUpload`, `HtmlImage`, `HtmlPasswordInput`, `HtmlRadiobutton`, `HtmlSelect`, `HtmlText`, `HtmlTextArea`, `HtmlTextInput`. All these have corresponding object types in the CSS model [3] of a web page and in the Interface Model (IM) accordingly.

3.4 Object's Context

The *context* of the selected object is a rectangular area which bounds the selected object together with its neighborhood. Given a context $c_o = c(o)$ for the selected object $o \in O$, containment function $\rho_c : C \rightarrow 2^B$, where C is a set of possible contexts in a document, B is a set of CSS boxes. In our implementation, contexts are those objects that are in P^{-1} relation [18] with the selected object and which contains the CSS boxes from the same *page* as a selected object. $\rho^-_c(c_o) = \rho_c(c_o) \setminus \rho_o(o)$.

Position and spatial expansion of the context c_o is defined by $\nu : O \rightarrow R$. We define the context's region to be equal to $2h \times 1.4w$, where h is height and w is width of the selected object; the minimal allowed size is $h + 500 \times w + 500$ pixels.

4 Features

Object features describe different aspects of structural web page objects and provide us with necessary information which is used in the task of object identification.

Depending on the considered elements of a web page during the object's features computation, we distinguish *inherent* (*intrinsic*) and *relative* features. The former describe certain characteristics of objects (e.g. of the selected object, context, page etc.) and are computed independently from other structural objects. Inherent features are only based on the computed styles of the contained CSS boxes or the attributes of the counterparts in the Browser Object Model (BOM). Examples are font color, tag name, height or font size. The relative features in their turn are defined subject to consideration of attributes of several structural objects, e.g. size of the certain context relative to size of a document, or average weighted distance color between the selected object and all other objects within the context, etc. Relative features are considered for a pair of structural elements, e.g. a selected object and its context or a selected object and the whole document, etc.

Depending on the considered web page's aspect, we single out four conventional categories of features (one feature can be referred to several categories):

- *Interface features* (IF) are relevant to the graphical user interface design, and covers characteristics regarding functional role of object (button, image, text, etc.) and structural types such as list and table. Computation of the features of this type is mainly based on the Interface Model (IM).

- *Spatial features* (SF), e.g. absolute position of an object, its size or the number of elements aligned with the selected object within the context. Spatial features are computed based on basic quantitative and qualitative spatial features and relations from the Block-based Geometric Model (BGM) [8], in particular, alignment, distance, topology, direction, etc. All qualitative spatial relations which are used for the feature computation are computed with inaccuracy $\varepsilon = 0.4$px, which was defined experimentally.

- *Visual perception features* (VPF), e.g. foreground and background colors, emphasis, font size, average weighted foreground color distance. These features reflect visual characteristics of objects and correspond to the fields of graphical user interface design, computer graphics and computer vision. The Visual Perception Model (VPM) and IM are mainly used for the computation of features of this type.

- *Textual features* (TF), e.g. textual content of the selected object, text above the object, text under, text of the nearest orthogonally visible objects, character density of links, number of lines, number of tokens, etc. Most of the textual features are adopted from the area of quantitative linguistics [19]. The IM and BGM are mainly considered during the computation of the features of this type.

The main groups of features, considered in the project TAMCROW [2] for the object identification problem, are depicted in Figure 3 and explained in detail in the following subsections.

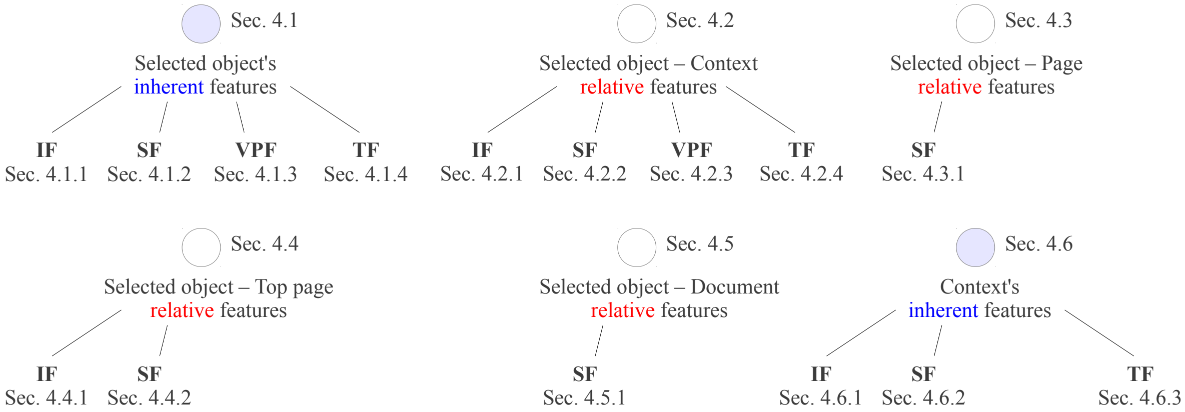


Figure 3: List of considered groups of features

4.1 Selected Object's Inherent Features

4.1.1 Interface Features

Object type is a type of selected object.
Category: Nominal.
Domain: $S \equiv$ `HtmlButton` $\sqcup$ `HtmlCheckbox` $\sqcup$ `HtmlFileUpload` $\sqcup$ `HtmlImage` $\sqcup$ `HtmlPasswordInput` $\sqcup$ `HtmlRadiobutton` $\sqcup$ `HtmlSelect` $\sqcup$ `HtmlText` $\sqcup$ `HtmlTextArea` $\sqcup$ `HtmlTextInput`.
Range: {`HtmlButton`, `HtmlCheckbox`, `HtmlFileUpload`, `HtmlImage`, `HtmlPasswordInput`, `HtmlRadiobutton`, `HtmlSelect`, `HtmlText`, `HtmlTextArea`, `HtmlTextInput`}.
Java data type: `Enum`.
System name: `TypeIO`.

Editable defines whether element can be edited by the user or not.
Category: Binary.
Domain: S.
Range: $\{true, false\}$
Java data type: `Boolean`.
System name: `EditableIO`.
Algorithm: Objects of a type `HtmlCheckbox`, `HtmlFileUpload`, `HtmlPasswordInput`, `HtmlRadiobutton`, `HtmlSelect`, `HtmlTextArea`, or `HtmlTextInput` are considered as editable; DOM element with attribute $contenteditable = true$ is also considered as editable.

Selection reflects the state of elements such as check box and radio button.
Category: Binary.
Domain: `HtmlCheckbox`, `HtmlRadiobutton`.
Range: $\{true, false\}$
Java data type: `Boolean`.
System name: `SelectedIO`.

4.1.2 Spatial Features

Area is an area of the minimum bounding rectangle of the corresponding selected object; $r_o = w_o \cdot h_o$.
Category: Numeric.
Domain: S.
Range: $[0, \infty)$
Java data type: `Real` .
System name: `AreaIO` .

Aspect ratio : $width/height$.
Category: Numeric.
Domain: S.
Range: $[0, \infty)$
Java data type: `Real`.
System name: `AspectRatioIO`.

4.1.3 Visual Perception Features

Foreground color is applied to those objects that contain textual information.
Category: Numeric.
Domain: `HtmlButton` $\sqcup$ `HtmlFileUpload` $\sqcup$ `HtmlSelect` $\sqcup$ `HtmlText` $\sqcup$ `HtmlTextArea` $\sqcup$ `HtmlTextInput` .
Range: $\{0, 1, 2, \ldots, \infty\}$. The representation format of the feature is RGBA[1].
Java data type: `Integer` .
System name: `ForegroundColorIO`.

Background color is applied to those objects which do not have image or any other multimedia object as their background.
Category: Numeric.
Domain: $S \sqcup \forall hasBGSRGBColor.\top$.
Range: $\{0, 1, 2, \ldots, \infty\}$. The representation format of the feature is RGBA.
Java data type: `Integer`.
System name: `BackgroundColorIO`.
Computation:
Going along the sequence of objects ordered by their `drawId` property in ascending order, for every object with transparent background color, `hasBGSRGBColor` property is set to be equal to the last considered CSS box which overlaps with current object. If such an object has image as its background, than we leave current object with transparent background. If the object with transparent background does not have any overlapped antecedents it inherits color of the root element of the DOM tree. `drawId` property is contained in the BGM [8]; it defines a draw order of CSS

[1] `http://en.wikipedia.org/wiki/RGBA_color_space`

boxes according to the CSS 2.1 Specification [3]. Developed algorithm for computing `drawId` property: `http://code.google.com/p/css-drawing-order-detection/`.

Emphasis reflects a presence of text formatting. Features such as font weight w, font style s, text decoration d from IM are considered.
Category: numeric.
Domain: `HtmlButton`, `HtmlFileUpload`, `HtmlSelect`, `HtmlText`, `HtmlTextArea`, `HtmlTextInput`.
Range: $(0.(6); 2.(2))$. 1 is a normal text without text decoration applied.
Java data type: `Double`.
System name: `EmphasisIO`.
Computation:
Value is computed according to the formula $(w_n + s_n + d_n)/3$.

$$w_n = \begin{cases} 1 - (400 - w)/300 & \text{if } w \leq 400, \\ (w - 400)/300 + 1 & \text{if } w \geq 400; \end{cases}$$

where w is a font weight, $w \in \{100, 200, \dots, 900\}$; 400 corresponds to the normal text without "weight", 700 is a *bold* text.

$$s_n = \begin{cases} 1 & \text{if } s = \mathit{normal}, \\ 2 & \text{otherwise}; \end{cases}$$

where s_n is a font style.

$$d_n = \begin{cases} 1 & \text{if } d = \mathit{none}, \\ 2 & \text{otherwise}; \end{cases}$$

where d_n is a text decoration.

Font size
Category: Numeric.
Domain: `HtmlButton`, `HtmlFileUpload`, `HtmlSelect`, `HtmlText`, `HtmlTextArea`, `HtmlTextInput`.
Range: $[0; \infty)$.
Java data type: `Double`.
System name: `FontSizeIO`.

4.1.4 Textual Features

Text of selected object
Category: Nominal.
Domain: `HtmlImage`, `HtmlButton`, `HtmlFileUpload`, `HtmlSelect`, `HtmlText`, `HtmlTextArea`, `HtmlTextInput`.
Range: λ^*
Java data type: `String`.
System name: `TextIO`.

Number of lines is a number of rows taken by the selected object. To compute number of lines client rectangles are considered [5].
Category: Numeric.
Domain: S.
Range: $\mathbb{N}$.
Java data type: `Integer`.
System name: `LinesQntIO`.

Number of tokens is a number of tokens in the selected object. Tokens are extracted by the regular expression $(\backslash s|\&nbsp;|\backslash u0020|\backslash u00A0|\backslash u200B|\backslash u3000)*$, which takes into account peculiarities of HTML and UNICODE.
Category: Numeric.
Domain: S.
Range: $\{0, 1, 2, \ldots, \infty\}$.
Java data type: `Integer`.
System name: `TokensQntIO`.

4.2 Selected Object – Context Relative Features

4.2.1 Interface Features

Type of the dominant orthogonally visible object is the most common type among the orthogonally visible objects (cf. Figure. 4).
Category: Nominal.
Domain: S.
Range: {`HtmlButton`, `HtmlCheckbox`, `HtmlFileUpload`, `HtmlImage`, `HtmlPasswordInput`, `HtmlRadiobutton`, `HtmlSelect`, `HtmlText`, `HtmlTextArea`, `HtmlTextInput`}.
Java data type: `Enum`.
System name: `DominantOrthogonalVisibleTypeROC`.

Number of similar types is quantity of objects of the type similar to the selected object's type, within the context.
Category: Numeric.
Domain: S.
Range: $\{0, 1, 2, \ldots, \infty\}$.
Java data type: `Integer`.
System name: `SimilarTypesQntROC`.

4.2.2 Spatial Features

Number of alignments is quantity of objects which has any of alignment relations with selected object. $AlignmentQntROC = AlignmentHorQntROC + AlignmentVertROC$
Category: Numeric.
Domain: S.

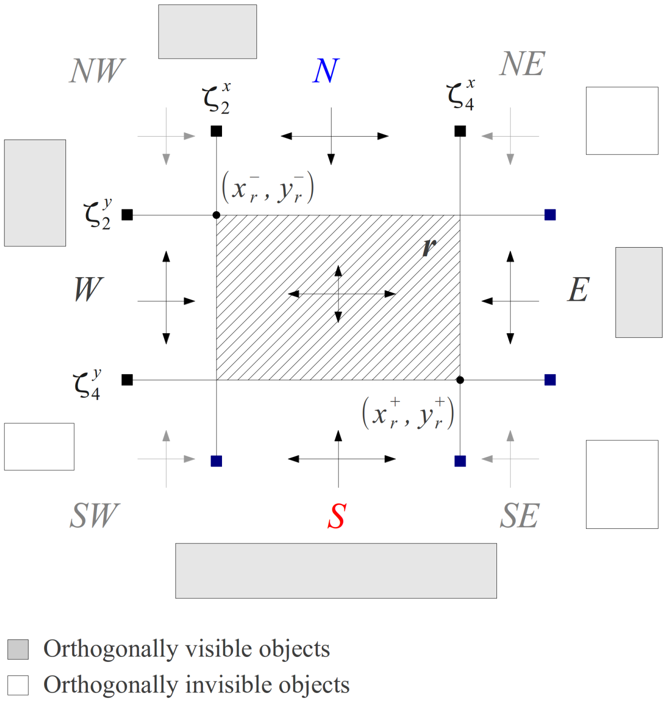


Figure 4: Orthogonal visibility

Range: $\{0, 1, 2, \ldots, \infty\}$.
Java data type: `Integer`.
System name: `AlignmentQntROC`.

Number of horizontal alignments is quantity of objects which has any horizontal alignment relations with selected object.
Category: Numeric.
Domain: S.
Range: $\{0, 1, 2, \ldots, \infty\}$.
Java data type: `Integer`.
System name: `AlignmentHorQntROC`.

Horizontal alignment index is an index of the selected object in the horizontally aligned sequence of CSS boxes within the context.
Category: Numeric.
Domain: S.
Range: $\{0, 1, 2, \ldots, \infty\}$.
Java data type: `Integer`.
System name: `AlignmentIndexHorROC`.

Number of vertical alignments is quantity of objects which has any vertical alignment relations with selected object.

Category: Numeric.
Domain: S.
Range: $\{0, 1, 2, \dots, \infty\}$.
Java data type: `Integer`.
System name: `AlignmentVertQntROC`.

Vertical alignment index is an index of the selected object in the vertically aligned sequence of CSS-boxes.
Category: Numeric.
Domain: S.
Range: $\{0, 1, 2, \dots, \infty\}$.
Java data type: `Integer`.
System name: `AlignmentIndexVertROC`.

Alignment factor : $(AlignmentQntROC + 1)/(TObjectsQntIC + 1)$.
Category: Numeric.
Domain: S.
Range: $(0; \infty)$
Java data type: `Double`.
System name: `AlignmentFactorROC`.

Ratio of vertical to horizontal alignments :
$(AlignmentVertQntROC + 1)/(AlignmentHorQntROC + 1)$.
Category: Numeric.
Domain: S.
Range: $(0; \infty)$
Java data type: `Double`.
System name: `AlignmentVertHorRatioROC`.

Number of orthogonally visible objects : cf. Figure 4.
Category: Numeric.
Domain: S.
Range: $\{0, 1, 2, \dots, \infty\}$
Java data type: `Integer`.
System name: `OrthogonalVisibleObjQntROC`.

Number of orthogonally visible aligned objects : cf. Figure 4.
Category: Numeric.
Domain: S.
Range: $\{0, 1, \dots, 12\}$
Java data type: `Integer`.
System name: `AlignedOrthogonalVisibleObjQntROC`.

Number of orthogonally visible fully aligned objects : cf. Figure 4. Fully aligned objects are those which has all types of vertical or horizontal alignment relations with the selected object.

$FullyAlignedOrthogonalVisibleObjQntROC \leq AlignedOrthogonalVisibleObjQntROC \leq OrthogonalVisibleObjQntROC$.
Category: Numeric.
Domain: S.
Range: $\{0, 1, 2, 3, 4\}$
Java data type: `Integer`.
System name: `FullyAlignedOrthogonalVisibleObjQntROC`.

Pixels to character ratio is an average area (in squared pixels) which is occupied by a character within the context.
Category: Numeric.
Domain: `HtmlText`
Range: $[0; \infty)$
Java data type: `Double`.
System name: `PixelsToCharacterIC`.

4.2.3 Visual Perception Features

Average weighted foreground color distance is an average relative color distance between foreground color (feature `ForegroundColorIO`) of the selected object and foreground color of other CSS boxes within the context. During the computation size (`AreaIO` feature) of objects is taken into account.
Category: Numeric.
Domain: `HtmlButton`, `HtmlFileUpload`, `HtmlSelect`, `HtmlText`, `HtmlTextArea`, `HtmlTextInput`.
Range: $[0; \infty)$.
Java data type: `Double`.
System name: `AvgWeightedFGColorROC`.
Computation:

$$\frac{\sum\limits_{o \in c(s)} \Delta HSV(color(s), color(o)) \cdot S_o}{\sum\limits_{o \in c(s)} S_o}, \tag{1}$$

where ΔHSV is an HSV color distance between selected object s and object (CSS box) o within the context $c(o)$; S_o is an area of o.

Average weighted background color distance is applied to those objects which do not have image or any other multimedia objects as their background. It is an average relative color distance between background color (feature `BackgroundColorIO`) of the selected object and background color of other CSS boxes within the context (1). During the computation size (`AreaIO` feature) of objects is taken into account.

Category: Numeric.
Domain: $S \sqcup \forall hasBGSRGBColor.\top$.
Range: $[0;\infty)$.
Java data type: `Double`.
System name: `AvgWeightedBGColorROC`.

4.2.4 Textual Features

Text above is formed by the horizontally ordered orthogonally visible CSS boxes (cf. Figure. 4) above the selected object.
Category: Nominal.
Domain: `HtmlButton`, `HtmlFileUpload`, `HtmlSelect`, `HtmlText`, `HtmlTextArea`, `HtmlTextInput`.
Range: λ^*
Java data type: `String`.
System name: `UpperTxtOfOrthVisibleObjsROC`.

Text on the right is formed by the vertically ordered orthogonally visible CSS boxes (cf. Figure. 4) to the right of the selected object.
Category: Nominal.
Domain: `HtmlButton`, `HtmlFileUpload`, `HtmlSelect`, `HtmlText`, `HtmlTextArea`, `HtmlTextInput`.
Range: λ^*
Java data type: `String`.
System name: `RightTxtOfOrthVisibleObjsROC`.

Text under is formed by the horizontally ordered orthogonally visible CSS boxes (cf. Figure. 4) under the selected object.
Category: Nominal.
Domain: `HtmlButton`, `HtmlFileUpload`, `HtmlSelect`, `HtmlText`, `HtmlTextArea`, `HtmlTextInput`.
Range: λ^*
Java data type: `String`.
System name: `BottomTxtOfOrthVisibleObjsROCF`.

Text on the left is formed by the vertically ordered orthogonally visible CSS boxes (cf. Figure. 4) to the left of the selected object.
Category: Nominal.
Domain: `HtmlButton`, `HtmlFileUpload`, `HtmlSelect`, `HtmlText`, `HtmlTextArea`, `HtmlTextInput`.
Range: λ^*
Java data type: `String`.
System name: `LeftTxtOfOrthVisibleObjsROC`.

Text of the nearest orthogonally visible objects : cf. Figure 4, page 12.
Category: Nominal.
Domain: HtmlButton, HtmlFileUpload, HtmlSelect, HtmlText, HtmlTextArea, HtmlTextInput.
Range: λ^*
Java data type: String.
System name: TextOfNearestOrthVisibleObjsROC.

Nearest text is a text of the element nearest to the selected object within the context.
Category: Nominal.
Domain: HtmlButton, HtmlFileUpload, HtmlSelect, HtmlText, HtmlTextArea, HtmlTextInput.
Range: λ^*
Java data type: String.
System name: TextOfNearestTxtObjROC.

4.3 Selected Object – Page Relative Features

4.3.1 Spatial Features

Relative width is width of the selected object relative to the corresponding page.
Category: Numeric.
Domain: S.
Range: $[0; \infty)$.
Java data type: Double.
System name: RelativeWidthROW.

Relative height is height of the selected object relative to the corresponding page.
Category: Numeric.
Domain: S.
Range: $[0; \infty)$.
Java data type: Double.
System name: RelativeHeightROW.

Relative x-position is x-coordinate of the left-top corner of the selected object relative to location and width of the corresponding page.
Category: Numeric.
Domain: S.
Range: $[0; \infty)$.
Java data type: Double.
System name: RelativeXPositionROW.

Relative y-position is y-coordinate of the left-top corner of the selected object relative to location and height of the corresponding page.

Category: Numeric.
Domain: S.
Range: $[0; \infty)$.
Java data type: `Double`.
System name: `RelativeYPositionROW`.

4.4 Selected Object – Top Page Relative Features

4.4.1 Interface Features

Link type is type of link which is computed relatively to the URL of the document (top web page).
Category: Nominal.
Domain: S.
Range: $\{local, domain, external\}$.
Java data type: `Enum`.
System name: `LinkTypeROTW`.

4.4.2 Spatial Features

Grid location specifies roughly which areas of the web page a given object touches. The web page is divided into a 3×3 symmetric grid, resulting in 9 equal areas that the object can possibly "touch" [13].
Category: Numeric.
Domain: S.
Range: $[0; 511]$. The value is represented as a bitmap where every bit relates to the specific cell.
Java data type: `Double`.
System name: `GridLocationX3ROTW`.

4.5 Selected Object – Document Relative Features

4.5.1 Spatial Features

Number of alignments is a quantity of objects (within the document) which has any of alignment relations with selected object. $AlignmentQntROD = AlignmentHorQntROD + AlignmentVertQntROD$
Category: Numeric.
Domain: S.
Range: $\{0, 1, 2, \ldots, \infty\}$.
Java data type: `Integer`.
System name: `AlignmentQntROD`.

Number of horizontal alignments is a quantity of objects (within the document) which has any horizontal alignment relations with selected object.
Category: Numeric.

Domain: S.
Range: $\{0, 1, 2, \ldots, \infty\}$.
Java data type: `Integer`.
System name: `AlignmentHorQntROD`.

Horizontal alignment index is an index of the selected object in the horizontally aligned sequence of CSS boxes (within the document).
Category: Numeric.
Domain: S.
Range: $\{0, 1, 2, \ldots, \infty\}$.
Java data type: `Integer`.
System name: `AlignmentIndexHorROD`.

Number of vertical alignments is a quantity of objects (within the document) which has any vertical alignment relations with selected object.
Category: Numeric.
Domain: S.
Range: $\{0, 1, 2, \ldots, \infty\}$.
Java data type: `Integer`.
System name: `AlignmentVertQntROD`.

Vertical alignment index is an index of the selected object in the vertically aligned sequence of CSS boxes (within the document).
Category: Numeric.
Domain: S.
Range: $\{0, 1, 2, \ldots, \infty\}$.
Java data type: `Integer`.
System name: `AlignmentIndexVertROD`.

Ratio of vertical to horizontal alignments :
$(AlignmentVertQntROD + 1)/(AlignmentHorQntROD + 1)$.
Category: Numeric.
Domain: S.
Range: $(0; \infty)$.
Java data type: `Double`.
System name: `AlignmentVertHorRatioROD`.

4.6 Context Inherent Features

4.6.1 Interface Features

Object number is a quantity of objects contained in the context c_o except selected object ($\rho_c^-(c_o)$).
Category: Numeric.
Domain: Context.

Range: $\{0, 1, 2, \ldots, \infty\}$.
Java data type: `Integer`.
System name: `TObjectsQntIC`.

4.6.2 Spatial Features

Spatial density of text is an area taken by text divided by the area of context. To compute area of links client rectangles are considered [5].
Category: Numeric.
Domain: Context.
Range: $[0; 1]$.
Java data type: `Double`.
System name: `TextSpatialDensityIC`.

Spatial density of links is an area taken by the links divided by the area of context. To compute area of links client rectangles are considered [5].
Category: Numeric.
Domain: Context.
Range: $[0; 1]$.
Java data type: `Double`.
System name: `LinkSpatialDensityIC`.

4.6.3 Textual Features

Character density of links is ratio of characters in links to all characters in the context, excluding spaces.
Category: Numeric.
Domain: Context.
Range: $[0; 1]$.
Java data type: `Double`.
System name: `LinkCharacterDensityIC`.

5 Distance computaton

The following paragraphs explain the basic distance calculation formulas. Each feature has a corresponding feature distance. The *feature distance vector* is the vector which results from the calculation of each of the feature distances in two given feature vectors—an array of the features for a given selected object. Since not all features are applicable to all types of objects, we sometimes encounter `null` values in feature vectors. By definition we assign the maximal distance to distance computations where one or more of the input values are `null`. The actual maximal distance value depends on the specific distance used. In this way we avoid `null` values in the resulting distance matrix.

For the different types of features, different formulas for calculating the distance are used. They are explained in the following paragraphs.

- *Relative distance:* For values like pixel height, the pure numerical difference is not a good comparison criterion for our purpose. In this case we rather calculate how much smaller the smaller value is than the larger value. In order to overcome several algorithmic issues, especially regarding possible divisions by zero and handling of negative values, we arrived at the following formula. f_1 and f_2 are two different values of a given feature:

  $$\delta_{rel} = \frac{1}{1 + e^{-max(f_1,f_2)}} - \frac{1}{1 + e^{-min(f_1,f_2)}}$$

  The maximum value for this distance is 1.0, which is also applied when one or both of the input values are `null`.

- *Absolute distance:* Especially for features which already have a percent value, i.e. a value between 0 and 1, we calculate the distance between the two values as $|f_1 - f_2|$. The maximum value for this distance depends on the feature used, but normally it is also 1.0, assuming a feature value in the range of 0 to 1.

- *Boolean distance:* For features that take on boolean values, the distance between two feature values is simply calculated by assign 1 if the values are identical and 0 if they are different. This is effectively a logical $\wedge$ operation with *true* being interpreted as 1 and *false* as 0.

- *Equality distance:* For features that can take on a value from a set of predefined enumerated values, we calculate the distance by assigning0 ('equal') if both features have the same value and 1 ('not equal') if they have different values.

- *String edit distance:* For comparing text, we use the string edit distance with transpositions, also known as the Damerau–Levenshtein distance [**?**]. For handling `null` values in the input, we assume that a missing value equals an empty string.

- *Grid overlap distance:* The grid distance feature specifies roughly which areas of the web page a given object touches. The web page is divided into a 3x3 grid, resulting in 9 areas that the object can possibly touch. The grid overlap distance is then calculated as two times the number of grid areas touched in both features' grids divided by the total number of grid areas touched. This number is subtracted from 1 in order to have 0 as similarity and 1 as the maximum dissimilarity.

- *Color distance:* All colors (e.g., foreground color) are represented in HSV color space, which roughly conforms to human perception. Accordingly, we calculate the color distance in HSV color space, with a minimum value of 0 (equal) and a maximal distance of 2.

6 ATW Dataset

To evaluate various approaches for the object identification problem, we have collected a number of real life web pages from the area of transportation search. In order to test our approaches under realistic conditions, we have defined four scenarios, namely searches for bus, flight and train connections, as well as a last one, which combines them all referring to it as the combined scenario. This last scenario is especially interesting as it is actually a meta-search scenario with searching over multiple different connection types. In all four scenarios, we focused on finding the relevant elements in the main search forms.

These web forms may have or may not have all of the relevant input fields we defined. The following list gives an outline of the different types of input fields for the mentioned scenarios. The elements of this list can be considered as sub tasks. In each scenario each input field (sub task) has to be found on the different websites.

Annotated objects in the dataset: 1) *departure location* annotated with tag `depLoc`, 2) *arrival location* (`arrLoc`), 3) *departure date* (`depDate`), 4) *one-way* (`oneWay`) which corresponds to the control that sets a one-way travel, 5) *adult passengers* (`adlPsgrs`) which is the number of adult travelers, 6) *submit* (`submit`), a button to send a request form, 7) *other* (`other`), a tag which correspond to all other elements on a web page.

In order to test our approaches over web pages in different languages, we decided to focus on those of them which were natively available in English, German or Russian (see Table 1).

The ATW dataset[2] [4] (Annotated Transport Web Forms) which provides us with necessary data contains 1) web pages from the transportation domain, 2) annotations of web page elements with mentioned tags and computed features listed in Section 4, 3) computed feature distances described in Section 5.

[2] (`http://www.dbai.tuwien.ac.at/proj/tamcrow/atw/`)

Table 1: List of web pages in the ATW dataset

Flight Search	**Webpage Id**	**From**	**To**	**Departure Date**	**One-way Trip**	**Adult Passengers**	**Search Button**	**Language**
http://www.britishairways.com/travel/home/public/en_at	ba	+	+	+	+	+	+	en
http://www.aa.com/homePage.do?locale=en_US&pref=true	american_airlines	+	+	+	+	+	+	en
http://www.emirates.com/at/english/index.aspx	emirates	+	+	+	+	+	+	en
http://www.aircanada.com/en/home.html	air_canada	+	+	+	+	+	+	en
http://www.qantas.com.au/travel/airlines/home/au/en	quantas	+	+	+	+	+	+	en
https://book.austrian.com/app/fb.fly?pos=AT&l=de	austrian_airlines	+	+	+	+	+	+	de
http://www.checkfelix.com/flugsuche/de/fluege.html?sourcedomain=at	checkfelix	+	+	+	+	+	+	de
http://www.lufthansa.com/online/portal/lh/de/booking	lufthansa	+	+	+	+	+	+	de
http://www.airberlin.com/site/start.php?LANG=deu&all=1&MARKT=DE	airberlin	+	+	+	+	+	+	de
http://www.germanwings.com/de/index.shtml	germanwings	+	+	+	+	+	+	de
http://www.aeroflot.ru/cms/ru/booking	aeroflot	+	+	+	+	+	+	ru
http://www.rossiya-airlines.ru/ru/tickets/zabronirovati_buy/	rossiya	+	+	+	+	+	+	ru
http://www.tatarstan.aero/tickets/buy/	tatarstan	+	+	+	+	+	+	ru
http://www.trip.ru/	tripru	+	+	+	+	+	+	ru
www.aviasales.ru/	aviasales	+	+	+	+	+	+	ru
Bus Search								
http://www.matkahuolto.fi/en/	matkahuolto	+	+	+	+	-	+	en
http://www.postbus.ch/	postbus_ch	+	+	+	-	-	+	en
http://www.buseireann.ie/	buseireann	+	+	+	-	-	+	en
http://www.greyhound.com/	greyhound	+	+	+	-	+	+	en
http://www.gotobus.com/	gotobus	+	+	+	+	+	+	en
http://195.110.209.27/ticketshop/DesktopDefault.aspx	eurolines	+	+	+	-	-	+	de
http://www.postbus.at/de/	postbus_at	+	+	+	-	-	+	de
http://www.regiobus.ch/	regiobus	+	+	+	-	-	+	de
https://www.berlinlinienbus.de/index.php	berlin_linienbus	+	+	+	+	-	+	de
http://www.publicexpress.de/buy-online/	public_express	+	+	+	-	+	+	de
http://www.avtovokzal.ru/	avtovokzal	+	+	+	-	-	+	ru
http://transport.marshruty.ru/	marshruty	+	+	+	+	-	+	ru
http://ticket.turistua.com/ru/bus/	turistua	+	+	+	-	-	+	ru
http://www.autovokzal73.ru/	autovokzal73	+	+	+	-	-	+	ru
http://www.avperm.ru/	avperm	+	+	+	-	-	+	ru
Train Search								
http://www.trenitalia.com/cms/v/index.jsp?vgnextoid=ad1ce14114bc9110VgnVCM10000080a3e90aRCRD	trenitalia	+	+	+	-	+	+	en
http://www.eurostar.com/	eurostar	+	+	+	-	+	+	en
http://www.voyages-sncf.co.uk/?rfrr=accueil_header_unitedkingdomgbp	scnf_uk	+	+	+	+	+	+	en
http://www.tgv-europe.com/en/	tgv_europe	+	+	+	-	-	+	en
http://www.irishrail.ie/	irishrail	+	+	+	+	-	+	en
http://www.oebb.at/	oebb	+	+	+	-	-	+	de
http://www.bahn.de/p/view/index.shtml	deutsche_bahn	+	+	+	-	+	+	de
http://www.sbb.ch/home.html	sbb	+	+	+	-	-	+	de
http://www.s-bahn-berlin.de/	s_bahn_berlin	+	+	+	-	-	+	de
http://www.saarbahn.de/de/start	saarbahn	+	+	+	-	-	+	de
http://www.tutu.ru/poezda/	tutu	+	+	+	-	-	+	ru
		+	+	+	+	-	+	ru
http://poezdato.net/	poezdato	+	+	+	-	-	+	ru
http://poezda.portal-poisk.ru/	portal-poisk	+	+	+	-	+	+	ru
http://pass.rzd.ru/	rzd	+	+	+	-	-	+	ru
Attribute tag:		**depLoc**	**arrLoc**	**depDate**	**oneWay**	**adlPsgrs**	**submit**	

7 Statistics

This section gives a brief overview of the ATW dataset [4] from the viewpoint of its statistical characteristics. We use boxplots and heatmaps. The former is used to demonstrable a distribution of the features' values in the ATW dataset (see Section 7.1). Whereas the latter is used to show correlations between the features (see Section 7.3). Section 7.2 presents a statistic ratios.

The following list maps names of features to their id used in the diagrams below.

1. AreaIO
2. AspectRatioIO
3. EmphasisIO
4. FontSizeIO
5. LinesQntIO
6. TokensQntIO
7. SimilarTypesQntROC
8. AlignmentQntROC
9. AlignmentHorQntROC
10. AlignmentIndexHorROC
11. AlignmentVertQntROC
12. AlignmentIndexVertROC
13. AlignmentFactorROC
14. AlignmentVertHorRatioROC
15. OrthogonalVisibleObjQntROC
16. AlignedOrthogonalVisibleObjQntROC
17. FullyAlignedOrthogonalVisibleObjQntROC
18. PixelsToCharacterIC
19. AvgWeightedFGColorROC
20. AvgWeightedBGColorROC
21. RelativeWidthROW
22. RelativeHeightROW
23. RelativeXPositionROW
24. RelativeYPositionROW
25. GridLocationX3ROTW
26. AlignmentQntROD
27. AlignmentHorQntROD
28. AlignmentIndexHorROD
29. AlignmentVertQntROD
30. AlignmentIndexVertROD
31. AlignmentVertHorRatioROD

32. TObjectsQntIC
33. TextSpatialDensityIC
34. LinkSpatialDensityIC
35. LinkCharacterDensityIC

7.1 Distribution Characteristics

Figures 5 and 6 present distributions of the different features of objects. For illustration purposes, data is standardized through the transformation of the original values into their z-values (2).

$$z_{i,j} = \frac{f_{i,j} - \hat{\mu}_{.j}}{\hat{\sigma}_{.j}}, \tag{2}$$

where $f_{i,j}$ is the i-th value of the j-th feature; $\hat{\mu}_{.j}$ is expected value and $\hat{\sigma}_{.j}$ is a standard deviation for the feature j in the dataset.

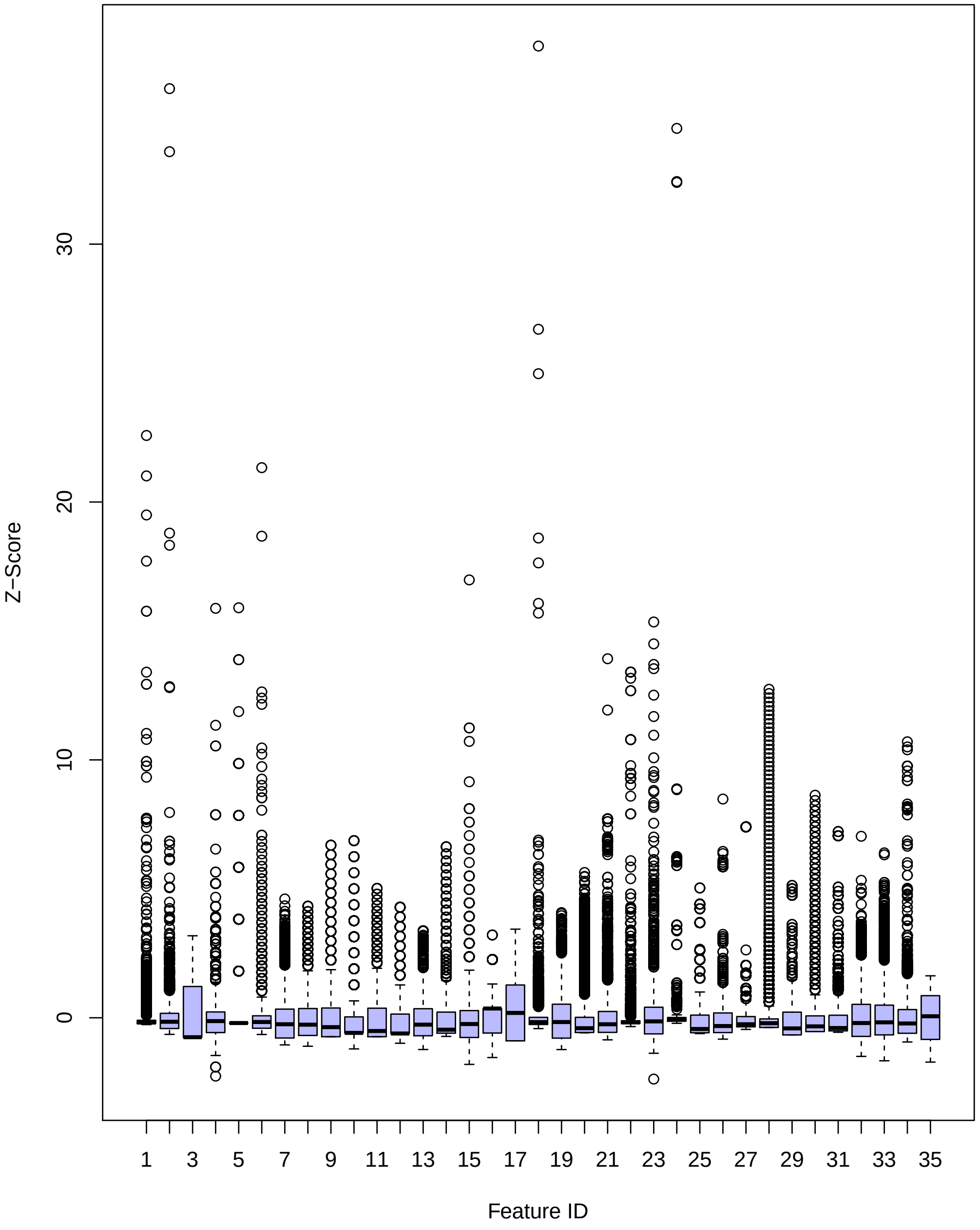


Figure 5: Distribution of the features as boxplot

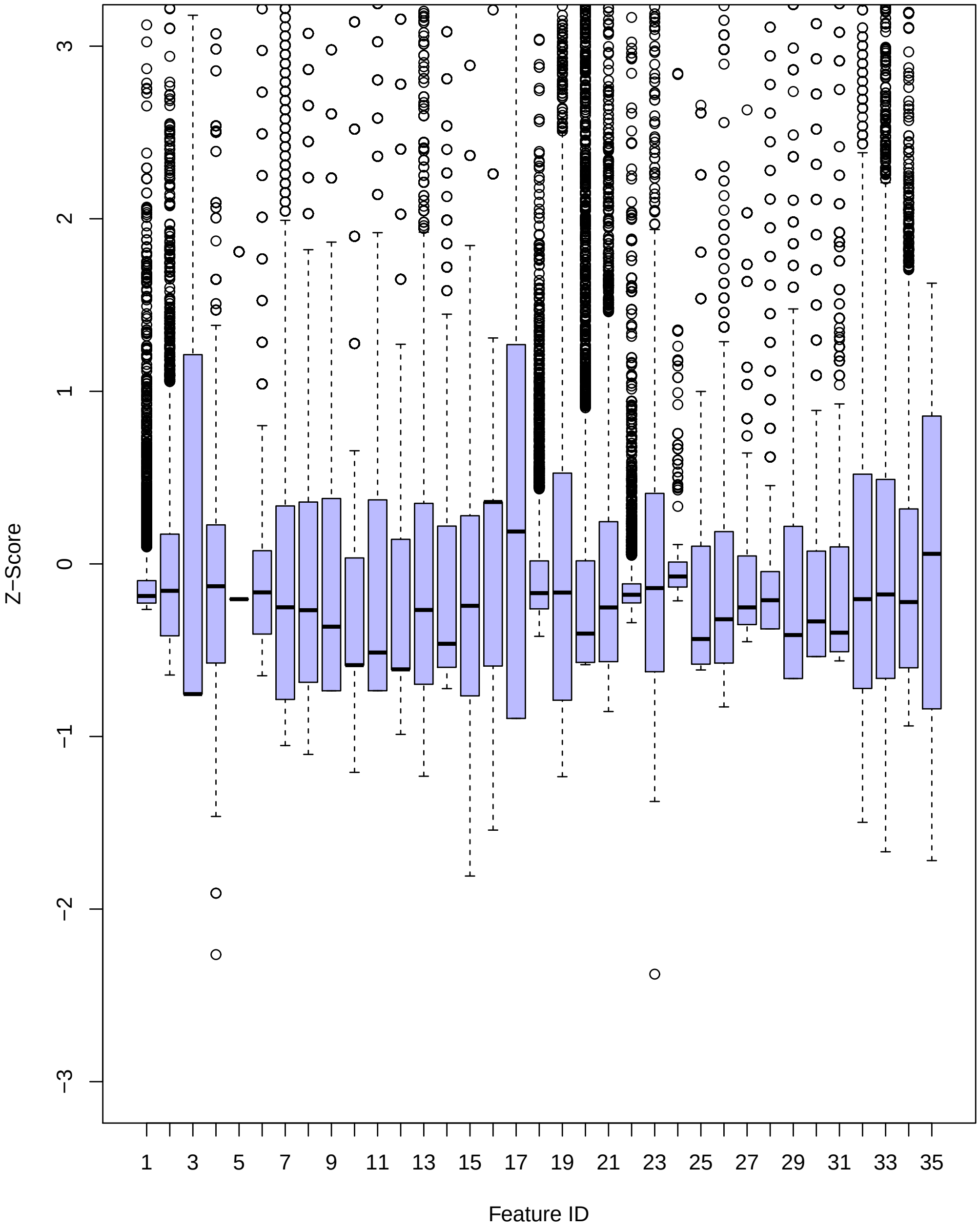


Figure 6: Distribution of the features as boxplot with zoom

7.2 Statistic Ratios

ID	Mean	Median	SD	Skewness	Kurtosis	Min	25% Q	75% Q	Max
1	3,695.74	1,092.00	14,002.44	11.96	187.81	1.00	518.00	2,340.00	319,872.00
2	5.28	4.00	8.18	19.41	594.53	0.01	1.87	6.69	300.00
3	1.13	1.00	0.17	0.75	-0.82	1.00	1.00	1.33	1.67
4	12.29	12.00	2.25	4.02	32.51	7.20	11.00	12.80	48.00
5	1.10	1.00	0.50	7.10	65.30	1.00	1.00	1.00	9.00
6	2.68	2.00	4.14	8.15	108.23	0.00	1.00	3.00	91.00
7	19.71	15.00	18.73	1.40	1.67	0.00	5.00	26.00	106.00
8	5.28	4.00	4.79	1.39	1.84	0.00	2.00	7.00	26.00
9	1.98	1.00	2.69	2.68	9.87	0.00	0.00	3.00	20.00
10	0.94	0.00	1.61	2.67	9.90	-1.00	0.00	1.00	12.00
11	3.32	1.00	4.52	1.84	3.57	0.00	0.00	5.00	26.00
12	1.62	0.00	2.65	1.98	3.68	-1.00	0.00	2.00	13.00
13	0.27	0.21	0.22	1.71	2.97	0.01	0.12	0.35	1.00
14	2.70	1.00	3.67	2.67	9.26	0.05	0.50	3.50	27.00
15	3.47	3.00	1.92	3.20	31.19	0.00	2.00	4.00	36.00
16	1.62	2.00	1.05	0.19	-0.42	0.00	1.00	2.00	5.00
17	0.83	1.00	0.92	0.84	0.03	0.00	0.00	2.00	4.00
18	1,520.81	964.78	3,286.06	21.00	609.13	141.86	665.08	1,578.01	125,343.00
19	0.36	0.31	0.29	0.99	0.81	0.00	0.13	0.52	1.57
20	0.11	0.03	0.19	2.60	6.75	0.00	0.00	0.11	1.15
21	0.10	0.07	0.12	4.21	28.42	0.00	0.03	0.13	1.72
22	0.03	0.01	0.07	8.99	93.10	0.00	0.01	0.02	1.00
23	0.46	0.40	0.50	5.58	56.45	-0.71	0.16	0.67	8.07
24	0.70	0.45	3.38	27.32	867.78	-0.03	0.24	0.73	117.19
25	54.83	16.00	89.23	2.30	5.16	0.00	3.00	64.00	504.00
26	9.79	6.00	11.81	3.62	17.07	0.00	3.00	12.00	110.00
27	4.54	2.00	10.06	6.07	41.16	0.00	1.00	5.00	79.00
28	2.27	1.00	6.02	7.71	74.00	0.00	0.00	2.00	79.00
29	5.27	2.00	7.94	2.68	8.64	0.00	0.00	7.00	46.00
30	2.64	1.00	4.91	3.66	17.95	0.00	0.00	3.00	45.00
31	3.40	1.00	6.04	3.84	19.20	0.01	0.33	4.00	47.00
32	28.95	25.00	19.33	1.15	1.88	0.00	15.00	39.00	165.00
33	0.12	0.11	0.07	1.50	4.11	0.00	0.07	0.16	0.58
34	0.09	0.07	0.09	4.23	31.17	0.00	0.03	0.12	1.09
35	0.51	0.53	0.30	-0.10	-1.11	0.00	0.26	0.77	1.00

7.3 Correlation

This subsection provides two heatmaps which represent the correlation matrix of the ATW dataset. In Figure 7, the Pearson's method is used. Correlation between variables x and y is computed according to the formula (3). Figure 8 illustrates a robust approach, Spearman correlation. The equation for the Spearman correlation coefficient is the same, whereas, for x and y, the rank of the ordered values is used instead of the actual values. Spearman's method is not influenced by outliers as much as Pearson's method and focus therefore more on the majority of the data, which is the advantage.

$$\rho_{x,y} = \frac{\sum_{i=1}^{n}((x_i - \hat{\mu}_x)(y_i - \hat{\mu}_y))}{\sigma_x * \sigma_y} \tag{3}$$

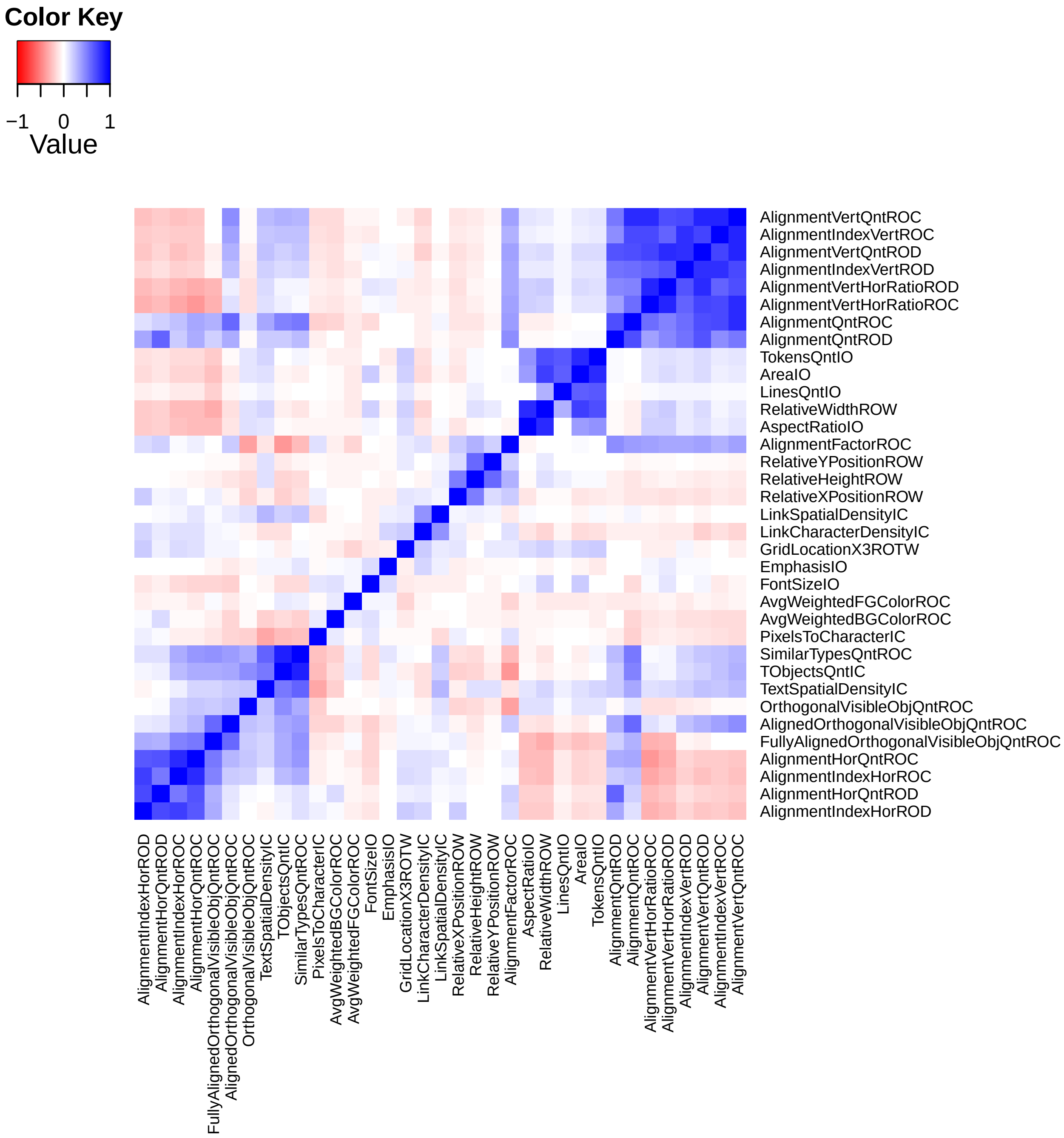


Figure 7: Heatmap illustrating Correlationmatrix using Pearson

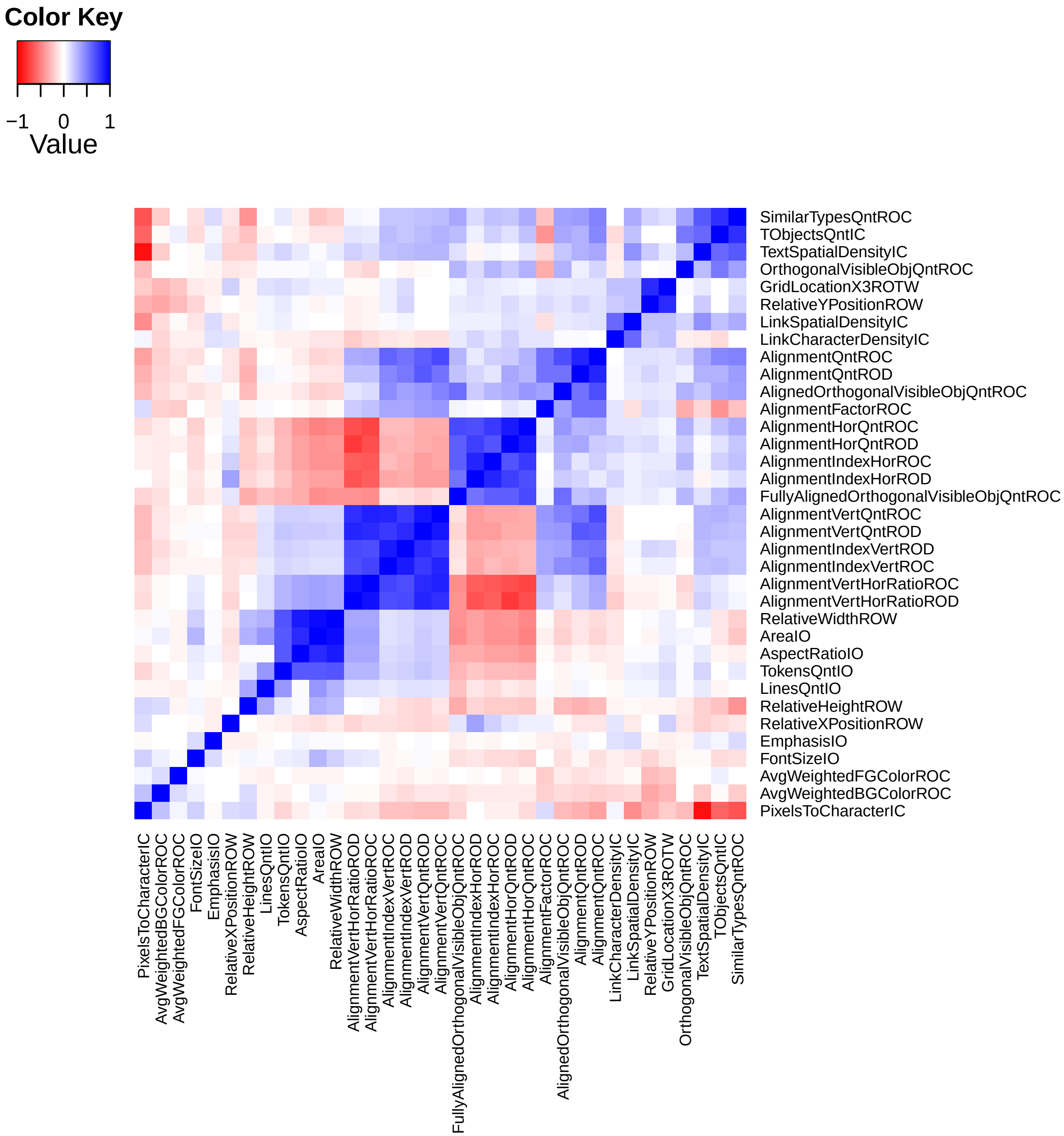


Figure 8: Heatmap illustrating Correlationmatrix using Spearman

References

[1] ABBA — Advanced barrier-free browser accessibility. FFG Fit-IT Project 819563, 2009–2010. http://www.dbai.tuwien.ac.at/proj/ABBA/.

[2] TAMCROW — Task mining and crowd sourcing. FFG Fit-IT Project 829614, 2011–2012. http://www.dbai.tuwien.ac.at/proj/tamcrow/.

[3] Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification (W3C Recommendation 07 June 2011), 2011.

[4] ATW Dataset. http://www.dbai.tuwien.ac.at/proj/tamcrow/atw/, 2012.

[5] CSS Regions Module Level 3. W3C Working Draft 23 August 2012, 2012.

[6] P. Balbiani, J.-F. Condotta, and L. F. n. Del Cerro. Tractability Results in the Block Algebra. *Journal of Logic and Computation*, 12(5):885–909, Oct. 2002.

[7] A. G. Cohn. Qualitative spatial representation and reasoning techniques. In G. Brewka, C. Habel, and B. Nebel, editors, *KI-97: Advances in Artificial Intelligence*, volume 1303, pages 1–30. Springer Berlin, Berlin, Germany, May 1997.

[8] R. R. Fayzrakhmanov. A blocks-based geometric model of web pages for automatic processing and information extraction. *Science and Business: Development Ways*, 15(9):56–64, 2012.

[9] R. R. Fayzrakhmanov. WPPS: A framework for web page processing. In X. S. Wang, I. Cruz, A. Delis, and G. Huang, editors, *In Proceedings of the 13th International Conference on Web Information Systems Engineering (WISE'2012), Demo Session, Paphos, Cyprus, 28–30 November, 2012*, pages 800–803. Springer, 2012.

[10] R. R. Fayzrakhmanov. WPPS: A novel and comprehensive framework for web page understanding and information extraction. In B. White and P. Isaías, editors, *Proceeding of the International Conference IADIS WWW/Internet, Madrid, Spain, 18–21 October, 2012*, pages 19–26, Madrid, 2012. IADIS Press.

[11] R. R. Fayzrakhmanov, M. C. Göbel, W. Holzinger, B. Krüpl, and R. Baumgartner. A unified ontology-based web page model for improving accessibility. In *Proceedings of the 19th international conference on World Wide Web (WWW'2010), Raleigh, USA, April 2630, 2010*, pages 1087–1088, New York, 2010. ACM.

[12] W. Gatterbauer, B. Krüpl, W. Holzinger, and M. Herzog. Web information extraction using eupeptic data in web tables. In *Proceedings of the 1st International Workshop on Representation and Analysis of Web Space (RAWS 2005)*, pages 41—-48, Prague, Czech Republic, 2005. VSB - Technical University of Ostrava.

[13] C. Herzog, I. Kordomatis, W. Holzinger, R. R. Fayzrakhmanov, and B. Krüpl-Sypien. Feature-based object identification for Web automation (to be published). In *Proc. of the 28th Annual ACM Symposium on Applied Computing. Web Technologies Track*, New York, 2013. ACM.

[14] J. Keith. *DOM Scripting: Web design with JavaScript and the Document Object Model*. Springer, New York, the USA, 1st edition, 2005.

[15] B. Krüpl, M. Herzog, and W. Gatterbauer. Using visual cues for extraction of tabular data from arbitrary HTML documents. In *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, pages 1000—-1001, Chiba, Japan, 2005. ACM.

[16] B. Krüpl-Sypien, R. R. Fayzrakhmanov, W. Holzinger, M. Panzenböck, and R. Baumgartner. A versatile model for web page representation, information extraction and content repackaging. In M. Hardy and F. W. Tompa, editors, *In Proceedings of the 11th ACM Symposium on Document Engineering (DocEng2011), Mountain View, USA, 1922 September, 2011*, pages 129–138, New York, 2011. ACM.

[17] I. Navarrete and G. Sciavicco. Spatial Reasoning with Rectangular Cardinal Direction. In *Proceedings of the ECAI 2006 Workshop on Spatial and Temporal Reasoning*, pages 1–9, 2006.

[18] D. A. Randell, Z. Cui, and A. G. Cohn. A Spatial Logic based on Regions and Connection. In B. Nebel, C. Rich, and W. Swartout, editors, *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*, pages 165–176, Los Altos, 1992. Morgan Kaufmann.

[19] R. Vulanović and R. Köhler. Syntactic units and structures. In *Quantitative Linguistics*, pages 274–291. de Gruyter, Berlin, 2005.