WEB PAGE SEGMENTATION AND CLASSIFICATION

Jan Zelený
Doctoral Degree Programme (1), FIT BUT
E-mail: xzelen11@stud.fit.vutbr.cz

Supervised by: Radek Burget
E-mail: burgetr@fit.vutbr.cz

Abstract: The World Wide Web contains more and more information each day and it is highly demanded, that search engine shows the most relevant result available. But it has to overcome a great amount of irrelevant information. This paper summarizes effort focused on filtering the content of web pages by segmenting the page to smaller pieces and filtering the irrelevant ones. This research area in a very important part of wider effort in content extraction, not just for its benefits to search engines, but also for ability to transform the content e.g. for special devices.

Keywords: web page segmentation, classification, VIPS, content extraction

1 INTRODUCTION

For last several years there is a growing trend in sharing a vast amount of information on the web. However with all this information new problem arises. Parsing and indexing it becomes a serious challenge. One of the main issues on the web is that the information contained on web sites is often mixed with irrelevant content, which might mislead automatic crawlers on the semantics of the page. A family of methods focused on how to deal with this issue has been developed and this paper brings summary of different approaches these methods are based on. The goal of these algorithms is **to find segments of the web page, which contain the relevant information**.

They receive data and meta data of a single web page as input and they output a content structure, usually in a form of semantic tree. The advantage of this family of methods is that they are designed to work on a single page (unlike Template detection methods). That also implies the main disadvantage – scaling. Especially for visual methods processing a large number of web pages can become a serious problem.

2 VISUAL PAGE SEGMENTATION

This family of methods is based on an approach with simple concept, but rather complex computing demands. The concept is to segment the web page as a user would segment it if he was looking at it. Some methods in this family even rely on user-assisted learning. Those methods however are not the subject of this paper, we will focus only on fully automatic ones. These have to simulate user view of the web page, which means a page has to be rendered either to an actual picture or at least to some kind of relevant internal representation of the visual information contained on the web page.

2.1 VISION-BASED PAGE SEGMENTATION ALGORITHM

Vision-based Page Segmentation algorithm [1] (or VIPS) is a basic algorithm in a family of visual based methods. The algorithm uses one essential term: *Degree of Coherence* or simply *DoC*. It is a measure of visual coherence defined for each block. It can be represented by any number (integer or

real), but it must grow with visual consistency of the block. Also a parent can never have greater DoC than its children in block hierarchy tree.

The algorithm segments page in three steps: (1) Extracting visual blocks, (2) Detecting separators between extracted blocks and (3) Detecting content structure based on results of previous two steps.

Extracting visual blocks consists of a top-down tree walking through the DOM tree[1]. The walk is iterative – in each iteration a new node representing visual block is detected in the DOM tree. After this detection a decision is made (based on certain properties like color, size, ...) whether the block shall be recursively segmented further or not. For each detected block which isn't segmented further a DoC is set according to its visual coherence.

The second step is separator detection. *Separator* is defined as horizontal or vertical line or rather rectangular area which doesn't intersect any of previously detected blocks. The algorithm is initialized by a single separator covering the whole page. Separators are always detected for a particular level of visual block tree. Then for each block we perform a detection of its relation to each existing separator:

- if the block is fully covered by area of a separator, divide this separator into two
- if the block partially intersects a separator, shrink that separator, so the intersection is eliminated
- if the block fully intersects with a separator (i.e. covers entire height of a horizontal separator or entire width of vertical separator), remove the separator entirely

The algorithm is finished by removing separators at the edges of document. After we have all separators, we assign the weights based on visual difference of adjacent blocks. Note that this algorithm produces either only vertical or only horizontal separators. However the page has to be segmented in both directions. Here it is important to realize that the whole algorithm is done recursively[2]. That means that the page can be separated only in one direction, but any of its child nodes can be separated on the other direction.

The final step of VIPS is content structure construction. In this step we iterate through a list of previously found separators and merge visual blocks adjacent to them. It's important to merge blocks adjacent to separators with the smallest weight first. Before merging we have to check whether blocks meet granularity requirement. If they do, there is no need for merging them. The granularity requirement is of course represented by PDoC and the general rule for meeting the granularity requirement is that $DoC > PDoC$. If the block doesn't meet the requirement, we return to step one with root node being that visual block.

This algorithm shows overall better results than its predecessors, but it has some shortcomings as well. Two were described in [2]:

- In some cases direct division of a visual block is impossible and utilization of virtual blocks is required. This can have negative impact on further processing, because blocks are not really present in the document.
- Resulting tree represents page segmentation but some information such as mutual position of blocks is missing. That information might be useful for improving algorithm's results.

[1] Of course additional visual information such as background color or font properties is required.

[2] in previous paragraph we stated that each of the detected blocks can be recursively segmented further – in that case the algorithm is invoked from the beginning, but the root node in the new run is replaced by node corresponding to the segmented block

2.2 OTHER VISUAL ALGORITHMS

Burget and Rudolfová in [2, 3] introduced a bottom-up method based on some concepts introduced by VIPS. The algorithm he described has a goal to deal with VIPS shortcomings described in section 2.1. It creates a tree of visual areas in four steps: (1) creating temporary tree of boxes, (2) finding boxes which represent standalone visual areas, (3) detecting continuous areas and (4) finding significant areas.

The *box* is a basic compound of the web page – it is defined as a rectangular area with defined position and width/height and containing either another set of boxes or a content of the web page. The first step creates a tree of these boxes. First we need to identify them, then we can build the tree. The tree basically defines nesting of the boxes in another boxes. There are three possibilities of a relation between two boxes:

- *Boxes don't intersect:* they are not related.
- *One box is completely nested in another:* the larger one is considered a parent
- *Boxes have a partial intersection:* This is detected by mutual position of boxes' corners. The parent box is then extended to contain the whole child box.

The whole page is representing a root node of the box tree. In the tree only two rules apply: parent box contains all child boxes and all child nodes on the same level of the tree mustn't have intersection with each other.

In the second step we create a tree of visual areas by merging visually same boxes from step one. The final tree of visual areas corresponds with the tree of visual boxes, but its nodes have the above described feature – all are visually distinct from each other. The third step brings more merging. This time visually similar nodes (e.g. adjacent paragraphs of text) are merged into one continuous block. In this step an information about mutual block position is required. In the fourth step we look for significant areas by finding optical separators like borders, lines or just big spaces between blocks.

This method has couple advantages over VIPS. One of them is particularly useful for our purposes – it is strictly focused on visual information rather than the DOM tree, which can bring better results e.g. for absolute positioning on the web page.

3 HTML BASED SEGMENTATION METHODS

These methods are based on analyzing web page without the need for rendering it. That means selected approach is either based on inspecting the HTML code directly or (more often) traversing the DOM tree and evaluating information gathered from it. They are usually much faster than visual-based methods. Quality and speed of these methods is given by used heuristics. Below are described some examples of algorithms which illustrate what heuristics can be used.

3.1 TEXT BASED

DOM or Document Object Model is a tree structure representing a nesting of elements within the page and it is often used for traversing the web page. Method introduced in [4] uses very crude heuristics. The segmentation part of the method is unusable nowadays, because it considers table data cells as content blocks (modern web pages don't use table layout). For this reason we shall omit it and focus on classification part of the method. The method classifies segments into two categories: informational and redundant. The entropy of content features (represented by meaningful words) is used as heuristics. A matrix of all features and their occurrence on all pages (in all content blocks) is

created and the probability of particular feature being in particular content block is calculated. The final step is to calculate entropy of the block from all of its features. If the entropy reaches threshold level, the block is considered redundant. Otherwise it is considered informative.

There are some other methods utilizing features of the text itself and building heuristics on them. An example of such can be a method proposed by Laber et al. [5]. The method is based on calculating link density and statistical F_1 score of all blocks. The basic assumption is that for every page there is one block with high link density and high F_1 – this block is then marked as the main content.

The algorithm itself just performs DFS traversal of the DOM tree and it is trying to find a leaf node which contains defined amount of characters and has an ancestor which passes conditions defined in [5] (text amount, link density, non-presence of siblings with similar properties).

3.2 DOM BASED

The last group of segmentation methods is based on general traversal of the DOM tree and identifying the content with usage of various heuristics. Some of related works might not even be directly solving the segmentation by traversing the DOM tree.

WISH algorithm: In [6] Hong et al. introduce their technique for traversing the DOM tree and selecting relevant content. They target so called *data records* which are pieces of a web page which are repeating themselves, but with a different content. An example of such record can be a category or search results listing. Their algorithm is divided in several steps. In the first step, they extract content candidate nodes using BFS-based algorithm. Data records are defined as tags on the same level of DOM tree, containing repetitive children sequences and having similar parent. The parent is denoted as *data region*. In case no nodes on a particular level of BFS satisfy the definition, the next tree level is inspected. Output of the first stage is a list of all data regions identified on the page. Other stages only filter results the algorithm gained in the first stage. Following observations are used for filtering heuristics:

1. Relative to the whole page, data records (and subsequently the whole data region) have large size
2. Data Records are usually repeated more than three times on a page
3. A regular expression can be devised for description of data record. Since all data records share the same template, it will apply on all of them
4. Data Records usually consist of a small amount of HTML tags

After the list of data regions is filtered, every data region has to be assigned its relevancy score. The soring function described in [6] determines the size of area taken by data records by counting characters and images each data record contain. Elements representing free space are taken into account as well. Data region with the best score is considered to be the main content of the page. This algorithm is the best example of how can different heuristics be used for page segmentation and classification.

4 FUTURE RESEARCH

There are several areas which might be convenient for future research because there wasn't too much research done in them. The first such area is combination of visual algorithm with DOM and text based heuristics. Although all visual methods still work with a DOM tree (at least to some extent), they rely heavily on computed visual information. These algorithms can be accelerated by adding

various non-visual heuristics. Their target would be to avoid computing of visual features, which is the most expensive part of visual segmentation algorithms.

Another interesting area of research is improving results of visual segmentation by transforming DOM tree into a *Semantic tree*. The HTML code and corresponding DOM tree is often very different from semantic structure of the page as user perceives it. The purpose of the transformation algorithm would be to design a new tree which would correspond with visual perception of the page. Of course the page would need to be rendered, then a semantic tree would need to be constructed and finally a minimal mapping between the DOM tree and the semantic tree would have to be found. Classification can be then performed on the semantic tree and by means of inverse mapping (from the semantic tree to the DOM tree) it can be applied to the page itself. The idea is that classification of nodes of semantic tree has a potential to be much more precise than the classification performed on the DOM tree.

5 CONCLUSION

This paper offered an overview of distinct methods which can be used for finding a relevant content on the web page. Each method has its advantages and disadvantages and their usage should be considered according to a particular task which needs to be solved. Many of presented algorithms were originally targeted at a detection of a content on news servers. But if we consider how modern web pages are designed, the same approach can be applied to blogs, CMS-based sites and also most of company web sites.

ACKNOWLEDGEMENT

This work was partially supported by the BUT FIT research plan MSM0021630528.

REFERENCES

[1] Cai, D., Yu, S., Wen, J.-R., Ma, W.-Y.: VIPS: a Vision-based Page Segmentation Algorithm.. Microsoft technical report. MSR-TR-2003-79. 2003

[2] Burget, R.: Vizualni segmentace elektronickych dokumentu, In: Znalosti 2007, Ostrava, CZ, VSB, 2007, s. 155-166, ISBN 978-80248-1279-3

[3] Burget, R., Rudolfová, I.: Web Page Element Classification Based on Visual Features, In: 1st Asian Conference on Intelligent Information and Database Systems ACIIDS 2009, Dong Hoi, VN, IEEE CS, 2009, s. 67-72, ISBN 978-0-7695-3580-7

[4] Lin, S. H., Ho, J. M.: Discovering Informative Content Blocks from Web Documents. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2002

[5] Laber, E., Souza, C., Jabour, I., Amorim, E., Cardoso, E.: A Fast and Simple Method for Extracting Relevant Content from News Webpages. In *Proceeding of the 18th ACM International Conference on Information and Knowledge Management*. 2009

[6] Hong, J. L., Siew, E. G., Egerton, S.: Information extraction for search engines using fast heuristic techniques. In *Journal of Data and Knowledge Engineering*. February 2010