Journal of Universal Computer Science, vol. 14, no. 11 (2008), 1893-1910
submitted: 30/9/07, accepted: 25/1/08, appeared: 1/6/08 © *J.UCS*

Recognising Informative Web Page Blocks Using Visual Segmentation for Efficient Information Extraction

Jinbeom Kang and Joongmin Choi
(Hanyang University, Ansan, Korea
{midgetfx,jmchoi}@hanyang.ac.kr)

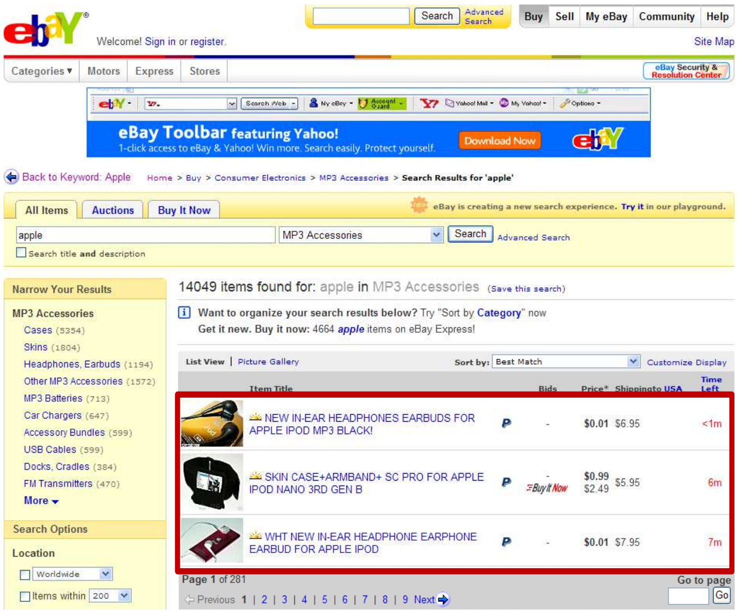
Abstract: As web sites are getting more complicated, the construction of web information extraction systems becomes more troublesome and time-consuming. A common theme is the difficulty in locating the segments of a page in which the target information is contained, which we call the informative blocks. This article reports on the Recognising Informative Page Blocks algorithm (RIPB), which is able to identify the informative block in a web page so that information extraction algorithms can work on it more efficiently. RIPB relies on an existing algorithm for vision-based page block segmentation to analyse and partition a web page into a set of visual blocks, and then groups related blocks with similar content structures into block clusters by using a tree edit distance method. RIPB recognises the informative block cluster by using tree alignment and tree matching. A series of experiments were performed, and the conclusions were that RIPB was more than 95% accurate in recognising informative block clusters, and improved the efficiency of information extraction by 17%.

Key Words: Information extraction, informative block, visual block segmentation
Category: H.3.7, H.5.4

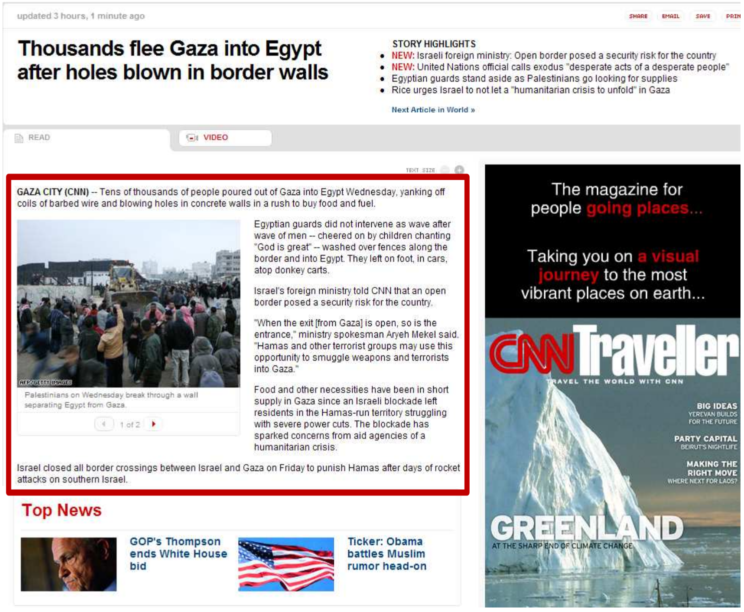
1 Introduction

One of the key issues in web information extraction is to locate and recognise target information correctly [Chang et al., 2006, Turmo et al., 2006]. Most information extraction systems rely on machine learning techniques to build extraction rules. Supervised learning is advantageous in terms of correctness, but typically a large volume of data are needed to train the system. Although unsupervised learning is very attractive because it does not require any training data, fewer results are available [Shi et al., 2005, Crescenzi and Mecca, 2004, Wong and Lam, 2007]. Wrapper induction relies on semi-automatic supervised learning [Kushmerick, 2000], i.e., it learns extraction rules from labelled web pages and data records and uses them to extract the relevant target information from new web pages with similar patterns as the training data. Current approaches to wrapper induction need to examine the whole pages, which might be problematic if the pages being examined have complex layouts or the induction algorithm is costly.

Recently, web information extraction has become more challenging due to the complexity and the diversity of web structures and representations. This is an expectable phenomenon since the Internet has been so popular and there are now many types of web contents, including text, videos, images, speeches, or flashes. The HTML structure of a web document has also become more complicated, making it harder to recognise the target content by using the DOM tree only. Another trend is that web designers are



(a) A product page in a shopping mall site.



(b) An article in a news site.

Figure 1: Examples of informative blocks.

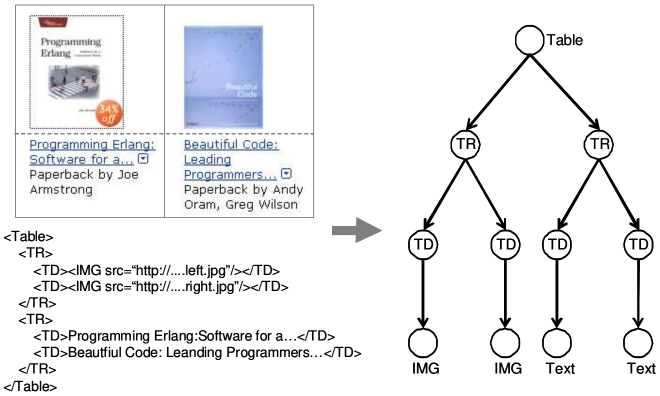
adding more advanced graphical features to the web content to make it more appealing. Therefore, we think that it would be helpful for wrapper induction and information extraction if we could provide some clues about where the content to be extracted resides.

Based on this motivation, this article proposes a method to identify informative web page blocks for efficient information extraction. An informative block is defined as a logical part of a web page that contains its core content. Examples of informative blocks are shown in Figure 1. In a shopping mall page, the informative block might contain a list of product descriptions, and in a news page, it might be an article. Locating the informative blocks in a web page is not an easy task, chiefly for web pages that are built by using many graphical and visual features for human readability. Our key idea to achieve this goal is to explore the visual characteristics of the web page, not just relying on the HTML hierarchical structure. This idea led us to developing the Recognising Informative Page Blocks algorithm (RIPB), which builds on the Vision-based Page Segmentation method (VIPS) [Cai et al., 2003]. VIPS partitions a web page into a set of blocks based on visual information and analyses the relationships amongst block segments. After visual block segmentation, RIPB groups related blocks with similar patterns into a block cluster and identifies the informative block cluster according to user interests. The visual block segmentation scheme in VIPS is also used in the wrapper induction phase by recognising the visual block that contains the user-specified target item to discover information patterns and construct wrapper rules efficiently. A series of experiments have been carried out in real web sites in the domains of e-commerce and on-line news. The results of these experiments indicate that RIPB greatly contributes to improve the efficiency of information extraction by allowing the system to focus on the visually-segmented blocks to generate wrapper rules, and also focus on the informative blocks in the extraction phase.

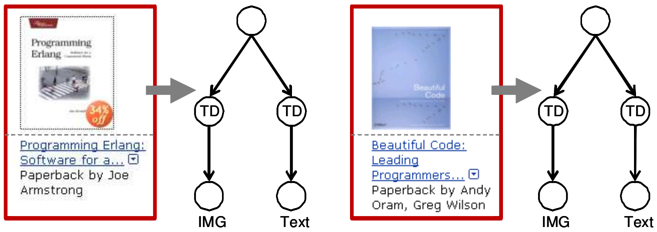
The rest of the article is organised as follows: Section 2 reports on the limitations of previous approaches to wrapper-based information extraction and streamlines our solution; Section 3 describes the overall architecture of our RIPB-based information extraction system and the details of the algorithm for recognising informative page block clusters; Section 4 reports on the results of evaluating the RIPB algorithm and the RIPB-based information extraction system; Section 5 concludes with a summary and reports on future research directions.

2 Related Work

There are many of proposals to build information extractors. Most of them rely on analysing HTML tag sequences only; recently, a few methods that use visual clues have been proposed. Our method relies on using visual segmentation to improve the efficiency of information extraction, which is a novel approach that advances the state of the art. To support this claim, we compare the RIPB algorithm with other related methods regarding two criteria: the ability to identify a target item uniquely, and the ability to extract it correctly.



(a) A traditional approach: hierarchy-based segmentation.



(b) Our approach: vision-based segmentation.

Figure 2: Comparison between hierarchy-based segmentation and vision-based segmentation.

The first criterion is whether a unique target item can be identified. Many current methods rely on hierarchy-based algorithms that consider any two elements as belonging to the same item only when their corresponding HTML tags are located under a common parent tag in the DOM tree [Wong and Lam, 2007, Buttler et al., 2001, Crescenzi and Mecca, 2004]. Incorrect extraction might occur for an item with several elements that are visually positioned closely but textually located under different parent tags in the tree hierarchy. For instance, assume that a search result containing information about two books is given as in Figure 2(a). Here, a book can be considered as an item, and every attribute of a book can be identified with an element, e.g., image or title. Visually, it is easy to recognise and identify book records, which consists of an image, a title, and information about the authors. However, at the HTML source level, they can be coded as a table in which the images are in the first row and the titles are in the second row. This can be represented by a DOM tree by using the table and tr tags, with the images (img) and the titles (text) located under different tr tags, as shown in the right part of Figure 2(a). A common parent containing both text and img is table, which means that a hierarchy-based approach might not be able to identify each book since it is almost impossible to extract a subtree containing only one item (in this case, a single book). In short, hierarchy-based extraction methods would find it difficult to identify the target item precisely. We tried to overcome this problem by a vision-based approach that visually segments a block for each book that results in more correct recognition as shown in Figure 2(b).

Recently, a few proposals for extracting web information by using visual information have been reported in the literature. Yang proposed a method to analyse the semantic structure of an HTML page in terms of the visual similarities amongst the content objects in the page [Yang and Zhang, 2001]. This method represents the hierarchical structure of an HTML document by a tree to maintain the visual consistency. The proposal attempts to find a visual block, but the blocks are not actually separated for the same reason as explained in the above example.

The second criterion is related to how target information is extracted correctly. A typical system extracts it by using extraction rules that are built from the entire HTML training pages [Zhai and Liu, 2006, Robinson, 2004, Arasu and Molina, 2003, Crescenzi and Mecca, 2004, Wang and Zhou, 2003]. It is very likely that the rules constructed in this way are too general or too specific. This is important since, if the rules are too general, then much of irrelevant information might be extracted and, if the rules are too specific, then the whole information would not be extracted. Our method tackles this problem by having the extraction phase consider only the informative blocks that are candidate to have the target information. In short, the informative blocks contribute to preventing the information extraction system from extracting irrelevant information.

In short, the RIPB algorithm has the advantage that it is able to recognise the target content area by using the concept of visually separable blocks. Certainly, this new technique might have an impact on the efficiency of the previous methods of wrapper-based information extraction listed above.

3 RIPB: Recognising Informative Page Blocks

The architecture of our RIPB-based information extraction system is shown in Figure 3. The system operates in two phases: the wrapper learning phase and the information extraction phase.

The first phase builds the extraction rules from the training web pages; the user selects the target information area, and the wrapper induction module performs vision-based segmentation of the block that contains the user-selected item. Then, a DOM tree is built for each user-selected block and the data about user-selected blocks are combined by using a tree alignment method. Finally, the tag pattern that corresponds to the target area is picked out and stored as an extraction rule.

The information extraction phase extracts the target information from the test pages by applying the rules generated by the wrapper learning phase to the informative block clusters that are recognised by RIPB. In this phase, the RIPB module serves as a front-end for the extraction process. RIPB operates in three stages, namely: first, it uses the

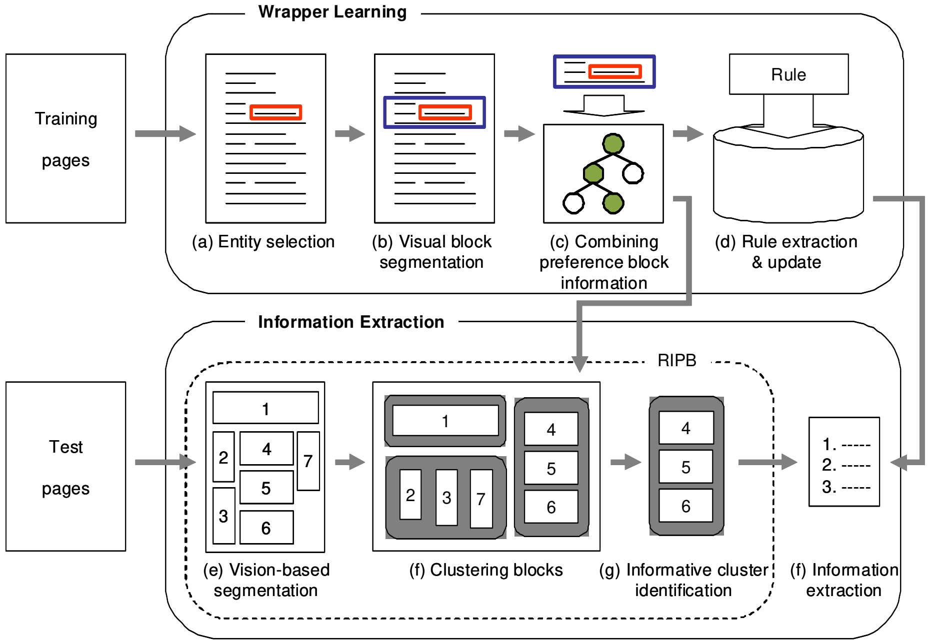


Figure 3: Architecture of the RIPB-based information extraction system.

VIPS algorithm to analyse and partition a web page into a set of visually-separated blocks; then, it groups related blocks with similar structures or patterns into a block cluster; finally, it recognises informative block clusters by reflecting the user-preferred block information represented by an aligned tree, which is built in the previous phase.

3.1 Visual Page Block Segmentation

VIPS partitions a web page into logical data blocks based on a visual segmentation algorithm that mimics human behaviour [Cai et al., 2003]. It fully relies on page layout features, including the DOM structure and visual clues. It first converts a web page into an HTML DOM tree; then, it extracts all the suitable blocks from the tree based on visual clues; later, horizontal or vertical lines that act as separators are identified. An example of visual block segmentation is shown in Figure 4.

An important parameter of VIPS is the so-called Degree of Coherence value (DoC), which measures how coherent every visual block is. Its values lie in the integer interval 1..10. There is a predefined Permitted Degree of Coherence value (PDoC) that allows to adjust the granularity of the algorithm for different applications. For a small PDoC value, we obtain a small number of large-sized visual blocks, and for a large PDoC

Figure 4: An example of visual block segmentation of a web page.

value, we obtain a large number of small-sized blocks.

3.2 Block Clustering

Although visually separated blocks provide a semantic partitioning of a page, a block might be too small to be considered as the source for information extraction, chiefly when the PDoC value of VIPS is set to a large value. For example, the page shown in Figure 4 contains a list of product descriptions, but the VIPS algorithm produces a visual block for each product description separately. Since the entire list of product descriptions should be the source of information extraction, they must be grouped together into a single unit. In RIPB, the blocks with similar content structures are clustered by using well-known edit distance methods. Since we are dealing with HTML pages and

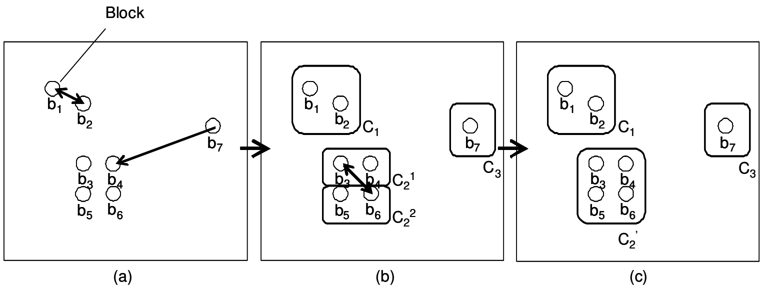


Figure 5: Block clustering.

their segmented visual blocks that are represented by tree structures, we apply a tree edit distance algorithm to measure the similarities amongst blocks [Bille, 2005].

Let T_1 and T_2 be two DOM trees with a left-to-right order amongst their siblings. Each node of these trees is assigned a label with its corresponding HTML tag name. The edit distance, $\delta(T_1, T_2)$, between T_1 and T_2 is defined as the minimum cost to transform T_1 into T_2 by using insertion, deletion, and replacement operations on nodes. Each edit operation is represented by $(n_1 \rightarrow n_2)$, where n_i is an actual node or an empty node denoted by ϵ. The operation is a node replacement if $n_1 \neq \epsilon$ and $n_2 \neq \epsilon$, a node deletion if $n_2 = \epsilon$, and a node insertion if $n_1 = \epsilon$. Given a metric cost function γ defined on pairs of labels, we define the cost of an edit operation by setting $\gamma(n_1 \rightarrow n_2) = \gamma(n_1, n_2)$. Based on these definitions, the tree edit distance $\delta(T_1, T_2)$ is calculated as follows:

$$
\begin{aligned}
\delta(\emptyset, \emptyset) &= 0 \\
\delta(F_1, \emptyset) &= \delta(F_1 - v, \emptyset) + \gamma(v \rightarrow \epsilon) \\
\delta(\emptyset, F_2) &= \delta(\emptyset, F_1 - w) + \gamma(\epsilon \rightarrow w) \\
\delta(F_1, F_2) &= \min \left\{ \begin{array}{l} \delta(F_1 - v, F_2) + \gamma(v \rightarrow \epsilon), \\ \delta(F_1, F_2 - w) + \gamma(\epsilon \rightarrow w), \\ \delta(F_1(v), F_2(w)) + \delta(F_1 - T_1(v), F_2 - T_2(w)) + \gamma(v \rightarrow w) \end{array} \right\} \\
\delta(T_1, T_2) &= \delta(F_1(v), F_2(w)) + \gamma(v, w)
\end{aligned}
\tag{1}
$$

where the root of T_1 is v and the root of T_2 is w. Every T_i denotes a tree, actually a root node, that is connected to an ordered sequence of disjoint trees; such a sequence is called a forest denoted by F_i; $F_i - v$ denotes the forest obtained by deleting v from F_i, and $F_i - T_i(v)$ denotes the forest obtained by deleting a tree rooted at v from F_i; $F_i(v)$ denotes the forest obtained by deleting the node v from F_i.

Our block clustering method is similar to a density-based clustering algorithm and consists of two steps: the first one is to build clusters by measuring the tree edit distances amongst blocks, and to eventually map each block onto a cluster that contains its nearest block; the second one is to merge the resulting clusters. Figure 5 shows an example of block clustering, assuming that the blocks are positioned in the space according to their relative distances. The distance $D(b_i, b_j)$ between two blocks b_i and b_j is measured by the tree edit distance between their corresponding DOM trees T_{b_i} and T_{b_j}, normalised by the size of the largest tree to minimise the impact of the tree size to the weight of the distance. Thus, D can be calculated as follows:

$$D(b_i, b_j) = \delta(T_{b_i}, T_{b_j}) / \max(|T_{b_i}|, |T_{b_j}|) \qquad (2)$$

Cluster building for this example proceeds as follows: the nearest block to b_1 is b_2, so b_1 and b_2 are grouped into a new cluster c_1. Since the nearest block to b_2 is also b_1 no new cluster is built. Similarly, b_3 and b_4 are clustered into ${c_2}^1$, and b_5 and b_6 are clustered into ${c_2}^2$. There is a problem with b_7: it can be merged into ${c_2}^1$ since its nearest block is b_4; however, if we assume that b_7 is located far away from any existing clusters, it would be better to create a new cluster c_3 for b_7. We solve this problem by setting a threshold value for the distance in a way that a block should not be merged into an existing cluster when the distance to the nearest block exceeds the threshold. In RIPB, the threshold is set to the median value of all the block distances, denoted by $Median$; from our experiments, we concluded that the mean value is not appropriate in cases in which there are many outliers or block distances are skewed.

The second step is to merge clusters, which is required in cases in which two blocks are not clustered by the cluster building process, although they are sufficiently near to each other. For instance, clusters ${c_2}^1$ and ${c_2}^2$ in Figure 5(b) should be merged into a larger cluster c_2, as shown in Figure 5(c). To determine if two clusters must be merged, we define the cluster distance between two clusters c_k and c_l as the maximum value of $D(b_i, b_j)$, for every two blocks $b_i \in c_k$ and $b_j \in c_l$. In the example, the cluster distance between ${c_2}^1$ and ${c_2}^2$ is equal to $D(b_3, b_6)$, since (b_3, b_6) is the farthest block pair for the two clusters. Two clusters are merged if their distance is smaller than $Median$.

Algorithm 1 describes these ideas formally. Note that it implements a clustering function $BlockClustering : B \rightarrow C$ that constructs clusters from blocks and maps each block onto a cluster. Here, $B = \{b_1, b_2, \ldots, b_m\}$ is the set of blocks obtained by VIPS, and $C = \{c_1, c_2, \ldots, c_n\}$ is the set of clusters. Note that C is a partition of B since $c_i \subseteq B$, $c_i \cap c_j (i \neq j) = \emptyset$, and $\cup_{i=1}^{n} c_i = B$. For the example shown in Figure 5, $BlockClustering$ returns $C = \{\{b_1, b_2\}, \{b_3, b_4, b_5, b_6\}, \{b_7\}\}$ for $B = \{b_1, b_2, b_3, b_4, b_5, b_6, b_7\}$. Figure 6 shows an example of block clustering applied to the result in Figure 4. Note that the list of three product descriptions that were previously segmented into separate blocks are now grouped into Cluster 8.

3.3 Recognising Informative Block Clusters

After clustering blocks with similar content structures, the RIPB algorithm identifies the informative cluster in a page that contains the blocks with meaningful information

Algorithm 1 Block Clustering

function BLOCKCLUSTERING(B) ▷ $B = \{b_1, b_2, b_3, \ldots, b_{|B|}\}$
    $C \leftarrow \{\{b_1\}, \{b_2\}, \{b_3\}, \ldots, \{b_{|B|}\}\}$
    for all $b_i \in B$ **do** ▷ Cluster Building
        if $\min_{b_j \in B}(D(b_i, b_j)) < Median$ **then**
            $C \leftarrow C - \{c_k, c_l\}$ **where** $b_i \in c_k, b_j \in c_l$
            $C \leftarrow C \cup \{c_k \cup c_l\}$
        end if
    end for
    repeat ▷ Cluster Merging
        for all $c_k, c_l \in C, k \neq l$ **do**
            if $\max_{b_i \in c_k, b_j \in c_l}(D(b_i, b_j)) < Median$ **then**
                $C \leftarrow C - \{c_k, c_l\}$
                $C \leftarrow C \cup \{c_k \cup c_l\}$
            end if
        end for
    until C is not changed
    return C
end function

that will be the source for information extraction. Informative block clusters include the cluster with the product description blocks in a product list page in a shopping mall site or the article block cluster in a news page.

In general, the product list pages and product detail pages of a shopping mall site have a cluster with many blocks since there exist repeating structural patterns in these pages. However, an article page of a news site rarely contains a repeating content pattern. Thus, it would not be a good idea to just count the number of blocks in a cluster to recognise the informative block cluster.

Our first attempt to measure the amount of information in a block cluster was to consider two features: tokens and images. Tokens are certainly the most important factors, but images can often provide some information about the contents of a cluster. We have implemented a function called $Bscore$ that evaluates the information content of a block by using a weighted linear combination of the number of tokens and the size of images in the block, namely:

$$Bscore(b) = (1 - \beta) \cdot |tok_b| + \beta \cdot size(img_b) \tag{3}$$

where, tok_b denotes the set of tokens in b, img_b denotes the set of images in b, and $size(img_b)$ denotes the sum of the sizes of the images in img_b. Note that the information content of a cluster is simply the sum of the information contents of all the blocks in the cluster. We have tested the performance of this method for real web sites and ob-

Figure 6: An example of block clustering.

tained good results with more than 85% average accuracy for recognising informative block clusters [Kang and Choi, 2007]. However, we have found a number of problems: first, the performance of this method largely relies on the weight factor β, i.e., although it shows good performance for some application domains, it may not be generally applicable since β should be set to different values for different sites to achieve the best performance; second, this method may work well for filtering out noise blocks such as

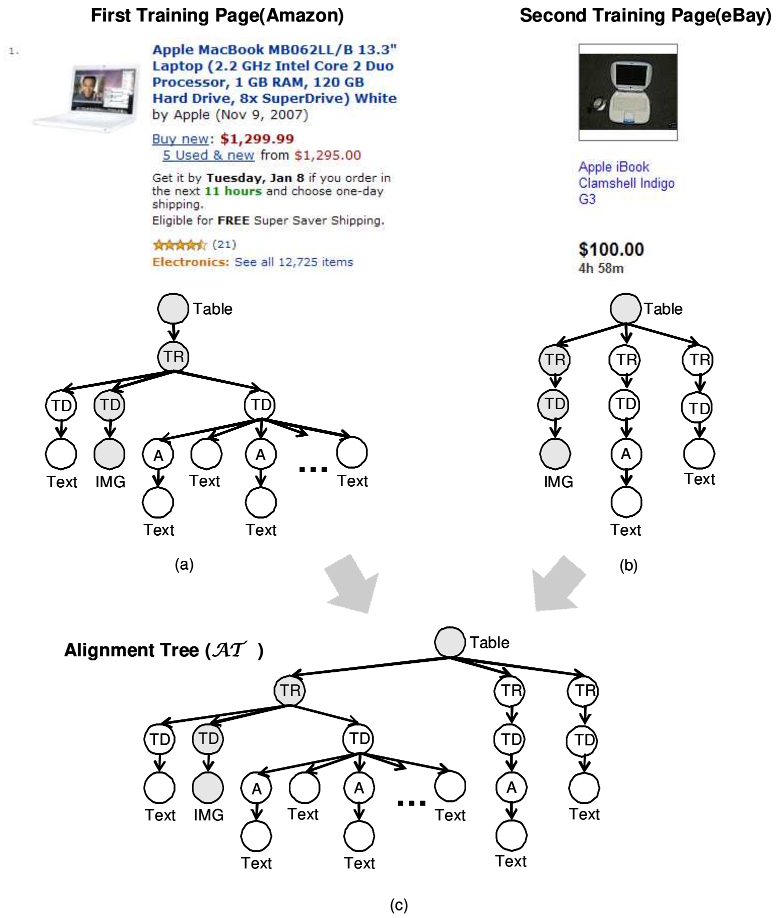


Figure 7: Combining user-preferred block information by using tree alignment.

advertisements or menus, since the characteristics of these blocks can be analysed by using tokens and images only, but it would not be a good criterion in other application domains; third, this method is a kind of unsupervised approach that does not take any context information into account, so it could not handle a situation in which the same block can be regarded as informative or noisy depending on user preferences.

The last observation led to another supervised approach that relies on the user. For instance, for a product detail page, some users might be interested in the detailed description of the product, whereas others might focus on customer reviews. In this situation, different informative clusters can result from the same page depending on user preferences. In order to capture user interests and apply them to the recognition of informative clusters, we build and combine trees for user-selected blocks by using the

Algorithm 2 Informative Cluster Recognition

function INFORMCLUSTERRECOG(C) $\triangleright C = \{c_1, c_2, c_3, \ldots, c_{|C|}\}$
  for all $c_i \in C$ **do**
    $sum\_Bscores \leftarrow 0$
    for all $b \in c_i$ **do**
      $sum\_Bscores \leftarrow sum\_Bscores + W(\mathcal{AT}, T_b)$
    end for
    $Cscore[i] \leftarrow sum\_Bscores/|c_i|$
  end for
  $c \leftarrow c_j$ **where** $j = \arg\max_{i=1,2,\ldots,|C|}(Cscore[i])$
  return c
end function

tree alignment method, as illustrated in Figure 7. In this example, there are two training pages, and we assume that the user wants to extract the titles of the books. In the wrapper induction phase, as the user selects the title of a book in the first page from Amazon, the block containing this title is recognised by VIPS, and a DOM tree is built for it as shown in Figure 7(a). Then, as the user selects the title of a book in the second page from eBay, another tree is similarly built as shown in Figure 7(b). Now, the two trees are merged to make an augmented tree as shown in Figure 7(c). Details of this process of tree alignment and merging are described in [Zhai and Liu, 2006]. The resulting tree, $\mathcal{AT}$, is used to identify informative blocks for new pages in the information extraction phase. The main idea is to find a cluster that contains blocks that are similar to user-preferred blocks represented by $\mathcal{AT}$. This requires a DOM tree T_b to be built for each block b in a cluster, and the similarity $W(\mathcal{AT}, T_b)$ between $\mathcal{AT}$ and T_b to be measured by using the tree matching method in [Yang, 1991]. This value is stored as the block similarity score, $Bscore(b)$. Then, the cluster similarity score, $Cscore(c)$, for a cluster c is obtained by taking the average of the $Bscore(b)$ values for all $b \in c$. Thus, the informative block cluster is the one with the maximum cluster similarity score, namely:

$$\begin{aligned} Bscore(b) &= W(\mathcal{AT}, T_b) \\ Cscore(c) &= \frac{\sum_{b \in c} Bscore(b)}{|c|} \\ inf\_cluster &= \arg\max_{c \in C}\{Cscore(c)\} \end{aligned} \tag{4}$$

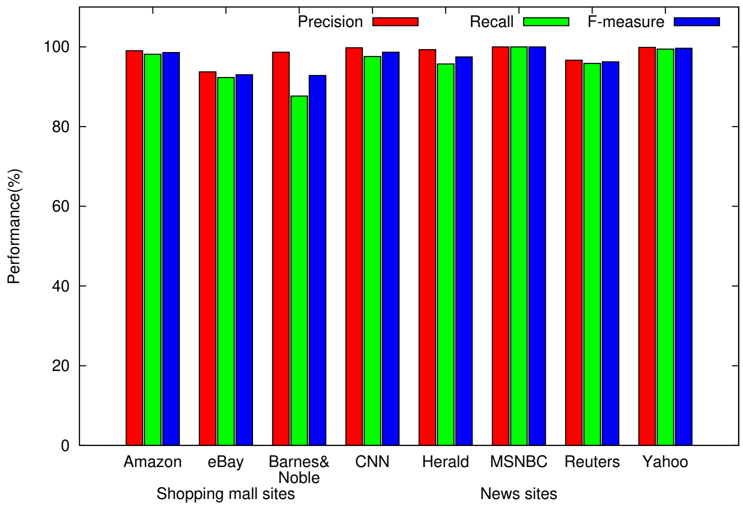
Algorithm 2 formalises these ideas. Note that this algorithm implements a function $InformClusterRecog : C \rightarrow c$ that determines the informative cluster from the set of clusters obtained by the block clustering algorithm, where $C = \{c_1, c_2, \ldots, c_n\}$ is the set of clusters obtained from block clustering, and c is either $c_i, i = 1, 2, \ldots, n$.

4 Empirical Evaluation

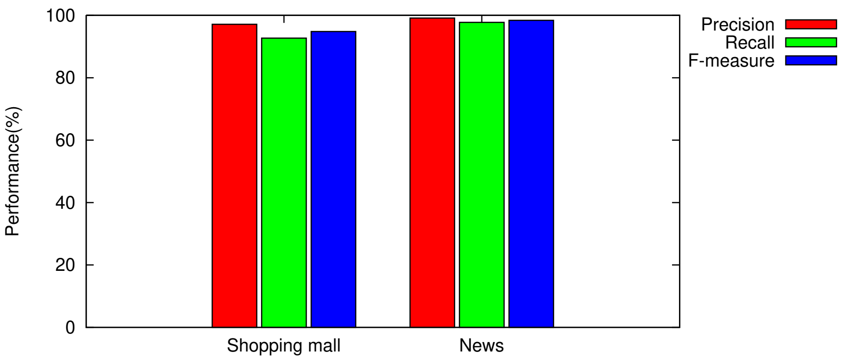
The sites we used to evaluate the RIPB algorithm and the RIPB-based information extraction system were shopping mall sites (Barnes&Noble, eBay, Amazon), and news sites (Yahoo News, Herald, CNN, MSNBC, and Rueters). We collected 100 pages from each site, i.e., 800 web pages were used for the experiments. Regarding the shopping mall sites, each page was either a listing page with a list of product descriptions or a details page with information about a single product; regarding the news sites, each page was an article page.

The first experiment consisted of evaluating the performance of RIPB regarding its ability to identify informative block clusters. The results are summarised in Figure 8 in terms of precision, recall, and F-measure. Figure 8(a) reports on individual sites, and Figure 8(b) on their categories. In the case of the shopping mall sites, it was usual to find adjacent blocks with information about products that had quite different structures, and could not be merged; this is the reason why the recall values are relatively low compared to the precision values. We have identified the following problems: first, the VIPS algorithm often results in incorrect block segmentation due to the PDoC value, which must be pre-set despite the number of blocks and their sizes are largely dependent on it; the most appropriate PDoC value is determined for each site through a number of experiments, so different PDoC values are used for different sites. Second, the recall is low when the threshold value $Median$ is small, which causes some related blocks not to be merged. Third, structurally similar but semantically unrelated blocks can be merged into the same cluster, which affects the precision measure. Despite of these problems, RIPB showed over 95% average accuracy for recognising informative block clusters.

The second experiment consisted of evaluating the effect of the RIPB algorithm on information extraction. We compared the performance of an information extractor when it is confronted with raw pages (Non-RIPB-IE) or just the blocks identified by RIPB (RIPB-IE). (The details on our information extractor are reported elsewhere [Kang and Choi, 2007].) In this experiment, the systems extracted titles, prices, and product reviews from the shopping mall sites, and titles and articles from the news sites. Figure 9 provides information about individual sites, and Figure 10 provides information about each category. Overall, RIPB-IE showed over 95% of extraction accuracy and outperformed non-RIPB-IE by about 17%. The main reason for this achievement is that the sources of information extraction are different. In RIPB-IE, the sources are the informative block clusters obtained by RIPB; consequently, the system had to locate the target information in a relatively small data area; in contrast, Non-RIPB-IE had to deal with the entire page containing complex content structures and many types of information, so it is more likely that it works wrongly or inefficiently.



(a) Performance comparison for recognising informative clusters



(b) Average performance comparison for the two groups

Figure 8: Recognising informative block clusters.

5 Conclusions

This article reports on the RIPB algorithm, a method to recognise the informative blocks in a web page for efficient information extraction. The RIPB algorithm is composed of three modules: visual page block segmentation, block clustering, and informative cluster recognition. Regarding block segmentation, it relies on the VIPS algorithm to analyse and partition a web page into a set of visually-separated blocks; regarding block clustering, it groups related blocks with similar content structures or patterns into a

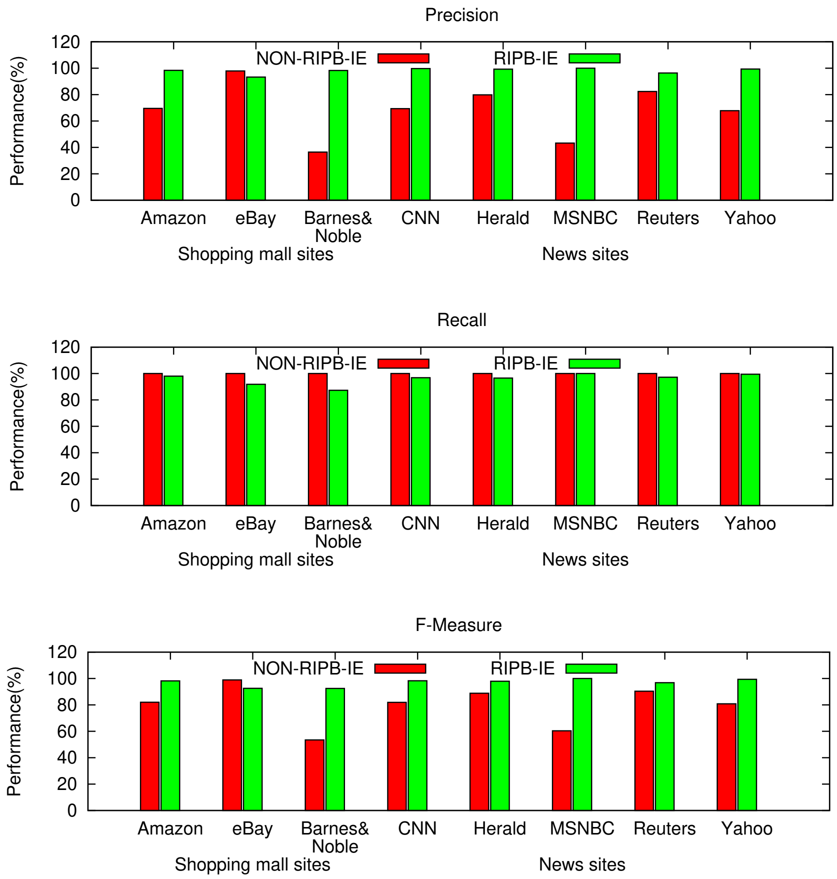
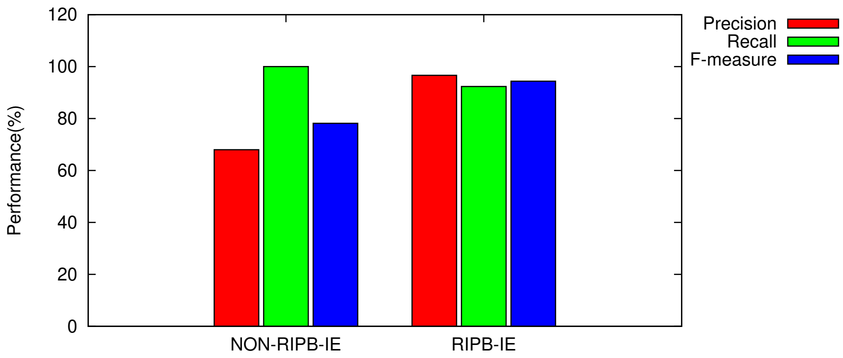


Figure 9: The effect of RIPB on information extraction.

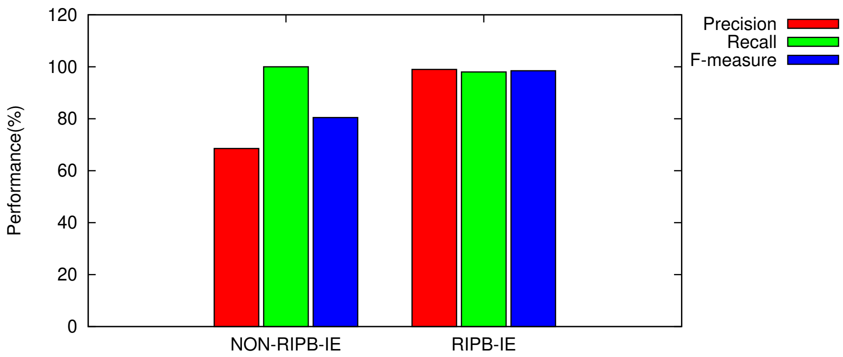
block cluster by using a tree edit distance method; regarding the recognition of informative clusters, it relies on tree alignment and matching algorithms.

We have performed several experiments to evaluate the performance of RIPB in information extraction for shopping mall sites and news sites. The first experiment tested how well RIPB recognises informative block clusters, and it proved more than 95% accuracy. The second experiment tested how RIPB affects the performance of information extraction by comparing with a non-RIPB method; RIPB-based information extraction showed more than 95% extraction accuracy and outperformed its non-RIPB counterpart by about 17%.

In summary, RIPB proves to be appropriate to improve the performance of information extraction. A limitation is that the recognition of the informative block cluster depends largely on the performance of VIPS. In other words, if the visual block



(a) Average performance for the shopping mall sites.



(b) Average performance for the news sites.

Figure 10: Performance comparison.

segmentation itself fails to generate the correct blocks or fails to cluster blocks that are informative but do not have similar content structure, the performance of information extraction might degrade since the source of information extraction contains noise blocks. The main reason for this problem can be found in the nature of the DoC value on which the VIPS algorithm relies. The preset value of DoC has a great impact on the performance of RIPB. We are currently working on determining the DoC value automatically and on developing a method for vision-based similarity measurement and clustering.

References

[Arasu and Molina, 2003] Arasu, A. and Molina, G.-H. (2003). Extracting structured data from web pages. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 337–348.

[Bille, 2005] Bille, P. (2005). A survey on tree edit distance and related problems. *Theoretical Computer Science*, 337(1–3):217–239.

[Buttler et al., 2001] Buttler, D., Liu, L., and Pu, C. (2001). A fully automated object extraction system for the world wide web. In *Proceedings of the 21st IEEE International Conference on Distributed Computing Systems*, pages 361–370.

[Cai et al., 2003] Cai, D., Yu, S., Wen, J.-R., and Ma, W.-Y. (2003). VIPS: A vision-based page segmentation algorithm. Technical Report MSR-TR-2003-79, Microsoft Research.

[Chang et al., 2006] Chang, C.-H., Kayed, M., Girgis, M., and Shaalan, K. (2006). A survey of web information extraction systems. *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1411–1428.

[Crescenzi and Mecca, 2004] Crescenzi, V. and Mecca, G. (2004). Automatic information extraction from large websites. *Journal of the ACM*, 51(5):731–779.

[Kang and Choi, 2007] Kang, J. and Choi, J. (2007). A preliminary report for an information extraction system based on visual block segmentation. Technical Report TR-IS-2007-1, Intelligent Systems Laboratory, Hanyang University.

[Kushmerick, 2000] Kushmerick, N. (2000). Wrapper induction: Efficiency and expressiveness. *Artificial Intelligence*, 118(1-2):15–68.

[Robinson, 2004] Robinson, J. (2004). Data extraction from web data sources. In *Proceedings of the 15th International Workshop on Database and Expert Systems Applications*, pages 282–288.

[Shi et al., 2005] Shi, Z., Milios, E., and Zincir-Heywood, N. (2005). Post-supervised template induction for information extraction from lists and tables in dynamic web sources. *Journal of Intelligent Information Systems*, 25(1):69–93.

[Turmo et al., 2006] Turmo, J., Ageno, A., and Català, N. (2006). Adaptive information extraction. *ACM Computing Surveys*, 38(2):#4.

[Wang and Zhou, 2003] Wang, Y. and Zhou, L. (2003). A hybrid method for web data extraction. In *Proceedings of the International Conference on Web Intelligence*, pages 417–420.

[Wong and Lam, 2007] Wong, T.-L. and Lam, W. (2007). Adapting web information extraction knowledge via mining site-invariant and site-dependent features. *ACM Transactions on Internet Technology*, 7(1):#6.

[Yang, 1991] Yang, W. (1991). Identifying syntactic differences between two programs. *Software Practice Experience*, 21(7):739–755.

[Yang and Zhang, 2001] Yang, Y. and Zhang, H. (2001). HTML page analysis based on visual cues. In *Proceedings of the 6th International Conference on Document Analysis and Recognition*, pages 859–864.

[Zhai and Liu, 2006] Zhai, Y. and Liu, B. (2006). Structured data extraction from the web based on partial tree alignment. *IEEE Transactions Knowledge and Data Engenieering*, 18(12):1614–1628.