Extraction of Semantic Text Portion Related to Anchor Link

Bui Quang Hung Masanori Otsubo Yoshinori Hijikata Shogo Nishida

Graduate School of Engineering Science, Osaka University, Osaka 560-8531, JAPAN

E-mail: {bqhung, ohtsubo, hijikata, nishida}@nishilab.sys.es.osaka-u.ac.jp

Abstract Recently, semantic text portion (STP) is getting popular in the field of Web mining. STP is a text portion in the original page which is semantically related to the anchor pointing to the target page. STPs may include the facts and the people's opinions about the target pages. STPs can be used for various upper-level applications such as automatic summarization and document categorization. In this paper, we concentrate on extracting STPs. We conduct a survey of STP to see the positions of STPs in original pages and find out HTML tags which can divide STPs from the other text portions in original pages. We then develop a method for extracting STPs based on the result of the survey. The experimental results show that our method achieves high performance.

Keyword text mining, web mining, semantic text portion, link structure, anchor, user experiment

1. Introduction

In the field of Web mining, many researchers come to focus on the link structure. When there is a link from a web page to another one, the former is called the original page and the latter is called the target page. One target page may have many original pages. One of the most important characteristics of the link structure is that the text portions around the anchors in the original pages describe the target pages [4]. Henzinger, in his survey on the link structure analysis [7], explains that this characteristic originates from the following human factor. Many authors of original pages create links because they think the links are useful for the readers. A link from an original page to a target page can be seemed as a recommendation about the target page by the author of the original page. The author also writes some texts around the anchor to explain the target page to the readers from his own viewpoint (also see Figure 1 as an example). These text portions are semantically related to the target page. We give the following definition about this kind of text portions.

Definition 1

Semantic text portion (STP) in an original page is a text portion which is semantically related to the anchor pointing to the target page.

Recently, STP is getting popular in the field of Web Mining. STPs can be used for many applications. One example is automatic summarization [1, 2, 6]. STPs may include important information about target pages. We can make summaries of target pages by collecting them. Another example is document categorization [5, 9, 11, 12, 13, 14]. Because the target page contains many noise parts such as banner ads and links for navigation, STPs may represent the content of the target page better. Compared to using the text of the target page, there is a possibility that we can make a better directory by using STPs.

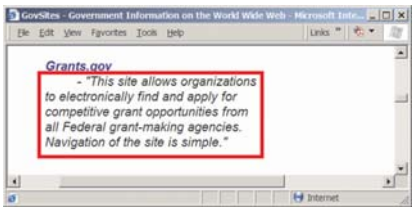


Figure 1. An example of explanation of the target page. The STP is the text portion around the anchor. Its content is *"This site allows organizations to electronically find and apply for competitive grant opportunities from all federal grant-making agencies. Navigation of the site is simple."* It is author's own viewpoint about the target page.

Researchers proposed various methods for extracting STPs. These methods are anchor-text method, fixed-window method, sentence-based method, paragraph-based method, and list-based method. The anchor-text method is the simplest one. It extracts the text portion between the tags <A> and </A> of the anchor. The fixed-window method extracts the anchor text and the pre-determined number of words around the anchor. The sentence-based method extracts one or more sentences around the anchor. The paragraph-based method extracts the paragraph which begins with the anchor followed by texts. The list-based method extracts the list item which includes the anchor.

These methods are too simple to extract all the STPs in one original page. The problem of extracting STPs is that they locate in various kinds of location like the text around the anchor, the page title, the list title, the first row of the table and so on. Therefore the previous methods cannot extract STPs in high precision and especially in high recall.

Our approach to solve this problem is as follows. We conduct a survey of STPs to see which kinds of text portion in an original page are related to the anchor. We hope that we will find out some HTML tags which can semantically divide STPs from the other text portions in original pages. Based on the result of the survey, we develop a method for extracting STPs. Our method represents an original page by a DOM tree to analyze its document structure. DOM (Document Object Model) is an API to access any parts of a Web page which is standardized by W3C [17]. Our method then extracts STPs by using specific HTML tags which are found in the survey.

The most serious shortcoming in the previous researches is that they did not survey where STPs are written in an original page and did not evaluate their methods from the viewpoint of extracting STPs. They only proposed their simple methods and used the text portions extracted by their methods for upper-level applications. They did not consider whether the extracted text portion itself is semantically related to the target page or not. In our research, we conducted a deep survey of the location of STPs and evaluated our method from the viewpoint of extracting STPs by inviting three evaluators. We made a dataset which consists of more than 1000 real original pages for the survey and a dataset which consists of 200 real original pages for the evaluation. The evaluators judged which text portions are real STPs in those pages. We decided on the text part which is a real STP by the majority vote. In the evaluation, we compared the texts extracted by our method to the real STPs given by evaluators. We then compared our method to the previous methods in extracting STPs. The experimental results showed that our method can achieve high precision and also the highest recall among the previous methods.

In brief, the contributions of this paper are as follows:

- We deeply survey the locations of STP in original pages for the first time.
- We propose a method for extracting STPs from the result of the survey.
- We evaluate extracted text portions by using real STPs given by evaluators for the first time.

The rest of this paper is organized as follows. Section 2 discusses the survey of STP and Section 3 explains our method for extracting STPs. In Section 4, we evaluate STPs extracted by our method and compare our method to other methods. Section 5 provides some concluding remarks and directions for future research.

2. Survey of STP

In this section, we explain our survey of STP. The purpose of this survey is to see the positions of STPs in original pages and find HTML tags which can divide STPs from the other text portions in original pages. We realize that there are two types of STP from the viewpoint of its locations. One type exists around the anchor. This means that it directly includes the anchor. The other type exists in the upper-level structure of the original page. A web page is described in HTML and all parts of the web page (document) are structured by tags. The latter type does not touch the anchor and exits in the upper-level of this document structure. We call the former type *the Local Semantic Portion (LSP).* We call the latter type *the Upper-level Semantic Portion (USP).* Our survey consists of the survey of LSP and the survey of USP.

2.1. Dataset and survey method

We prepared 1108 real original pages in our survey. These 1108 web pages are 752 original pages of 50 official target pages such as a government's web page and a company's web page and 356 original pages of 50 personal target pages such as an individual's web page about his hobby. We collected these original pages as follows. We randomly selected 50 official target pages and 50 personal target pages from Open Directory [3]. For each target page, we found its original pages by using Google [8]. To get original pages of a target page, Google offers a search function by the query type *"link:URL of the target page".* We used 20 original pages at most for each target page.

We invited three evaluators to give us the right answer of USPs. The method we used in the survey is as follows. For each original page in the dataset, we show the three evaluators its content and the anchor pointing to its target page. The evaluators see the content of the target page. After that, we ask them to judge which text portions are semantically related to the anchor. We define a real STP as the text portion which is judged to be semantically related to the anchor by at least two evaluators.

2.2. Survey of LSP

2.2.1. Positions of LSPs in original pages

Through the survey, we realized that LSPs are located in one of the following five places: table, list (ordered and un-ordered list), definition list, paragraph, or DIV object. Table 1 shows the number of LSPs in each place in 1108 original pages.

Position	Total
Paragraph	320
Ordered and un-ordered list	354
Definition list	56
Table	339
DIV	39

Table 1. Number of LSPs in each place

2.2.2. HTMLs tags for dividing LSPs from the other text portions

This subsection explains the result of survey about what kind of HTML tag can divide the LSP from the other text portions in each place.

a) When the LSP is in a paragraph. We found that there are the following two cases when a LSP is in a paragraph. After here, we call the paragraph which has the anchor to the target page the *current paragraph.*

Case 1: *The LSP covers the whole paragraph.* In this case, we realized that there is only one anchor in the paragraph or there is no line feeder tag in the paragraph. We found that <P> tag and </P> tag can divide the LSP from the other text portions.

Case 2: *The LSP is one part of the paragraph.* In this case, we realized that there are several anchors and several line feeder tags in the paragraph. We found that there are four sub-cases as shown in Figure 2.

- ***Sub-case 1:*** The paragraph begins with an anchor followed by texts and there is no line feeder tag between the anchor and the following texts.
- ***Sub-case 2:*** The paragraph begins with an anchor followed by texts and there are one or more line feeder tag(s) between the anchor and the following texts.
- ***Sub-case 3:*** The paragraph begins with texts and there is no line feeder tag between the texts and the following anchor.
- ***Sub-case 4:*** The paragraph begins with texts and there are one or more line feeder tag(s) between the texts and the following anchor.

We found that in Sub-case 1 and Sub-case 2, the LSP is divided by the line feeder tag before the anchor and the line feeder tag before the next anchor. We also found that in Sub-case 3 and Sub-case 4, the LSP is divided by the line feeder tag after the previous anchor and the line feeder tag after the anchor.

b) When the LSP is in a DIV object. We found the following two cases.

Case 1: *The LSP covers the whole DIV object.* We found that the LSP is divided from the other text portions by the <DIV> tag and </DIV> tag.

Case 2: *The LSP is one part of the DIV object.* We found that the LSP is divided from the other text portions by line feeder tags like a)-Case 2.

c) When the LSP is in a list. We found the following two cases. After here, we call the list item which includes the anchor to the target page the *current list item.* We call the list which has the current list item the *current list.*

Case 1: *The LSP covers the whole current list item.* We found that the LSP is divided from the other text portions by the <LI> tag and </LI> tag.

Case 2: *The LSP covers one part of the current list item.* We found that the LSP is divided from the other text portions by line feeder tags like a)-Case 2.

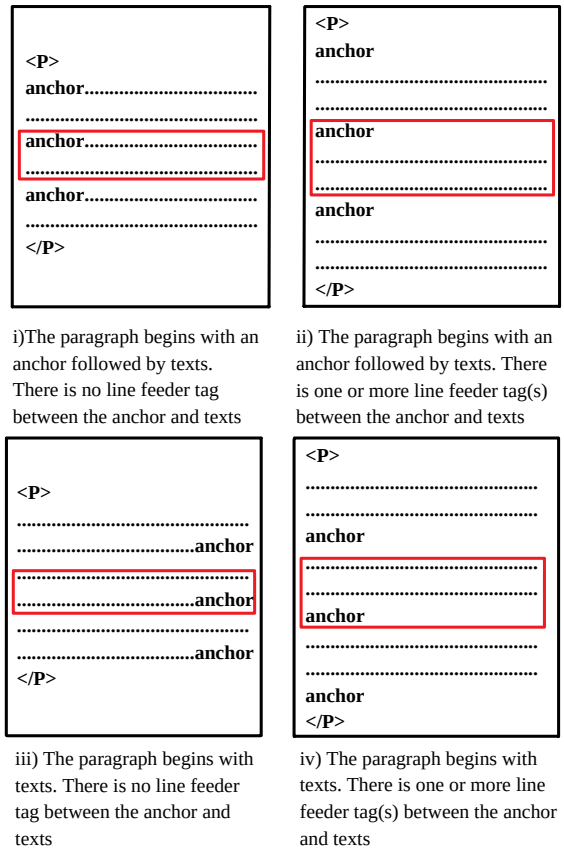


Figure 2. Four sub-cased when the LSP is one part of a paragraph

d) When the LSP is in a definition list. We found the following two cases (also see Figure 3).

Case 1: *The LSP covers the definition term including the anchor and the definition description of the definition term.* We found that the LSP is divided from the other text portions by the <DT> tag before the anchor and the </DD> tag after the anchor

Case 2: *The LSP is a part of the definition description.* We found that there are several anchors and several line feeder tags in the definition description. The line feeder tag before the anchor and the line feeder tag before the next anchor can divide the LSP from the other text portions.

e) When the LSP is in a table:

We found that there are the following five cases (also see Figure 4). After here, we call the cell, where the anchor to the target page exists, the *current cell.* We call the row which has the current cell the *current row.* We call the table which has the current row the *current table.*

Case 1: *The LSP covers the whole current cell.* We found that the LSP is divided by <TD> tag and </TD> tag.

Case 2: *The LSP is a part of the current cell.* We found that the LSP is divided from the other text portions by line feeder tags like a)-Case 2.

Case 3: *The LSP covers several cells (not all cells) in the current row.* We found there are the following two sub-cases.

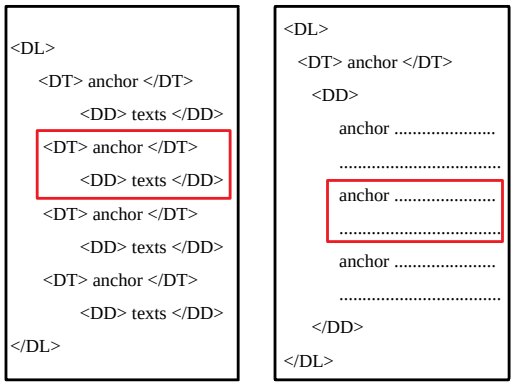
- ***Sub-case 1:*** The current row begins with an anchor.
- ***Sub-case 2:*** The current row begins with texts.

We found that in Sub-case 1, the <TD> tag before the anchor and the </TD> tag before the next anchor divide the LSP from the other text portions. We found that in Sub-case 2, the <TD> tag after the previous anchor and the </TD> tag after the anchor can divide the LSP from the other text portions. Case 4: The LSP covers the current row.

We found that the <TR> tag and </TR> tag of the current row can divide the LSP from the other text portions. Case 5: The LSP covers several rows of the table. We found there are the following two sub-cases.

- ***Sub-case 1:*** The table begins with an anchor.
- ***Sub-case 2:*** The table begins with texts.

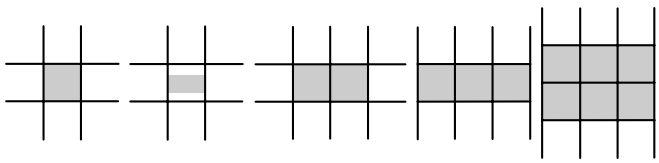
We found that in Sub-case 1, the <TR> tag before the anchor and the </TR> tag before the next anchor can divide the LSP from the other text portions. We found that in Sub-case 2, the <TR> tag after the previous anchor and the </TR> tag after the anchor can divide the LSP from the other text portions.



i) The LSP covers the definition term including the anchor and the definition description.

ii) The LSP is a part of the definition description and there are several anchors and several line feeder tags in the definition description.

Figure 3. Two cases in which the LSP is in a definition list



i) The LSP covers the current cell

ii) The LSP is one part of the current cell

iii) The LSP covers several cells of the current row

iv) The LSP covers the current row

v) The LSP covers several rows

Figure 4. Five cases in which the LSP is in a table

2.2.3. Summary of HTML tags for dividing LSPs from the other text portions

We found there are three kinds of HTML-tag set which can divide LSPs from the other text portions in original pages: the set including only the parent tag (parent-tag set), the set including only the sibling tag (sibling-tag set), and the set including the ancestor tag without the parent tag or both the parent tag and its sibling tag (relative-tag set).

A parent-tag set consists of the parent tag which directly includes the anchor. Using the parent-tag set can divide a LSP from the other text portions when the LSP covers the whole of the paragraph, list item, table cell, or DIV object. For example, when a LSP covers the whole paragraph, the LSP can be divided by the <P> tag and </P> tag of the paragraph.

A sibling-tag set consists of the sibling tag which is at the same level as the <A> tag of the anchor in the document structure. Using a sibling-tag set can divide a LSP from the other text portions when the LSP covers one part of the paragraph, list item, table cell, or DIV object. For example, when a LSP covers one part of the paragraph which includes the anchor, the LSP is divided from the other text portions by the two sibling tags which are the line feeder tag before the LSP and line feeder tag after the LSP.

A relative-tag set consists of either the ancestor tag without the parent tag or the both of the parent tag and its sibling tag in the document structure. Using a relative-tag set can divide a LSP from the other text portions when the LSP covers several cells (not all cells) of the current row, the current row, or several rows of the current table. For example, when a LSP covers several cells of the current row and the current row begins with an anchor, it is divided by the <TD> tag which is the parent of the anchor to the target page and the </TD> tag of the next cell which is its sibling tag. Furthermore, using a relative-tag set can divide a LSP from the other text portions when the LSP covers the definition term including the anchor and the definition description of the definition term.

Table 2 shows the numbers of LSPs which can be divided from the other text portions by using each type of tag set.

	Parent-tag set	**Sibling-tag set**	**Relative-tag set**
Paragraph	216	102	0
Ordered and un-ordered list	329	25	0
Definition list	0	12	44
Table	165	63	113
DIV	21	18	0

Table 2. Numbers of LSPs which can be divided from the other text portions by using each type of tag set

2.3. Survey of USP

This subsection explains the result of the survey about which kind of location the USP exists and what kind of HTML tag can divide the USP from the other text portions. Table 3 shows its result. The left column shows the type of upper-level object which is related to the anchor, the center column shows the number of pages which has each type of upper-level object, and the right column shows the HTML tags which can divide the USP from other text portions.

In our survey, we found 1097 original pages in which the page title is related to the anchor. There were 739 original pages in which headers (from H1 to H6) are related to the anchor. We also realized that if there are several headers at the same level (for example, there are several headers H3), the header nearest to the anchor is related to the anchor.

Upper-level object	Total	HTML tags used for extracting
Page title	1097	<Title> and </Title>
H1	326	<H1> and </H1>
H2	209	<H2> and </H2>
H3	153	<H3> and </H3>
H4	18	<H4> and </H4>
H5	26	<H5> and </H5>
H6	7	<H6> and </H6>
Table Header	6	<TH> and </TH>
The first row of the current table	48	<TR> and </TR>
The first row of an upper-level table	82	<TR> and </TR>
The text portion at the top of the current list	64	line feeder tags
Another row of the current table	46	cannot extract
Another row of an upper-level table	167	cannot extract
Another table	278	cannot extract
Another list	36	cannot extract
Another paragraph	372	cannot extract

Table 3. Result of the survey of USP

We found six original pages in which the table header of the current table is related to the anchor. We realized that authors of original pages rarely use table headers. They usually use the first row of the current table or the first row of the upper-level table instead of the table header. Therefore, in 48 original pages, the fist row of the current table is related to the anchor; and in 82 original pages, the first row of the upper-level table is related to the anchor. We also realized that, some of the authors of original pages also write some texts related to the anchor in the row before the current row in the current table (not the first row) or in a row of the upper-level table (not the first row). We found 46 original pages with the former case and 167 original pages with the latter case.

We also found that the authors of original pages usually write some texts at the top of the list as a list title. There were 64 original pages in which the text portion at the top of the current list is related to the anchor. We realized that the numbers of words of these text portions are small. The biggest one among them is 19.

In many original pages, there is an object, which does not directly include the anchor, but is related to the anchor. As explained in the above, there were text portions related to the anchor in another row of the current table or in another row of the upper-level table. There were 278 original pages in which the text portion in another paragraph from the current paragraph is related to the anchor. There were 36 original pages in which the text portion in another list from the current list is related to the anchor. There were 372 original pages in which the text portion in another table from the current table is related to the anchor. Currently, it is impossible to extract these text portions because this requires that the computer can semantically understand the content of the text.

3. Extraction of STP

In this section, we propose a method for extracting STPs based on the result of the survey of STP.

3.1. Extraction of LSP

Firstly, our method represents an original page by a DOM tree. It then identifies which location (paragraph, list item, definition list, table, or <DIV> object) the anchor to the target page belongs to. After that, the method extracts the LSP from the identified location. The detail of the method is as follows.

The method identifies which location the anchor belongs to according to the type of the parent tag as follows:

- <P>: the anchor is in a paragraph.
- <LI>: the anchor is in a list item.
- <DT> or <DD>: the anchor is in a definition list.
- <TD>: the anchor is in a cell table.
- <DIV>: the anchor is a DIV object.

Then the method extracts the LSP from each location as follows:

a) If the anchor is in a paragraph, list item, definition object (<DD>) or DIV object. The method checks the number of line feeder tags in the parent object to the anchor.

- If there is no line feeder tag, it then extracts the whole texts of the object.
- If there is one or more line feeder tag(s), it then checks the number of anchors in the object. If there is only one anchor, it then extracts the whole text of the object. If there are several anchors, it then checks whether the object begins with an anchor or texts. If the object begins with an anchor, it extracts the text portion between

the line feeder tag before the anchor and the line feeder tag before the next anchor. If the object begins with texts, it extracts the text portion between the line feeder tag after the previous anchor and the line feeder tag after the anchor.

b) If the anchor is in a cell of a table. The method tries to expand to nearby cells by following the left and right directions from the current cell. It repeats this expansion until it meets a cell which includes a different anchor. If it can expand to all cells of the current row which includes the current cell, it tries to expand to nearby rows by following the up and down directions. It repeats this expansion until it meets a row which includes a different anchor. There are the following four cases in the result of this expansion:

■ ***Case 1:*** *The method cannot expand to any other cells*. The method extracts the LSP from the current cell by the same method as in a).

■ ***Case 2:*** *The method can expand to other cells but it cannot expand to all cells of the current row.* The method then checks whether the current row begins with an anchor or texts. If it begins with an anchor, the method extracts the text portion between the <TD> tag before the anchor and the </TD> tag before the next anchor. If it begins with texts, the method extracts the text portion between the <TD> tag after the previous anchor and the </TD> tag after the anchor.

■ ***Case 3:*** *The method can expand to all cells of the current row.* It extracts the whole texts in the current row.

■ ***Case 4:*** *The method can expand to other rows of the table*. It checks whether the table begins with an anchor or texts. If it begins with an anchor, the method extracts the text portion between the <TR> tag before the anchor and the </TR> tag before the next anchor. If it begins with texts, the method extracts the text portion between the <TR> tag after the previous anchor and the </TR> tag after the anchor.

c) If the anchor is in <DT> object of a definition list. It extracts the whole texts of the <DT> object and the whole texts of its <DD> object.

3.2. Extraction of USP

Our method extracts USPs as follows:

- It extracts the page title and all the upper headers from H1 to H6. If there are several headers at the same level, it extracts the nearest one to the anchor.
- It checks whether the anchor is in a table. If the anchor is in a table, it checks whether a table header exists.

- If a table header exists, the method extracts the table header.

- If a table header does not exist, the method checks whether or not the first row of the current table satisfies at least one of the following two conditions. (1) The number of its cells is smaller than the number of cells in the other rows. (2) There is no anchor in it while there are anchor(s) in all the other rows of the current table. If the first row of the current table satisfies at least one condition, the method extracts the first row of the current table. If the first row of the current table does not satisfy any condition, the method checks whether or not the first row of the upper-level table (if it exists) satisfies at least one of the above two conditions. If it satisfies at least one condition, the method extracts it. If it does not satisfy, the method continues to check the first row of the upper-level table of the previous upper-level table (if it exists). The method repeats this process until it finds out the first row which satisfies at least one condition or there is no more upper-level table.

■ The method checks whether the anchor is in a list item. If it is in a list item, the method checks whether there is a text portion at the top of the list. If there is a text portion and its number of words is smaller than a threshold a, the method extracts this text portion. We set a as 20 because in our survey of USPs, there is no list title which has the number of words which is greater than 19.

4. Evaluation

In the previous researches, they did not evaluate their methods from the viewpoint of extracting STPs. In our research, we invited three evaluators to participate in our experiments to give the real STPs. We evaluated the extracted text by using the correct answer of STP given by the evaluators. We also compared our method to other conventional methods in extracting STPs.

4.1. Dataset and experimental method

The dataset we prepared for our experiments contains 200 original pages. This dataset is different from the dataset which we used in our survey of STP.

The experimental method we used in our experiments is as follows:

(1) Identifying real STPs in original pages. For each *i-th* original page (i =1...200) in the dataset, we show the evaluators its content and the anchor to the target page. The evaluators see the content of the target page. After that, we ask them to extract STPs by themselves. We call the STPs extracted from the *i-th* original page by three evaluators P_{iA},P_{iB}, and P_{iC} (A, B, and C are IDs of three evaluators). We call the real STP in the *i-th* original page P_i.

(2) Calculating the precision and the recall. We call the text portion extracted from the *i-th* original page by the extraction method S_i. Let $|S|$ be the length of a text portion S (number of words in the text portion S). The $precision_i$ *and the* $recall_i$ *are* calculated by the following two equations:

$$precision_i = \frac{|P_i \cap S_i|}{|S_i|} \qquad recall_i = \frac{|P_i \cap S_i|}{|P_i|}$$

The precision and the recall of the method when it extracts STPs from the dataset of 200 original pages are calculated by the following two equations:

$$precision = \frac{1}{200}\sum_{i=1}^{200} precision_i \qquad recall = \frac{1}{200}\sum_{i=1}^{200} recall_i$$

(3) Calculating the average number of words of the extracted STPs. We calculate the average number of words of the text portions extracted by the method because this number reflects the precision and the recall.

4.2. Evaluation of our method

We evaluated our method in extracting LSPs, USPs, both LSPs and USPs. Table 4 shows the experimental results. Experimental results show that our method extracts LSPs in high precision (97.01%) and in high recall (93.94%). The number of words in the texts extracted as LSPs (20.36 words) is quite similar to the average number of words of the real LSPs (21.07 words). From this result, we can see that our method can identify the positions of LSPs in original pages accurately.

	Precision (%)	Recall (%)	Avg. number of words of extracted texts	Avg. number of words of real STPs
LSPs	97.01	93.94	20.36	21.07
USPs	89.43	74.35	8.54	9.35
both of LSPs and USPs	94.08	85.03	28.89	30.43

Table 4. Evaluation of our method for extracting STPs

Our method extracts USPs in 89.43% precision and in 74.35% recall. The average number of words in the texts extracted as USPs (8.54 words) is almost same as the average number of the real USPs (9.35 words). These precision and recall are smaller than the precision and the recall in extracting LSPs. This can be explained as follows. Based on the result of the survey of USP, our method extracts the page title, the headers (H1~H6), the first row of the current table, the first row of the upper-level table, and the text portion at the top of the current list. However, in some original pages, these text portions are not related to the anchor. For example, some authors put the same name (in most cases, the name of the web site) to all pages. Some authors use headers or tables not for structuring the content of the document but for decorating the web page or creating the layout for the web page. This is why our method extracts noise keywords. Our method cannot extract STPs in another paragraph, in another row of the current row (not the first row), in a row of the upper-level table (not the fist row), in another table or in another list. This means that there are STPs which exist in popular places and our method cannot extract.

Our method extracts both LSPs and USPs in 94.08% precision and in 85.03% recall. The average number of words of extracted texts is 28.89. This is almost same as the average number of the real STPs (30.43 words). We do not know this precision and recall is high among other existing methods. The next subsection compares our method to the existing methods.

4.3. Comparison of our method to other methods in extracting both LSPs and USPs

We compared our method to the anchor-text method, the fixed-window method, the sentence-based method, the paragraph-based method, the list-based method, the object-based method, the method which extracts all upper-level objects and Roy's method in extracting both LSPs and USPs. Table 5 shows the results.

Method	Precision (%)	Recall (%)	Avg. number of words of extracted texts
Our method	94.08	85.03	28.89
Anchor-text method	100	19.37	3.43
Fixed-window method (50 words around the anchor)	29.52	52.78	51.38
Sentence-based method (3 sentences around the anchor)	60.10	51.03	25.54
Paragraph-based method	100	18.32	7.16
List-based method	84.24	19.78	7.92
Object-based method	70.95	50.28	367.45
Extracting all upper-level objects	13.01	39.98	1081.71
Roy's Method	84.17	20.06	5.89

*Average number of words of the real STPs is 30.43

Table 5. Comparison of our method to other methods in extracting both LSPs and USPs

The anchor-text method, the fixed-window method, the sentence-based method, the paragraph-based method, the list-based method, and the object-based method extract both LSPs and USPs in the same precision as the precision when they extract LSPs. It is because they do not extract any text portions in the upper-level structure of the original page. This makes the recall to extract both LSPs and USPs smaller than the recall to extract LSPs. Similarly, in Roy's method and the method which extracts all upper-level objects, the precision to extract both LSPs and USPs is same as the precision to extract USPs. The recall to extract both LSPs and USPs is smaller than the recall to extract USPs.

Our method extracts both the text portion around the anchor and the text portion in the upper-level structure in the original page. Therefore, it achieves the highest recall

(85.03%) compared to the other methods. The precision of our method is 94.08%. It is lower than the precision of the anchor-text method (100%) and the sentence-based method with Option 1 (100%). Because anchor text and the sentence, which includes the anchor, are always related to the anchor. We can see that these methods take the safest measure to extract STPs. Therefore their recalls are low. From this result, we can see that our method brings a good balance between precision and recall.

5. Conclusion and Future Work

This paper concentrates on extracting a semantic text portion (STP) from an original page. STP is a text part which is related to the anchor to the target page. Firstly, we found two types of STP: local semantic portion (LSP) and upper-level semantic portion (USP). We conducted a survey for each type of STP by using 1108 real original pages to find HTML tags which can semantically divide STPs from the other text portions in original pages. We invited three evaluators to participate in our survey to judge which text portions in an original page are STPs. We then developed a method for extracting STPs based on the result of the survey. Our method represents an original page by a DOM tree to analyze its document structure. It then extracts STPs by using specific set of HTML tags which are found in the survey. We then conducted experiments to evaluate our method and compare it to the previous methods in extracting STPs. We evaluated the texts extracted by each method by comparing them to the real STPs given by three evaluators. The experimental results showed that our method achieves high precision and the highest recall compared to the previous methods.

The shortcoming of our survey and our extraction method is that they just consider the relevance to the anchor but do not consider the type of relevance. We found in the survey that STPs are either facts or people's opinions (evaluation and categorization). In some applications, we should select the type of STPs. In the summarization, the user may want to see only the people's evaluation. In the categorization, the user may want to see a categorization created by a user group from their viewpoints. We will study STPs by considering whether they are facts, people's evaluation or people's categorization as a future work.

References

[1] J.Delort, B.B.Meunier, M.Rifqi, Enhanced Web Document Summarization Using Hyperlinks, Proc. 14th ACM Conference on Hypertext and Hypermedia (HT'03), pp.208-215, 2003.

[2] E. Amitay, Using common hypertext links to identify the best phrasal description of target web documents. Proc. Post-Conference Workshop on Hypertext Information Retrieval for the Web (SIGIR'98), pp.271-276, 1998.

[3] Open Directory http://dmoz.org/

[4] B. D. Davison, Topical Locality in the Web. Proc. 23rd Annual International Conference on Research and Development in Information Retrieval (SIGIR2000), pp.272-279, 2001.

[5] S.Roy, S.Joshi, R.Krishnapuram, Automatic Categorization of Websites based on Source Type, Proc. 15th ACM Conference on Hypertext & Hypermedia, pp.38-39, 2004.

[6] E.Amitay and C.Paris, Automatically summarizing web sites: Is there a way around it?, Proc. ACM 9th International Conference on Information and Knowledge Management , pp.173-179, 2000.

[7] M.Henzinger, Link Analysis in Web Information Retrieval, IEEE Data Engineering Bulletin, 23(3):3-8, 2000.

[8] Google http://www.google.com/

[9] M.Otsubo, B.Q.Hung, Y.Hijikata, S.Nishida, A Basic Study on Web Page Classification Method by Anchor-Related Text, Proc. SICE Annual Conference, pp.3622-3625, 2005.

[10] S. Chakrabarti, B. Dom, D.Gibson, J. Keinberg, P. Raghavan and s. Rajagopalan, Automatic Resource list Compilation by Analyzing Hyperlink Structure and Associated Text. Proc. 7th International World Wide Web Conference, pp.65-74, 1998.

[11] E.J. Glover, K.Tsioutsiouliklis, S.Lawrence, D.M. Pennock, and G.W. Flake, Using web structure for classifying and describing web pages, Proc. 11th International World Wide Web Conference, pp.562-569, 2002.

[12] G. Attardi, S. Di Marco, D. Salvi, Categorisation by context, Journal of Universal Computer Science, 4(9):719-736, 1998.

[13] J. Furnkranz, Exploiting Structural Information for Text Classification on the WWW, Proc. 3rd Symposium on Intelligent Data Analysis (IDA-99), 1999.

[14] A.Blum, T.Mitchell, Combining Labeled and Unlabeled Data with Co-Training, Proc. 11th Annual Conference on Computational Learning Theory, pp.92-100, 1998.

[15] Jon M.Kleinberg, Authoritative Sources in a Hyperlinked Environment, Journal of the ACM (JACM), 45(5):604-632, 1999.

[16] S. Brin, L.Page, The Anatomy of a Large-Scale Hypertextual Web Search Engine, Proc. 7th International Conference on World Wide Web, pp.107-117, 1998.

[17] http://www.w3.org/DOM/