

A Scalable Image Snippet Extraction Framework for Integration with Search Engines

Sheikh Muhammad Sarwar¹, Md. Mustafizur Rahman¹, Md. Haider Ali¹ & Ashique Mahmood Adnan²

¹ Department of Computer Science and Engineering, University of Dhaka, Dhaka, Bangladesh

² Department of Computer Science and Engineering, University of Liberal Arts, Dhaka, Bangladesh

Correspondence: Sheikh Muhammad Sarwar, Department of Computer Science and Engineering, University of Dhaka, Dhaka, Bangladesh. E-mail: sheikh.sarwar@ulab.edu.bd

Received: October 26, 2012 Accepted: January 8, 2013 Online Published: January 13, 2013

doi:10.5539/cis.v6n1p89

URL: <http://dx.doi.org/10.5539/cis.v6n1p89>

Abstract

Search result visualization is a task performed by search engines that enables users to find their desired documents, in an effective and efficient manner. Image based summary or best images of a web document, displayed as a part of the visualization process, has become indispensable, as a human perceives images instantaneously. But, selection of the best image increases latency in search result generation, and workload for the search process. In this paper, we propose and implement a search framework by integrating text and image search engines that increases the speed of extracting a representative image of a web document. Text associated with an image, image area and position are incorporated with the ranking function that finds the image snippet. By comparison, we show that our framework significantly improves over the existing ones in terms of time complexity, while maintaining the quality of image based summaries.

Keywords: information retrieval, web search

1. Introduction

Over the last couple of decades, we observe an exponential growth in both the size of the World Wide Web (WWW) and the number of Internet users. The availability of knowledge and services are the main reason behind this enormous growth. WWW contains web sites and services which are related to news broadcasting, financial services, retail shopping, social networking, multimedia sharing, personal information publishing, business developing etc. As a consequence, the number of Internet users is also increasing and they are from different localities, age groups and professions. The users also have different types of queries and they need to locate their answers quickly. To meet the diversified demand of the users, the requirement of a tool like search engine has become inevitable.

The basic service that is offered by a web search engine is providing the user a list of web links dynamically according to the user query. But in order to let the users select his/her desired web link, search engines provide extra information with each of the links like the title of the web page, useful links in the web page, excerpt of the web page content in which query terms appear and in some cases, even a snapshot of the web page. Addition of these visual aids works as a clue for the users to find their desired web page without visiting the web page, which in turn saves valuable time and increases user satisfaction. However, most of the modern search engines have one important visual component in common, which is a text snippet provided with the link of each web page in the search result. A snippet helps a user greatly to find relevant pages without having to look at the details of the pages and shorten the search time. Snippets also provide necessary clues for query reformulation. The users of the search engine usually provide queries which contain few numbers of keywords. The short queries create the problem of ambiguity for search engine as there are different types of topics containing same keywords. Snippets are extracted based on user query and usually contain a sentence or passage of text in which the query terms used by the user are present. Common strategies for extracting and presenting text snippets to meet user needs either process document fragments that have been delimited a priori, or use a sliding window of a fixed size to highlight the results (Li & Chen, 2010).

Interaction with a search engine becomes easier when imagery is included in the search results, as images are often perceived quickly by human. Importance of images in education and understanding of a topic has been

found in research (Keegan, 2007). An image snippet is a salient and representative image from a web page, which is selected based on the intensity of semantic relation between the image and user query. The performance of image snippet extraction component of a search engine depends on three key issues. The first key issue is the total latency in search result generation for addition of the component. The second key issue is the processing overhead imposed on the search engine back-end. Even though search engine vendors use sophisticated hardware, less computational complexity would eventually lead to better throughput. But, decreasing the complexity should not be performed at the cost of an unsatisfied user with an inelegant, insensitive and tasteless search interface and that is the third key issue. In this work, we propose an image snippet generation framework, implement it and analyze its performance. The proposed framework performs better than existing ones in terms of time complexity.

2. Background

2.1 Web Page Parsing for Indexing

In order to index a web page for retrieval, the first step is to parse the web page. A web page is a semi-structured HTML document containing different elements like title, meta, header, paragraph etc. HTML Parsing is necessary to extract the contents for each of the above mentioned elements. Each HTML element contains several other elements in them, and that is a problem, as extracted contents are full of HTML tags. Those tags are further can be further eliminated to obtain clean content. The clean content for each of the fields like title, meta, header, bold contains no HTML tag embedded inside.

2.2 Vector Space Model

Vector Space Model (VSM) is a statistical model for Information Retrieval (Wong & Raghavan, 1984). In this model a document and a query both are represented as vectors. SMART Information Retrieval System was the first to use VSM, which was first developed by Gerard Salton and his colleagues (Salton, Wong, & Yang, 1975). The main idea of VSM is to model a document as a point in space. The space can contain any number of dimensions. So, for n number of documents in the collection there can be n points, and the points that are close together in the space are considered similar according to the semantics, whereas points that are far apart are considered as semantically distant (Turney & Pantel, 2010). As an Information Retrieval System answers queries, the representation of a query as a point in the same space as the document collection is also considered. When ranking is performed, the document that is closest to the query is ranked first and the others are also ranked in the same fashion. One of the most popular methods to determine the closeness of the query and the document, and rank them in order of relevance is Cosine Similarity (Garcia, 2011).

2.3 Lucene Search Framework

Lucene is a high performance, scalable text searching library. It is used by applications, which need indexing and searching. Lucene is a mature, free, open-source project implemented in Java; licensed under the liberal Apache Software License (Foundation, 2011). Lucene combines Boolean Model (BM) and Vector Space Model (VSM) of IR. BM can specify whether a term belongs to a document or not. No further information regarding the document, such as to which degree a document matches a query cannot be derived from BM. So, ordered ranking is not provided in BM.

2.4 Image Annotation

Image annotation is the process of assigning textual content to an image. In a web page, related text with an image can be found. In web image annotation process, all the occurrences of the HTML element “img” are found using an HTML Parser. After that, source URL for every image is retrieved using the parser from the “src” attribute in the “img” element. For extracting the name of an image file and the alternative text associated with it, the attributes of the “img” element is examined. The process of surrounding text extraction is a challenging task. Surrounding texts are the text near an image in the web page.

3. Related Works

There have been a lot of researches performed for better search engine visualization. Most of the works were focused on thumbnails. Thumbnails are miniature versions of web pages. Researchers also focused on visual tags and visual snippets. But little study on scalability for visualization purposes was performed. In the sections below short descriptions on visualization researches are provided.

3.1 Visual Summaries

Visual summaries of web contents exhibited by search engines provide important clues for the user to find the appropriate document. Visual summaries are essential and undeniable component when considering the

re-visitation process of the users. Re-visitation or re-finding is the process of finding a web document which was previously found using a search engine. To find or revisit a web document the user recalls the query that he had previously issued to the search engine to find the document. According to previous literatures, there are four types of visual summaries which are each found useful to some extent for ensuring better search experience for the user. They include thumbnails, salient images, visual snippets and visual tags. When visual summaries are present users do not have to spend time to find their relevant keywords in the text based summaries. Snippets, thumbnails, or video previews represented as visual aids give searchers rapid access to the relevant Web resources (Dork, Carpendale, & Williamson, 2009). Compared with the text the time for comprehending an image is as small as that of reading one or two words, whereas the amount of information an image exhibits is the same as a passage of text (Xue, Zhou, & Zhang, 2008).

3.2 Thumbnails

A thumbnail is a miniature version of a web page (Aula, Khan, Guan, Fontes, & Hong, 2010). Thumbnails are the most common types of visual summaries and have been studied by many researchers (Aula et al., 2010; Cockburn, Gutwin, & Alexander, 2006; Dziadosz & Chandrasekar, 2002). With the help of thumbnails users can get the layout of the web page without actually visiting the page and recognize if the web page was previously visited. Thumbnails, even though found as an effective tool for web page summarization in some researches, major search engines have not yet adapted the concept of thumbnails. It's difficult to judge the relevance of a web page using thumbnails. But, thumbnails can act as an effective visual cue for the users to find out the pages seen before.

3.3 Salient Images/Image Snippets

A web page may contain any number of images in it to enrich its visual content and an image snippet is one of them. But an image snippet is chosen dynamically as it has to be closely related with the user information need and as well a dominant one on the web page. Use of image snippets can be very beneficial as understanding an image takes a very short time for a human being. According to a report, a human can comprehend an image in 110 ms or less (Coltheart, 1999) and in this period of time, he/she can only read less than 1 word, or skim 2 words. Another report shows that the average English reader can read about 4.2 words per second, and can glance through roughly 17 words per second (Chapman, 1993). Image snippet detection and presentation with search results had been studied in a few researches (Teevan et al., 2009; Xue et al., 2008), where it was shown that image snippets or internal images are useful for search result navigation and web page re-finding. But there was no indication about how to incorporate this process into an existing search engine considering the scalability issues.

3.4 Visual Tags

Information visualization with tag clouds have become a popular method for visualizing information on the web. Tag clouds represent the terms present in the document where important terms are emphasized using different font sizes, weight or color (Maqbal, Scholer, Thom, & Wu, 2010). Tag clouds can be effective for providing a quick view of the content of the web page. Visual summaries based on the combination of tag clouds and thumbnails are becoming essential parts of search engine interface.

3.5 Snippet Extraction Using VIPS Algorithm

As far as we know, the most related work to our one has been done by Xue, who used the Vision-based Page Segmentation algorithm (VIPS) for image snippet generation (Xue et al., 2008). VIPS algorithm is a popular algorithm for web page segmentation based on the vision of a human and hence useful in many contexts like information extraction, information retrieval and automatic understanding of a web page (Cai, Yu, Wen, & Ma, 2003). VIPS algorithm makes use of the HTML DOM (Document Object Model) tree and extracts suitable blocks from it. After that, it tries to find separators between the extracted blocks. After the processing each block is assigned a Degree of Coherence (DOC) value, which indicates how coherent the content of the block is. VIPS algorithm uses a top-down approach, which is very effective from the perspective of the semantic segmentation of a web page, but not efficient if considered as a part of image snippet generation framework. As VIPS works on an HTML DOM tree in a depth first manner, theoretically its complexity is $O(N+(N+1))$. Here N is the number of nodes in the final tree extracted by the VIPS algorithm. If VIPS is performed on N web pages, then it would lead to a process with quadratic time complexity. Xue proposed to use VIPS algorithm on web pages to find the most coherent block and extracted the image from that block as snippet. We ran the VIPS algorithm for different number of web pages and realized, as the number of web pages increase the time for segmenting the web pages increase and reach to a value which will not be tolerable as searching latency.

4. Proposed Framework

In this section, we describe the theoretical framework for snippet generation and its implementation methodologies. The framework is mainly composed of three components as shown in Figure 1, which are crawler and indexer, searcher and user interface. As our solution is focused to scalability, we also analyze the time complexity of our solution.

4.1 Web Page Indexer

Indexing is a process that extracts the terms or words from a web documents, and tags the corresponding URL of the web page with them. Text search engines index web pages or other textual documents and image search engines index images. The proposed framework combines the functionality of both. Figure 1 shows, how the indexer component is integrated with the complete search framework. Figure 2 shows the components of the indexer in detail. The yellow shaded regions in Figure 2 show our contribution areas inside the indexer. Our proposed indexer contains three core components: parallel indexer, web id generator and web page content indexer.

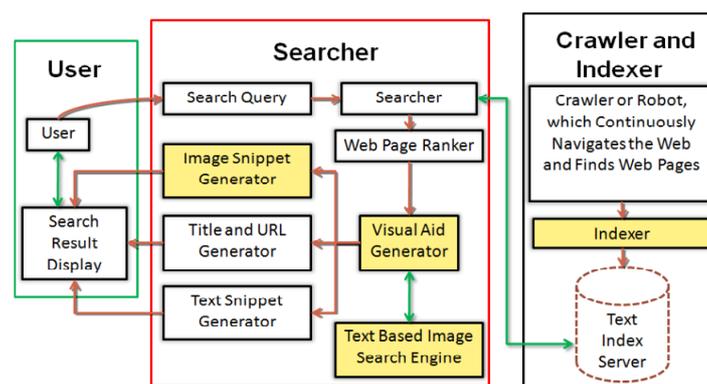


Figure 1. Basic components of the proposed framework

4.2 Parallel Indexer

Text based image search engines perform image indexing, which is actually a variation of text indexing (Kherfi, Ziou, & Bernardi, 2004). Web text search engines index web page URL against the terms found in the web page, whereas, image search engines index image URL and web URL both, against the terms found in the image file name, surrounding text and alternative text. Extracted terms play a crucial role in searching, as these terms are matched against the user query, to find the desired documents or images. Image search engines and text search engines work independently and perform the indexing separately. That means each search engine crawl around the WWW separately, and indexes the documents. There is no explicit link between the both. In our framework, we created an indexer, which performs the indexing of both the contents and images of a web page. We termed this phenomenon as parallel indexing, and the unification of both search engines begins in this process.

4.3 Web ID Generator

The link between both engines is made in our design by a web id generator as shown in Figure 2, which serves both the text and image indexer with a unique identification number. The identification number is stored against the content as well as the images. So, if a web page is given an id number 201, then all the images in it will be tagged with 201. This is the point where, text and image search engines are integrated. The index servers for image and text search engines may be different, but with the help of identification numbers, images for a web page can be found very quickly.

From the concept of distributed search index, where index data is distributed across more than one machines, the web id can be very beneficial from the perspective of snippet image finding. Images containing a range of web id values can be stored in a search server. For example, index content for images containing web id values between 1-1000 can be stored in server number 1, between 1001-2000 can be stored in server number 2 and so on. So, whenever we want to find the index content of all the images that appear on a web page, we can easily find the server to search in constant time. It is even more beneficial in a situation, where we have a list of web id values and we need to find the index content of all the images for each web id. We can easily issue parallel queries to image index servers and obtain better throughput.

4.4 Web Page Content Indexer

The web page textual content indexer was developed as a part of the total search framework. There are different ways for developing a textual content indexer. It involves parsing HTML pages, removal of stop words, taking the words to the root forms and then indexing them. For parsing web pages, we have the HTML content extractor component as shown in Figure 2. For obtaining the terms to be indexed we have the term analysis engine. There can be different types of indexers. We have used the indexing module of Apache Lucene (Foundation, 2011) in our work.

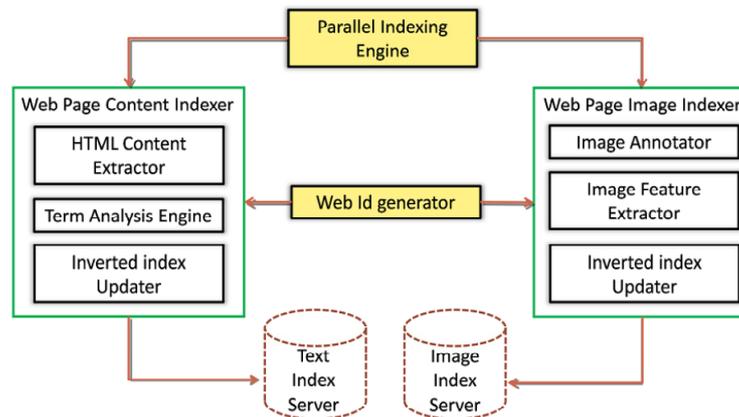


Figure 2. Parallel indexing of web page content and images

Text based image retrieval systems usually perform automatic image annotation from the desired web pages. The “img” element is used to embed an image into an HTML web page. “img” element contains information about the source URL of the image, description (if provided by the developer) and the file name of the embedded image. Using this information along with the surrounding text, annotation of web images can be performed.

Image in a web page has features like image position and area occupied by that image. The position of an image in a web page is a valuable feature, as developers tend to place the important images at the beginning of the web page. The position value is inversely related to the importance of an image. For example, an image visually placed at the first position has the highest importance value. The area occupied by an image is also an indicator of the importance of an image. The more area an image occupies in a web page, the more important it becomes with respect to that web page. The values of image position and area can be calculated by parsing the HTML document corresponding to a web page.

4.5 Searching and Image Snippet Selection

The framework improves the efficiency of the image snippet selection process of a search engine and that in turn helps the users to visualize the search results. The image snippet finding objective is realized by the searcher component of the framework as shown in Figure 1. The yellow shaded regions inside the searcher component in Figure 1 show our contribution areas in the searcher. The detailed searching process is illustrated in Figure 3. When a user places a query q to the system, the system finds the relevant documents that match the query. From Figure 3, we can see that in step 4, $top-k$ web pages with title, URL and web id are retrieved. We actually show a fraction of the retrieved results in the interface. Commercial search engines usually show 10 results in Search Engine Result Page (SERP). In our framework, only first ten pages are shown in the first search result page. Next ten pages are shown in the second result page. So, image snippets for only 10 pages are to be retrieved. As a result, for each search result page 10 *web id* values are extracted from 10 results. After that, for each *web id* a query to the image search index is sent. Using this web id we retrieve all the images in a web page. We also retrieve the normalized area value, image position value and annotated image text. The annotated text is composed of three sections like filename, alternative text and surrounding text. The text annotation process is described in the Section 2.4.

In this way, all the indexed information about the images in a web document is retrieved. After that, a scoring function is used for computing the score of each of the images. The function evaluates the score based on indexed data. The image with highest score is selected as image snippet. The scoring function is as follows:

$$Score(s) = \alpha \times 1 / (\log(P) + 1) + \beta \times S(q, \text{annotated text}) + \gamma \times A \quad (1)$$

In Equation 1,

P = image position value

$S(q, \text{annotated text})$ = similarity score between query and annotated text

A = normalized value of area

α, β, γ are free parameters and $\alpha + \beta + \gamma = 1$.

The value of each parameter can be set heuristically and according to importance. If the value of α is set 0, the position of the image will have no effect in the scoring function. If the value of α is set to 1, and values of all other parameters are set to 0, then only image that comes first while visualizing a web page will be selected as snippet. In this way, importance of each feature can be set by the user of the system by setting the parameters appropriately. The final score of an image will be in the range of [0, 1]. The process of scoring the images and finding the best image has been shown in Algorithm 1.

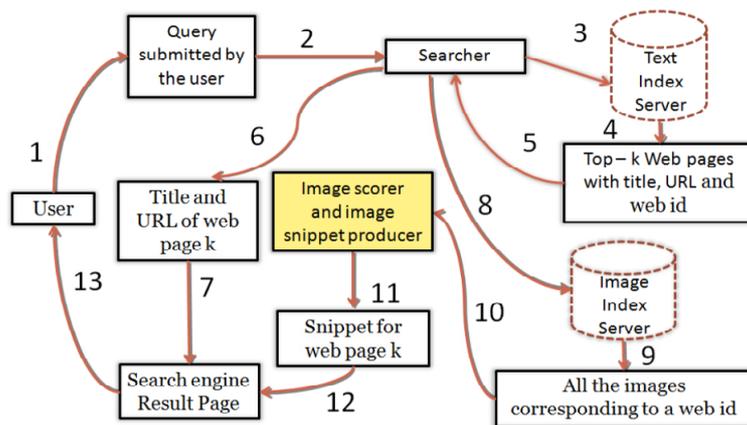


Figure 3. The detailed flow diagram of the proposed searcher

4.6 Similarity Score Calculation between Query and Annotated Text

We annotate an image with the text from the image file name, alternative text and surrounding text as described in section 2.4. For the calculation of similarity score, we took an approach that is simple and has linear time complexity. First, we take the user query q and tokenize it into terms. For each term, we check if the term is found in image file name (f), alternative text (a) and surrounding text (s). We give binary score for match in each field. So, if a term occurs in a field then the score will be one and otherwise zero. For three fields the accumulated score for a term will be in the range of [0, 3]. The accumulated score is then divided by 3. In this way, for all the terms scores are calculated. After that, the score is divided by the number of terms.

The complexity of similarity score calculation for a single image data is:

$$C = \text{Number of terms} \times O(l(f) + l(a) + l(s)) \quad (2)$$

Here, $l(f)$ is the length of the image file name, $l(a)$ is the length of alternative text and $l(s)$ is the length of the surrounding text. This complexity is derived from the fact the finding the occurrence of a query string (key) in a search string is $O(n)$, where n is the length of the search string.

4.7 Complexity of Snippet Generation

In this section, we will show the complexity analysis of the snippet generation process. Suppose, we have n number of web pages to be displayed as search results in a result page. For each of n web pages, first we have to retrieve all the images using a web id. Retrieving with web id takes $O(1)$ time, if we create hash index (with collision handling) on the *web id* values. For each image in a web page we have to calculate the score of the images according to the scoring function defined in section 4.5. From Equation 2, it can be shown that for k images, the complexity C_k of the score calculation function is:

$$C_k = k \times t \times O(l(f) + l(a) + l(s)) \quad (3)$$

In Equation 3, t is the number of terms in the query. $l(f)$ is the length of image file name, $l(a)$ is the length of the annotated text and $l(s)$ is length of surrounding text. The complexity of snippet generation for n web pages using Equation 3 is:

$$n \times C_k = n \times k \times t \times O(l(f) + l(a) + l(s))$$

Usually the value of $n = 10$, as in most of the commercial search engines, we see that 10 results are shown in a search result page. Also, the number of images in a web page is not usually more than 50 and hence, $k \leq 50$ can be considered. The number of query terms in a query is usually less than 3 and hence $t \leq 3$ can be considered. This leads to the overall complexity of $O(l(f) + l(a) + l(s))$. So, If the number of documents m increases, it will not have affect on the snippet generation process. ***Our snippet generation framework is scalable, as it is not affected by the growth of documents in the search engine index.***

Algorithm 1: Find Image Snippet

Input: User query q

Definitions:

- Free Parameters α, β, γ
- P is normalized position value
- A is normalized area
- wid is web page identifier

Begin

Retrieve the set S of most relevant documents for q

Create $D \subseteq S$, where $D = \{\text{documents to be displayed as search result}\}$

For each element in D , extract wid and insert into a list $lwid$

Extract wid and insert into a list $lwid$

End For

For each wid in $lwid$ do

Retrieve index information in list I for all the images with the wid

For each image i do

$l_i.s \leftarrow$ similarity score calculation using q and annotated text

$l_i.score \leftarrow \alpha \times I / (\log(P) + 1) + \beta \times l_i.s + \gamma \times A$

End For

Find l_i where $l_i.score$ is maximum

Get $l_i.url$ and display the image with the corresponding URL as snippet image

End For

End

5. Experimental Results and Findings

We have performed different types of experimentation and measured the performance of our snippet generation process. We have developed our experimental framework, which is available at www.icekite.com. We also have faced some implementation challenges. In this section, we address these facts.

5.1 Experimental Setup

In order to experiment with our theoretical assumptions and logics, we have developed our experimental framework. We have developed a crawler, indexer and searcher module. Using our crawler, we have crawled Wikipedia web pages and indexed them. For parsing the web pages, we have used Jericho HTML Parser (Jericho, 2011). For developing the indexer and searcher, we have used the Apache Lucene Framework (Foundation, 2011). We have used JSP (Java Server Page) and HTML as our front-end technology. We have performed all the necessary steps as mentioned in Section 2.

5.2 Performance Evaluation

5.2.1 Performance of Snippet Generation

We have crawled and indexed 10000 web pages from Wikipedia and tested the efficiency of our image snippet generation algorithm in finding the snippets. It can be seen from Figure 4 that for most of the queries the snippet extraction time is around 1 or 2 ms. By examining Figure 5, it can be seen that our image snippet generation process takes almost the same time as the search result generation process. So, it can be revealed, that there is low latency in generating the snippets.

The last and the most important part of our findings is the comparison of our algorithm with an existing one proposed by (Xue et al., 2008). From the Figure 6, we can see that, as the number of documents increases, the snippet generation process slows down according to Xue’s algorithm. Our algorithm can produce result in less than 0.1 seconds even for 15 documents. Search engines typically show only 10 results in each result page and hence generate image snippet 10 results. We have shown that our algorithm can perform well even for 15 documents.

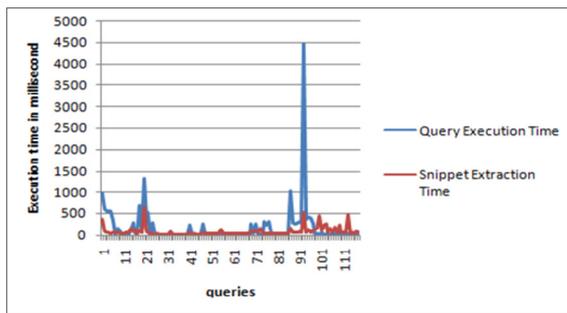


Figure 4. Comparison of query execution time and snippet generation time

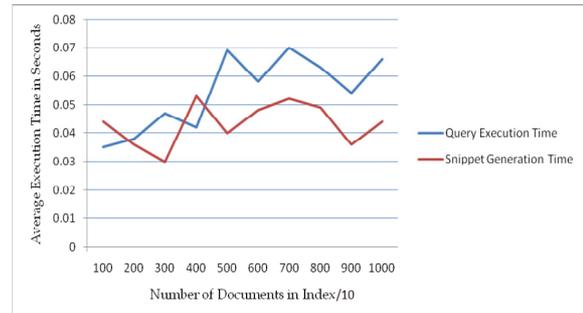


Figure 5. Efficiency of our image snippet generation algorithm

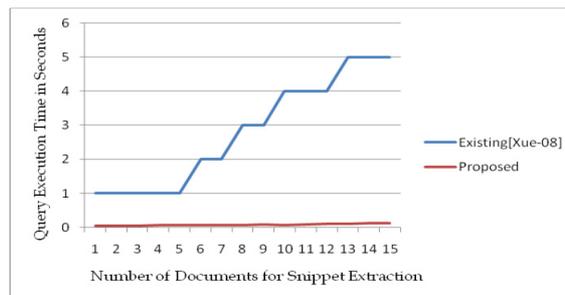


Figure 6. Comparison of our algorithm with an existing one

5.2.2 Output of Our System

We have shown the output of our system using Figure 7 to Figure 10. As our snippet scoring function given in Section 4, we can assign importance on different image and query based properties by setting the values of parameters. Using Figure 7 to Figure 10, we have shown the output of our system using full importance on different parameters. For example, full importance on area indicates that only the images with large area are selected as image snippets. No other parameters are considered. We have also shown the output of our system using same importance on each of the features or parameters using Figure 10. In every case, our query was “**Intel Corporation**”. With careful observation, it can be easily seen that the products offered by Intel Corporation come as image snippets.

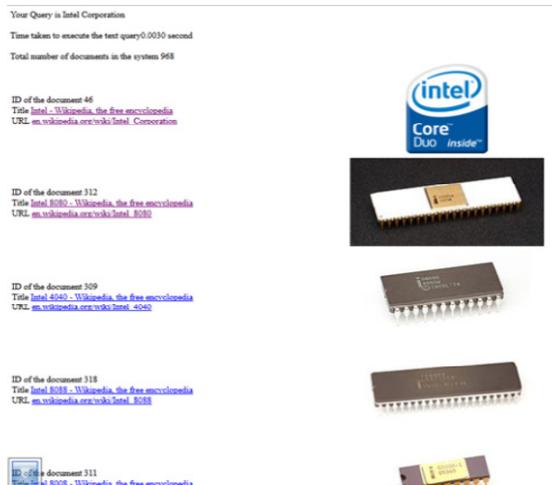


Figure 7. Output of our system with full importance to text matching

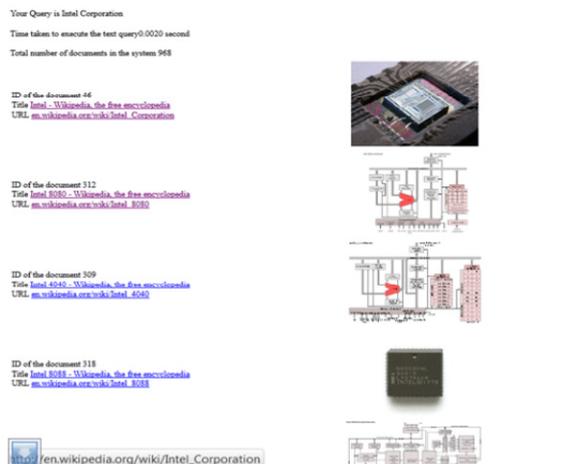


Figure 8. Output of our system with full importance on image position

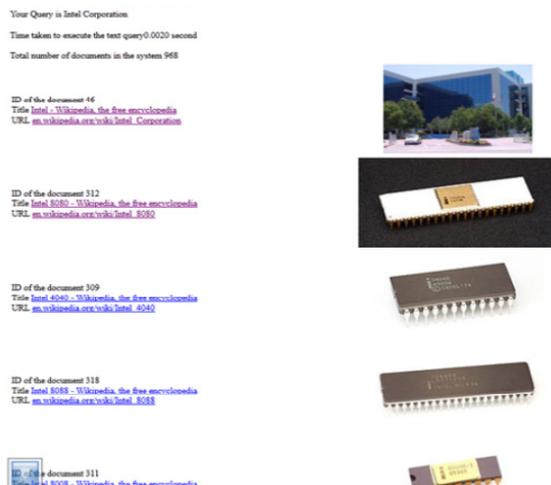


Figure 9. Output of our system with full importance on area covered by an image

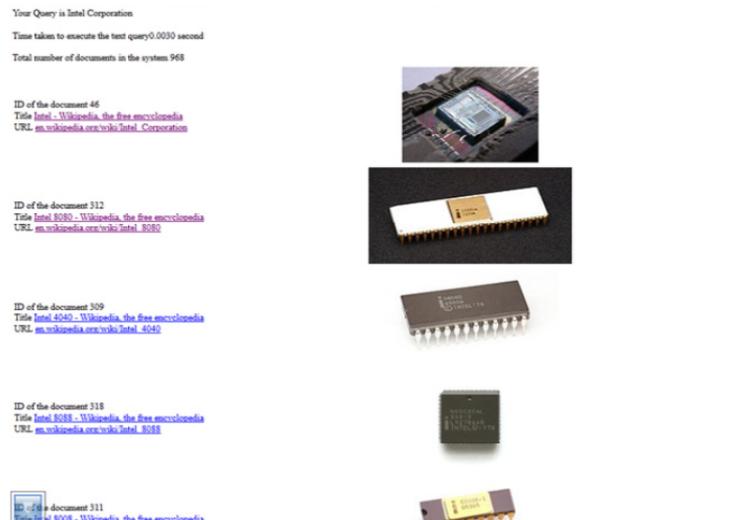


Figure 10. Output of our system with same importance on each field

6. Conclusions and Future Works

In this work, we have proposed an image snippet generation framework by enhancing the currently available ones and showed the computational complexity associated with the operation of the framework. The proposed framework provides an image for each search result that matches user query intention and for this purpose we have integrated text and image search engines. Our method takes less time and can be very easily incorporated with the currently available search engines. The framework is exposed to minimum amount of load, because of the use of a simple scoring or ranking function. The technology of our search framework can be used from small scale enterprise search engines to large scale hypertext search engines. We have developed a prototype of the framework (available to use at www.icekite.com) using an enterprise search framework Apache Lucene. As we have obtained better results, we can conclude that enterprise search engines can certainly incorporate the idea of our framework. For large scale web search engines, we have to go for practical implementation and analysis. But according to the theoretical perspectives, we can conclude that the process can easily be incorporated with a large scale search engine.

References

- Aula, A., Khan, R. M., Guan, Z., Fontes, P., & Hong, P. (2010). A comparison of visual and textual page previews in judging the helpfulness of web pages. In *Proceedings of the 19th international conference on World Wide Web* (pp. 51-60). New York, NY, USA: ACM. <http://dx.doi.org/10.1145/1772690.1772697>
- Cai, D., Yu, S., Wen, J. R., & Ma, W. Y. (2003). Extracting content structure for web pages based on visual representation. In *Fifth Asia Pacific web conference (apweb2003)*.
- Chapman, A. (1993). *Making sense: teaching critical reading across the curriculum*. College Board, New York, NY.
- Cockburn, A., Gutwin, C., & Alexander, J. (2006). Faster document navigation with space-filling thumbnails. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 1-10). New York, NY, USA: ACM. <http://dx.doi.org/10.1145/1124772.1124774>
- Coltheart, V. (1999). *Fleeting memories: cognition of brief visual stimuli*. MIT Press.
- Dork, M., Carpendale, S., & Williamson, C. (2009). Visual Web Exploration: Beyond Ranked Snippets and Thumbnails. *Vis Week*.
- Dziadosz, S., & Chandrasekar, R. (2002). Do thumbnail previews help users make better relevance decisions about web search results? In *Proceedings of the 25th annual international acm sigir conference on research and development in information retrieval* (pp. 365-366). New York, NY, USA: ACM. <http://dx.doi.org/10.1145/564376.564446>
- Foundation, A. S. (2011). *Apache lucene - apache lucene core*. Retrieved from <http://lucene.apache.org/core/>

- Garcia, D. E. (2011). *Cosine similarity and term weight tutorial*. Retrieved from <http://www.miislita.com/information-retrieval-tutorial/cosine-similarity-tutorial.html>
- Jericho, M. (2011). *Jericho html parser*. Retrieved from <http://sourceforge.net/projects/jerichohtml/>
- Keegan, S. N. (2007). Importance of visual images in lectures: Case study on tourism management students. *Journal of Hospitality, Leisure, Sports and Tourism Education*, 6, 58-65. <http://dx.doi.org/10.3794/johlste.61.147>
- Kherfi, M. L., Ziou, D., & Bernardi, A. (2004). Image retrieval from the World Wide Web: Issues, techniques, and systems. *ACM Comput. Surv.*, 36(1), 35-67. <http://dx.doi.org/10.1145/1013208.1013210>
- Li, Q., & Chen, Y. P. (2010, January). Personalized text snippet extraction using statistical language models. *Pattern Recogn.*, 43, 378-386. <http://dx.doi.org/10.1016/j.patcog.2009.06.003>
- Maqbali, H. A., Scholer, F., Thom, J. A., & Wu, M. (2010). Evaluating the effectiveness of visual summaries for web search. In *Proceedings of the 15th Australasian document computing symposium*.
- Robertson, S. (2004). Understanding inverse document frequency: On theoretical arguments for idf. *Journal of Documentation*, 60(5), 503-520. <http://dx.doi.org/10.1108/00220410410560582>
- Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Commun. ACM*, 18, 613-620. <http://dx.doi.org/10.1145/361219.361220>
- Teevan, J., Cutrell, E., Fisher, D., Drucker, S. M., Ramos, G., Andre, P., & Hu, C. (2009). Visual snippets: summarizing web pages for search and revisitation. In *Proceedings of the 27th international conference on human factors in computing systems* (pp. 2023-2032). New York, NY, USA: ACM. <http://dx.doi.org/10.1145/1518701.1519008>
- Turney, P. D., & Pantel, P. (2010). From frequency to meaning: vector space models of semantics. *J. Artif. Int. Res.*, 37, 141-188.
- Wong, S. K. M., & Raghavan, V. V. (1984). Vector space model of information retrieval: a reevaluation. In *Proceedings of the 7th annual international acm sigir conference on research and development in information retrieval* (pp. 167-185). Swinton, UK, UK: British Computer Society.
- Wu, H. C., Luk, R. W. P., Wong, K. F., & Kwok, K. L. (2008, juin). Interpreting tf-idf term weights as making relevance decisions. *ACM Trans. Inf. Syst.*, 26(3), 13:1-13:37.
- Xue, X. B., Zhou, Z. H., & Zhang, Z. M. (2008). Improving web search using image snippets. *ACM Trans. Internet Technol.*, 8, 21:1-21:28.