# ICICLE: A Semantic-based Retrieval System for WWW Images

Beng Chin Ooi*  Heng Tao Shen  Kian Lee Tan

School of Computing, National University of Singapore, 3 Science Drive 2, Singapore 117543. Phone: 65-68746465, Email: {ooibc, shenht, tankl}@comp.nus.edu.sg.

---

## ABSTRACT

Images are increasingly being embedded in HTML documents on the WWW. Such documents over the WWW essentially provide a rich source of image collection from which users can query. Interestingly, the semantics of these images are typically described by their surrounding text. Unfortunately, most WWW image search engines fail to exploit these image semantics and give rise to poor recall and precision performance. In this paper, we present *ICICLE* (Image ChainNet and Incremental CLustering Engine), a prototype system that we have developed to effectively and efficiently retrieve WWW image. ICICLE has two distinguishing features. First, it employs a novel image representation model called *Weight ChainNet* to capture the semantics of the image content. Weight ChainNet is based on lexical chain that represents the semantics of an image from its nearby text. A new formula, called *list space model,* for computing semantic similarities is also introduced. With Weight ChainNet, ICICLE is able to effectively retrieve images based on their image semantics. To further improve the retrieval effectiveness, we also propose two relevance feedback mechanisms. Second, ICICLE employs an incremental clustering mechanism, *ICC* (Incremental Clustering on ChainNet) that narrows the search space of the retrieval process to the relevant partitions. In ICC, images with similar semantics are clustered into the same partition. Moreover, ICC facilitates incremental updates and can adaptively adjust the number of clusters and cluster sizes. We conducted an extensive performance study to evaluate ICICLE. Our results show that ICICLE is not only effective but also efficient. The Weight ChainNet used in ICICLE is able to provide better recall and precision compared to existing techniques. The relevant feedback mechanisms can further improve the retrieval effectiveness. Moreover, our results show that ICC facilitates speedy retrieval of images without sacrificing on the quality of the relevance of the images.

**EDICS:**

**7-WEBS;**

**4-ANSY;**

**4-KEEP.**

## 1. INTRODUCTION

With the increase in Internet bandwidth and CPU processing speed, the use of images in WWW pages has become very prevalent. Images are used to enhance description of content, to capture attention of readers and to reduce the textual content of a WWW page. An image is worth 1000 words. Images have become an indispensable component of WWW pages today. This pool of WWW images becomes a very rich source from which users can obtain interesting images. However, as the web crawler keeps crawling, the growing number of images embedded in WWW pages makes the WWW a gigantic image database. To retrieve relevant images from this collection poses two challenges to the research community. First, more semantic-based effective (measured in terms of recall and precision) method should be designed. Second, the exponential growth rate would eventually, if not already, render any existing techniques ineffective and inefficient.

In this paper, we present our solutions to address the issues of effectiveness and efficiency. To tackle the effectiveness problem, we employ a novel scheme to capture the semantics of an image within a HTML document. This is based on the observation that an image in a Web page is typically s*emantically* related to its surrounding texts, with the exception of *functional* images (such as *new* symbol and *under construction* symbol). These surrounding texts are used to illustrate some particular semantics of the image content, i.e. what objects are in the image, what is happening and where the place is. In particular, in a HTML document, certain components are expected to provide more semantic information than other portion of the text. These include the caption of the image, its title and the title of the document. We propose a novel image representation model called *weight ChainNet*. Weight ChainNet is based on *lexical chain* obtained from an image's nearby text. A new formula, called *list space model*, for computing semantic similarities is also introduced. To further improve the retrieval effectiveness, we also propose two relevance feedback mechanisms.

To speed up the searching process, two research areas can be considered: indexing and clustering. For those earlier mentioned searching systems, various indexing methods can be applied to indexing image text (words) and visual (high dimensional points) features, like converted file, R-tree, HG-tree, the signature file and hashing technique. However, they are not applicable to our new WWW image representation, whose basic unit is a Lexical Chain with various dimensions, instead of word. To overcome the efficiency problem, we propose that the database be split into multiple smaller partitions based on the semantic representation model mentioned above. To this end, we propose a novel clustering scheme, called ICC (Incremental Clustering on ChainNet) that clusters images with similar semantics into the same partition.

ICC facilitates incremental updates. In this way, the newly added data are inserted into the relevant partitions or a “noise” partition. In addition, ICC can dynamically adjust the number of partitions and the partition size by splitting larger partitions or merging small partitions. ICC is supported by two important mechanisms. First, it employs a hierarchical tree structure, Hierarchical-ChainNet Summarization Tree (denoted HC-ST), whose leaf nodes represent summary information of clusters (one leaf node per cluster), and whose internal nodes contain summary data on their children nodes. Second, the summary data at internal nodes are obtained using a two-step novel scheme, called Vertical and Pyramidal Summarization Tree (VP-ST). The first step generates the summary of each cluster (or node), while the second step further combines the summary data into a more concise form. Each summary information is also represented in the form of a summary ChainNet. Given a query image, we first locate the partitions that contain images that are relevant to it. This is done by comparing its ChainNet with that of the summary ChainNet at internal nodes. Finally, the relevant partitions are examined.

We implemented a prototype WWW image retrieval system, called ICICLE (Image ChainNet & Incremental Clustering Engine) that employed the proposed mechanisms. We evaluated the system on a collection of 10,000 images obtained from documents identified by more than 2,000 URLs. Our results show that ICICLE is both effective and efficient compared to existing techniques. In particular, the Weight ChainNet model outperforms known techniques [1] in terms of recall and precision. Moreover, the relevant feedback mechanisms can lead to significantly better retrieval effectiveness. In addition, ICC can also lead to faster retrieval time without sacrificing on the quality of the images retrieved.

A preliminary version of this paper appeared in [33]. There, we addressed the issue of representing the semantics of a WWW image using the surrounding text. Here, in this paper, we extended the work in several ways. First, we examine the issue of efficiency using an incremental clustering scheme. Second, we present the prototype system ICICLE. Third, we report on more experimental results.

The rest is organized as follows. In the next section, we briefly review some related work in WWW image retrieval systems. In Sections 3, we present our models for retrieval effectiveness, including image semantic representation model, the similarity measure, and the relevance feedback approaches to refine queries for further retrieval. In section 4, we present our efficiency methodologies, including incremental clustering algorithm and the summarization technique. In Section 5, we present the prototype system, ICICLE, and report the results of a performance study conducted on the models and algorithms, and finally, we conclude in Section 6 with directions for further work.

## 2. RELATED WORKS

With the increasing need in WWW image retrieval, many recent WWW image search engines have been developed in last decade. However, most of the existing image retrieval systems cannot adequately address the issue of effectiveness and efficiency. Text-based systems use keywords or free text description of images supplied by the authors as the basis for retrieval. These systems can be adopted for WWW images since the textual content of the HTML page in which the image is embedded provides the free text description. However, the entirety of the textual content does not represent the semantics of the image adequately for them to be useful in retrieving the images. In other words, while the textual content may contain information that captures the semantics of the embedded image, it also contains other descriptions that are not relevant to the image. These "noises" may lead to poor retrieval performance if query contains some of these noises. Many first generations of WWW text engines, like Lycos and Alta Vista, extracted keywords using standard algorithm that consider keyword placement, frequency, etc. They do not require solving the image understanding problem, which results in poor performance in image retrieval.

Typically, [1] conducted experiments on using semantic distances between words in image caption retrieval. They calculated word similarity between related words in a thesaurus. Similarity between words is used to identify if two images are relevant. Meanwhile, only image caption are involved for identification. More recently, [19] also extended [9] to work with WWW-based image collections. In [19], an image's content is given by the combined content of the text *nodes*. An image's set of text nodes include textural content (e.g., caption) obtained from the document in which it is embedded, as well as those obtained from its neighboring pages (those pages that are reached by a single hyperlink from the embedded page). This model was further extended to take into account not only the textual content of the immediate neighbors of an image, but also all nodes that can be reached from the image by following at most two hyperlinks (a *two-step link*), thus considering more information about an image node. However, there are no explicit image/query semantics considered. The inner semantic relationship within a text node was lost based on this model. Moreover, while keeping more information is desirable, the approach extracted too much unrelated information, as relatively low precision can indicate. For example, an image’s own caption usually describes its content, but its neighboring pages' image captions do not reflect the same content. In addition, the similarity measure did not take into account any semantic structure. Such a similarity measure may not be good enough to show the *real* semantic similarity between an image and a query.  Relevance feedback (RF) is a very important way to improve the accuracy. The system refines the query by using feedback information from users to improve subsequent retrieval.  The use of relevance feedback using multiple attributes of color has been investigated in [7]. Their results showed significant improvement in retrieval effectiveness by applying RF mechanisms.

On the other hand, content-based image retrieval systems [7,17,27,35,37,38] capture the visual content of an image (such as color, texture and shape) as its semantics and use these features as the basis for similarity matching. Unfortunately, retrieval by content is still far from perfect and their results are not reliable. First, their effectiveness depends on how precise the user specifies the query. Second, they are at very low performance levels as they cannot capture the more useful image semantics, like object, event, and relationship.  Finally, they do not scale well. For the WWW image database, content-based image retrieval systems are not reliable, since low-level visual features cannot represent the high-level semantics of WWW images.

A combination of textual and visual features has been used in integrated image searching [4,12,16, 22,28,31,35,38,41,43,44]. Works in above engines uses image features and associated text for automatic indexing of images. However, the key issue is how to obtain the high-level semantic features. Unfortunately, the image’s surrounding descriptive texts are not well identified, but used to extract keywords for words matching purpose only. The internal semantic relationships among words are not remained anymore, which leads to poor precision in text-based searching component. Thus the overall performance is not very satisfied, as content-based systems are still in performance low level.

In terms of efficiency, most of the existing works employ access methods such as R-trees and its variants [29,39], HG-tree, the signature file and hashing technique [16] to speed up the retrieval process. While efficient for small or moderate size databases, these methods are not expected to scale well with very large image collections in high dimensions like images provided by the WWW. More importantly, unlike earlier WWW image representation, in our image representation model, the basic unit is LC, not single word. Indexing methods are not practical to be applied since LCs are in various dimensions, which is the number of words. Another area to improve the efficiency of the system is to cluster the image collection into partitions. Most of the existing clustering schemes, however, are designed for static databases. Existing static clustering schemes [2,3,6,23,20,26,42] that have to perform the clustering from scratch should there be any new data to be added. Clearly, because the WWW image database keeps updating over time, they are not suitable for such kind of database. Few incremental clustering methods [15,21] have been also proposed. However, they impose a fixed number of clusters as a constraint on the solution.

## 3. A SEMANTIC MODEL FOR WWW IMAGES

Two key issues must be addressed in designing an image retrieval system to support WWW images:

- Determine a representation for a WWW image and the query semantics.
- Determine a similarity measure between an image and a query based on their representations.

To further improve the precision, RF is an important tool.

### 3.1 Image Representation Model

To understand the relationship between an image embedded in a HTML document and its surrounding text, we conducted a preliminary study on a collection of images obtained from HTML documents. We randomly choose 500 images and to see if the embedded image is well described by its surrounding text. Based on our findings, out of 500 images, 94% can be semantically identified by the combination of the following four textual parts. These are

- Image title. Image file title (simply image title) is a single word that basically indicates the main object that the image is concerned with.
- Image ALT (alternate text). The image ALT tag in HTML document is a phrase that usually represents an abstract of the image semantics.
- Image caption. The image caption usually provides the most semantics about an image. It is the image's surrounding text in the HTML document. It can range from one sentence to a paragraph of text that contains many sentences.
- Page title. Since images are used for enhancing the Web page's content, page title is most probably related to the image's semantics. It is usually a short sentence that summarizes the Web page's content.

From our founding, we can see that at least 6% of images in WWW may be missed in text-based retrieval because the text is not informative. Content-based searching can be used to reduce this ratio. However, this is not this article's focus. We argue that these four parts are reliable to represent image semantic since 94% images can be well described by them. There are also some other parts which may provide some information about the image, such as other HTML meta data. However, they contain too much unrelated information. We have also excluded the textual content of the whole HTML document as part of the image's semantics for the same reason, i.e., that some information may be completely unrelated to the image content, and indexing the whole HTML document for each image in a very large database is not expected to provide an efficient solution. Therefore, we just use these four parts to represent image content. We note that all these four parts -- image title, image ALT, page title and image caption -- can be automatically extracted from the HTML document based on hypertext structures.

Based on the above findings, we propose representing the image semantics using the Weight ChainNet model that is based on the concept of *lexical chain* [25]. Figure 1 illustrates an example. In [25], a lexical chain (LC) was defined as a sequence of semantically related words in a text. Inspired from that, here we treat a LC as a sentence in image's surrounding text. In grammar, a sentence can be regarded as a sequence of semantically related words. As an image title is just a single word, we say it's a trivial lexical chain - *Title Lexical Chain (TLC).* The text obtained from the ALT tag is referred to as the *Alt Lexical Chain (ALC).* The page title is represented as a LC too - *Page Lexical Chain (PLC)*. Finally, since a caption comprises multiple sentences, we represent it as three types of lexical chains. Type one is called *sentence lexical chain (SLC)*, which represents one single sentence in an image caption. In Figure 1, each sentence is shown as one column in the caption component, i.e., each column is a SLC. Type two is called *reconstructed sentence lexical chain (RSLC)*, and it represents one new sentence reconstructed from related sentences. Two sentences are *related* if both share one or more words. One common word in two SLCs splits each SLC into two. Based on the first common word, the second SLC's second half is connected to the first SLC's first half to form a RSLC. In Figure 1, a RSLC exists if there is an arrow from one column to another column. The last type is called *caption lexical chain (CLC)*, which represents the whole image caption. A CLC is formed by connecting SLC one after another. In Figure 1, the connections are made by dotted arrows. To illustrate, the followings are some examples from Figure 1.

SLC (1→ 2→ 3→ 4→ 5), RSLC (1→ 2→ 8→ 9), CLC (1→ 2→…→ 13→ 14).

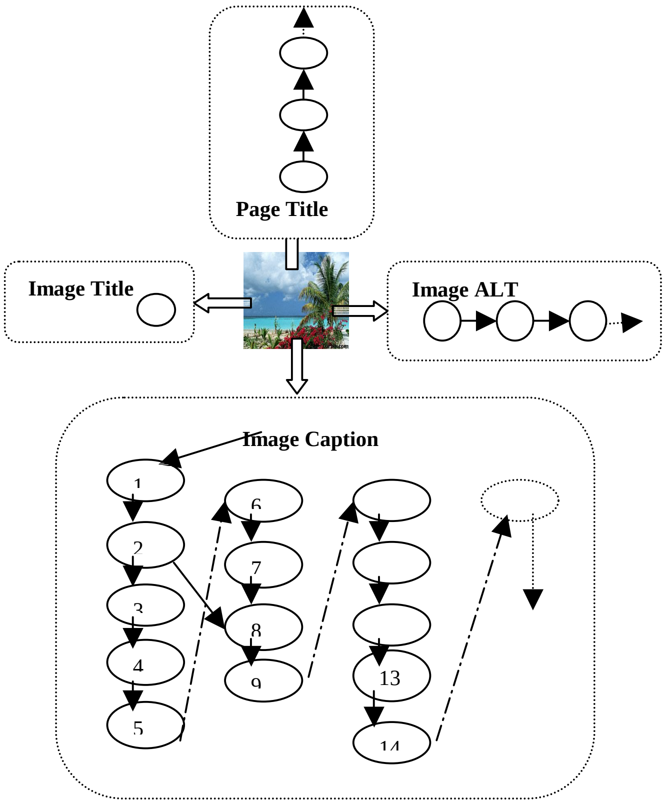



Figure 1: Image Semantic Representation - *Weight ChaiNet*

The ChainNet model is built by these 6 types of lexical chains. Each chain captures a portion of the semantic structure of the image. A TLC indicates the main subject of an image. An ALC provides short description about an image. A PLC shows part of its content. An SLC captures the semantics of a single sentence in the image caption. An RSLC captures related sentences' semantics, and a CLC keeps the image's overall semantics. That's why we call it ChainNet, which is basically made of a chain of LCs. A real example can be found in figure 4 and corresponding table 1.

However, the ChainNet treats each type of LC as of equal importance, Now, simply representing an image in this way without capturing the relative importance of the various components is not expected to lead to good performance. For example, the image title, ALT, page title and image caption play different roles in representing an image's semantics. The reason we have divided the entire image caption into three types of lexical chains is that we want to differentiate the importance of each type of sentences due to their positions and inner relationship within an image caption. The three types of lexical chains in an image caption are not equally important. The importance order from high to low is expected to be like this: SLC > RSLC > CLC. For instance, if all the same words in a query appear in an SLC, an RSLC and a CLC respectively, the SLC possesses the most semantic meanings among the three, followed by RSLC and finally CLC. For example, if a query matches an SLC in the first image, only matches an RSLC in the second image, and only matches a CLC in the third image, it's most likely the case that the first image is most relevant to the query, followed by the second image and then the third image, because an SLC is more semantically structured than an RSLC, which is more semantically structured than a CLC.

To capture the relative importance of the various types of LCs, we assign weights to the various LCs such that LCs that are deem to be more representative of the image content are assigned larger weight values. We shall see how these weights come into play in the similarity measure to be discussed in the next section. We note that for the caption, one word in the caption may have up to three different weights with respect to the lexical chains it belongs to. Of course, each word has at least two weights: a SLC weight and a CLC weight. If the word belongs to one RSLC, it will have three weights. One image caption may have several SLCs and several RSLCs, but only one CLC. If only one SLC exists, CLC and SLC are identical. The different weight for each type of LC can be confirmed from figure 11 in experiments section.

Thus, the resultant Weight ChainNet model can be formally defined as: a network that is composed of the collection of the different types of lexical chains with different weights. It uses a well-structured notion of image's content to capture the semantic relationship between an image and its nearby text. Such a model can be seen as a semantic representation of the content of an image. This model has the properties of exactness, since it captures an image's essential semantic meanings by an image title, ALT, page title and caption. The content can be automatically extracted. Finally, similarity matching is computationally inexpensive using the proposed *list space model* that we shall introduce in the next section. For a user query, it's usually a free sentence that describes the image content. Naturally, we represent it as a *Query Lexical Chain - QLC.*

## 3.2 Semantic Measure Model

In this subsection, we will present our similarity measure model between two lexical chains, and between an image and a query respectively.

### 3.2.1 Similarity between two Lexical Chains

We have presented the model for representing image/query semantics. To calculate the semantic similarity between a query and an image, we start from determining the similarity between two basic components in an image ChainNet - LC. In our implementation, we store terms of each LC as a list. All lists belonging to an image are connected to the image root as shown by the ChainNet model (see Figure 1). We propose a *list space formula* to compute the similarity between two LCs as follows:

$$Similarity_{list1,list2} \equiv \frac{\sum_{i=0}^{list1.size()} \sum_{j=0}^{list2.size()} e_i.weight * e_j.weight}{\sqrt{list1.size()} * \sqrt{list2.size()}} * MatchScale$$

where $e_i$ and $e_j$ are ith and jth words in list 1 and list 2 respectively, and the $e_i$.weight is the weight of LC that $e_i$ belongs to. Dot product is only applied if two words are matched. Two words are *matched* if they are the same word. We note that we have removed stop words and performed stemming from the various LCs.

In the formula, one important parameter is considered: *MatchScale*. Match scale is defined as the closeness of two lists from the view of match order. For example, one LC is " *US president Clinton and wife visited China in 1997*", and the other one is: " *China president Jiang Zemin welcomed Clinton and wife in Tian'an square*". For these two LCs, there are four matching words. For the first LC, the matched words are in order of "*president Clinton wife China"*, and in the other, they are "*china president Clinton wife"*. We treat each one as a child LC of its original LC. Therefore, the orders of matched words in the two original LCs are not the same. Obviously, the closer the matched order of two children LCs are, the closer the semantics of the original two LCs are. Inspired from the formula for the *angle between two nonzero vectors in 2d-space*, we define the match scale as below:

$$MatchScale_{v1,v2} \equiv \frac{v1 \bullet v2}{\| v1 \| * \| v2 \|} \qquad \text{where} \qquad v1 \bullet v2 \equiv \sum_{i=1}^{v1.size()} v1_i * v2_j$$

where v1 and v2 represent the child LC of the first and second original LCs respectively. The element value is the position in their respective LC. But the *dot product* between two LCs is redefined above, where *v*2*j* is the matched word in *v*2 for *v*1*i* in *v*1. As mentioned, two words are matched as long as they are the same. The above measure determines the similarity between two LCs. However, the two LCs may not be semantically related. For example, consider the query "Singapore Map". An image about Singapore Food, say I1, that contains several occurrences of "Singapore" in CLC may result in a high similarity value even though the images are not semantically related. On the other hand, another image about Singapore Map, say I2, contains only one occurrence of "Singapore" in CLC may result in a lower similarity value despite the fact that it is a desired image. To ensure that two LCs are *semantically related*, we need another parameter called: *Match Level*. Match

Level is the number of the *distinct* matched words by a LC and a QLC, denoted as: LCMatchLevel (LC, QLC). The *match level threshold* is the minimum match level for a LC to keep its original semantics. We say one LC is semantically related to a QLC, if and only if the LC's match level is equal to or greater than QLC's match level threshold. Therefore, in our semantic measure model, *semantic similarity* for a LC with respect to a QLC is indicated by the *similarity* calculated by list space formula in its *match level*. The match level determines *if* the LC is semantically related to the QLC. And the similarity calculated by list space model shows how *well* it is semantically related to the QLC.

### 3.2.2 Similarity between ChainNet and QLC

Now it is time to calculate the semantic similarity between an image and a query. From the discussion above, we know that an image is represented by a Weight ChainNet, and a query is in the form of a lexical chain. To calculate their similarity, we use the following formula:

$$imageSimilarity_{image,query} = S(TLC, QLC) + S(ACL, QLC) + S(PLC, QLC) + \sum_{i=1}^{SLC.number} S(SLC_i, QLC) + \sum_{i=1}^{RSLC.number} S(RSLC_i, QLC) + S(CLC, QLC)$$

where S is the similarity between two LCs. The image match level is defined as:

ImageMatchLevel (ChainNet, QLC) =MAX ( TLC.weight * LCMatchLevel( TLC, QLC), ALC.weight * LCMatchLevel( ALC, QLC), PLC.weight * LCMatchLevel( PLC, QLC), SLC.weight * LCMatchLevel( SLC, QLC), RSLC.weight * LCMatchLevel( RSLC, QLC), CLC.weight * LCMatchLevel( CLC, QLC) )

We say one image is semantically related to a query if and only if its match level is equal to or greater than the query's match level threshold. It has the similarity calculated by the above formula with the query in its match level.

## 3.3 Relevance Feedback

Because of the large image collection and the impreciseness of a query, it is important to provide mechanisms to help users in specifying their queries more accurately. One such mechanism is to exploit feedback from users based on resultant images returned from the initial query. By allowing users to indicate the relevant (and irrelevant) images, the original query can be refined to further improve the retrieval effectiveness. For this purpose, we develop two techniques: *semantic accumulation* and *semantic integration and differentiation*.

### 3.3.1 Semantic Accumulation

The first method, called *semantic accumulation*, allows the user to pick the most relevant image (from the user's subjective judgment) from the result of previous retrieval as the feedback image. The method accumulates all the previous feedback images' semantics to construct a new query for the next retrieval. The resultant query is represented as a kind of ChainNet called Weight F/Q ChainNet (Feedback/Query ChainNet) since it is constructed by query and the feedback image's ChainNet. This kind of new query is represented in Figure 2. Obviously, the combination of every entire ChainNet from each previous feedback images is tedious if the user searches again and again. More seriously, more noise will be added into the new query. Therefore, rather than a whole image ChainNet, we use just one single lexical chain which is most semantically related to the original query in the previous feedback image's ChainNet. This is calculated by the list space model. The steps for this method are:

1. Perform search using the F/Q Weight ChainNet (or Weight ChainNet for first attempt).
2. User selects the current feedback image.
3. Construct the feedback image's Weight ChainNet.
4. Extract the closest lexical chain to the original query from the feedback image.
5. Use the QLC and the weight ChainNet to construct F/Q ChainNet.
6. Use extracted LC and old QLC to construct new QLC by adding together from head to tail.
7. Go to step 1.

In this algorithm, the semantic is accumulated by adding one most related LC from every previous ChainNet to QLC to form a new QLC. Therefore, the QLC carries richer and richer semantics as users provide more feedback.

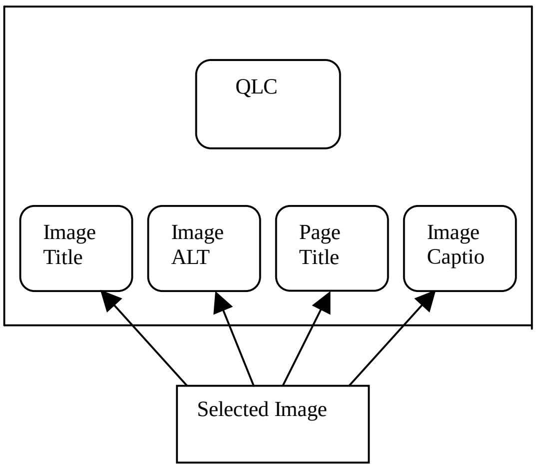



Figure2: F/Q ChainNet in Semantic Accumulation.

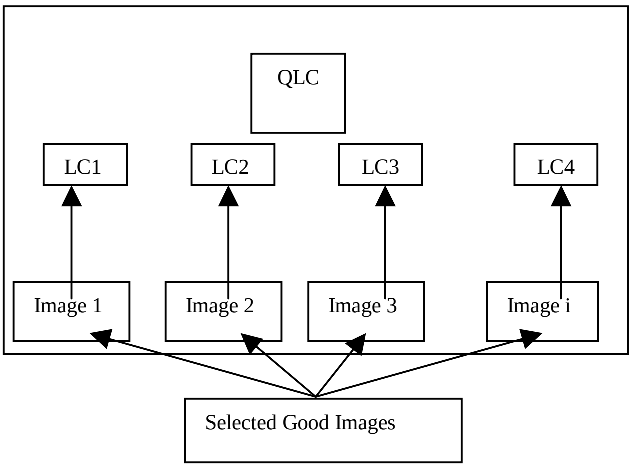



Figure3: F/Q ChainNet in Semantic Integration and Differentiation.

### 3.3.2 Semantic Integration and Differentiation

In the semantic accumulation feedback approach, users can only select one image at a time as the feedback information. To save time and to filter more unrelated images, we introduce another technique: semantic integration and differentiation. In this method, users can select several relevant and irrelevant images simultaneously. By relevant, we mean images that are semantically related to the query as judged by the user and hence should be retrieved. On the other hand irrelevant images are those that the user considers to be unrelated and should not have been retrieved. The system *integrates* the related semantics obtained from the relevant feedback images to construct a new query for the next try. After that, the system combines the semantics from irrelevant images to *differentiate* the irrelevant images from the returned results. The new query is also represented by a F/Q Weight ChainNet as shown in Figure 3.

The steps for this method are:

1. User selects a number of relevant and irrelevant images.
2. Extract the most semantically related LC from each relevant image's ChainNet to form a new F/Q ChainNet with QLC as a new query.
3. Extract the most unrelated LC from each irrelevant image to form a ChainNet for bad images
4. Submit the query
5. From each returned image, remove it from results if it's more related to bad images.
6. Go to step 1

The semantic similarity formula between two ChainNets can be easily extended from the formula for measuring the similarity between an image and a query.

## 4. INCREMENTAL CLUSTERING OF CHAINNET

In this section, we present our approach to speed up the retrieval process. As current indexing methods, like converted file, are all based on analysis of words (mostly nouns and verbs), they are not applicable to ChainNet, where the basic unit is LC and the semantic relationship between LCs should be kept. So as to the current incremental clustering techniques. Here our basic idea is to partition the image collection so that only the relevant partitions need to be examined during the retrieval process. Instead of predetermining the partitions (e.g., fixing the number of partitions and the types of images that belong to the partitions), we propose using a clustering strategy based on ChainNet. Moreover, our clustering strategy can dynamically adapts the number of partitions and partition sizes. We shall use the example shown in Figure 4 as our running example in our discussion. Its corresponding Weight ChainNet representation is shown in Table 1.

### 4.1 Incremental Clustering Algorithm

We define the *incremental clustering* problem as follows: for a continuously growing data sets, maintain a collection of reasonable number of clusters and cluster size corresponding to data size. Whenever a new point is presented, either it is assigned to one of the existing clusters or treated as noise, or it triggers a merge or split process. Our Incremental Clustering on ChainNet (ICC) model has the following salient features: (1) it can construct arbitrary shaped cluster, (2) it handles the temporary "noise" by using a 'noise box' we introduced, (3) its overall cluster structure is indexed in a Hierarchical ChainNet Summarization Tree (HC-ST), (4) it can adapt the number of clusters and the cluster sizes dynamically, (5) it is not sensitive to the data order, and (6) it requires little main memory storage and has $O(N^2)$ time complexity.

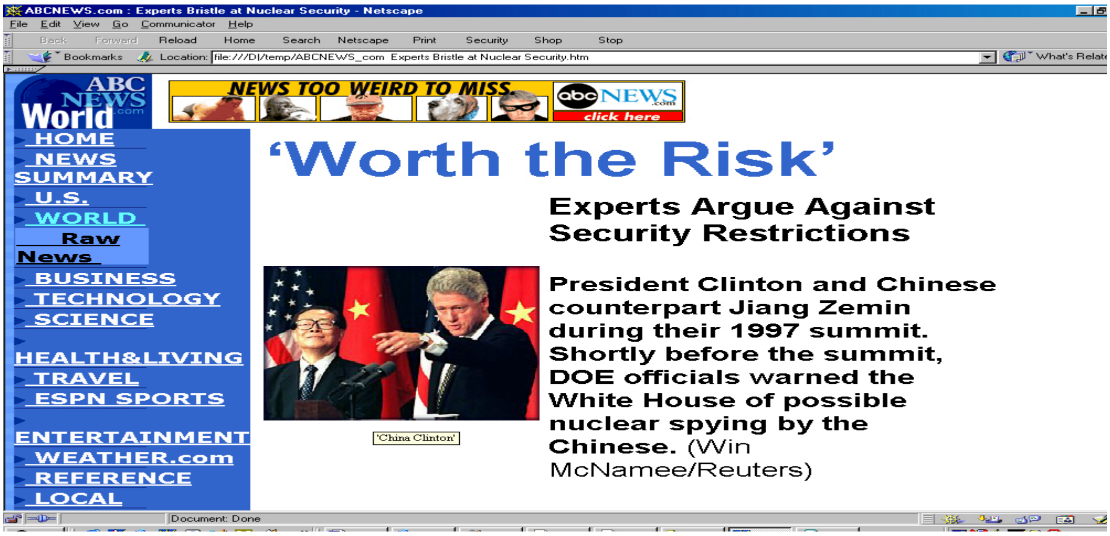



Figure 4: An example image from ABCNEWS.

| LC Type | Description |
|---|---|
| TLC | ap_china_nuclear_h |
| ALC | China Clinton |
| PLC | ABCNEWS Experts Bristle at Nuclear Security |
| SLC | President Clinton and Chinese counterpart Jiang Zemin during their 1997 summit |
| | Shortly before the summit, DOE officials warned the White House of possible nuclear spying by the Chinese |
| RSLC | None |
| CLC | President Clinton and Chinese counterpart Jiang Zemin during their 1997 summit. Shortly before the summit, DOE officials warned the White House of possible nuclear spying by the Chinese. |

Table 1: LCs in ChainNet of the image in Figure 4.

Figure 5 shows the main flow of the ICC algorithm. ICC starts as the first data point comes into the database. Initially for the first few points, we treat them as noise. There are several issues involved in this routine. (1) how to identify a cluster? (2) how to decide when to merge and when to split? (3) how to merge and how to split? (4) how to decide when to construct a new cluster from the noise dataset? (5) how to update the HC-ST in all cases? In the following, we will discuss them one by one.

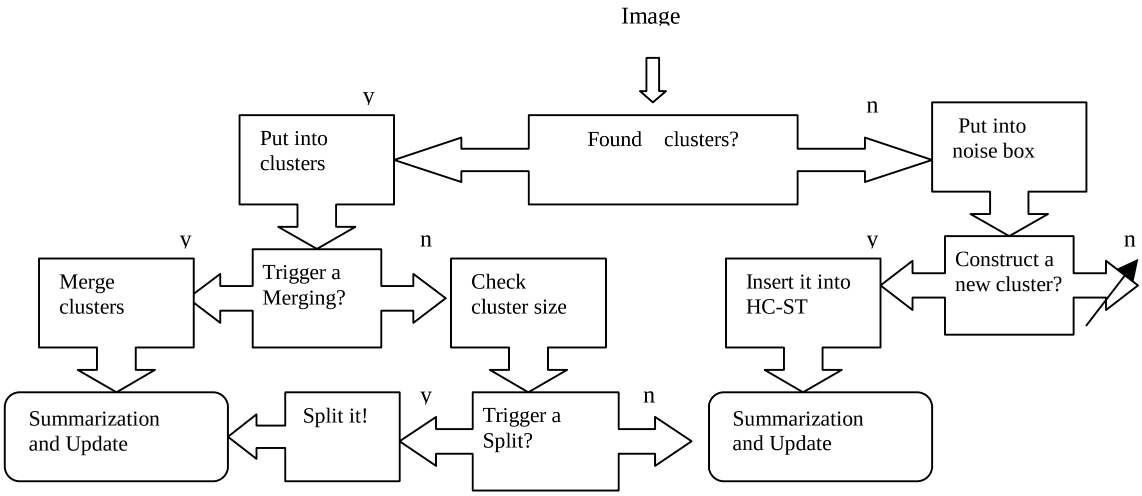



Figure 5: ICC routine.

**Hierarchical ChainNet Summarization Tree (HC-ST) Structure:** In ICC, we employ a hierarchical tree structure, Hierarchical ChainNet Summarization Tree, to facilitate the clustering and retrieval process. Figure 6 shows a HC-ST structure. In HC-ST, the leaf nodes are the summary information of clusters, one leaf node per cluster. The higher level (or internal) nodes are the summarized representation of all its children nodes. We shall defer the discussion on how the summary information is obtained to the next subsection.

All the nodes employ the same uniform data structure - ChainNet. The root node will represent the whole database's description. As we may notice, there is a 'noise box' in HC-ST. This 'noise box' contains all the temporary noises that are not related to any of the current clusters. Completely new clusters (i.e., those not related to existing clusters) are generated from the 'noise box'. This HC-ST starts with the generation of new clusters from the 'noise box', and is updated by merge and split operations. It may not be balanced as the split process goes on, but it always keeps a binary tree (except for the root node). The Split operation is a new operation that never occurs in static clustering methods. This hierarchical tree structure can speed up the process of finding the right cluster. To construct this tree, there are four main operations: insert, merge, split and generate new cluster. The first thing, of course, is to find the right cluster given a new point.

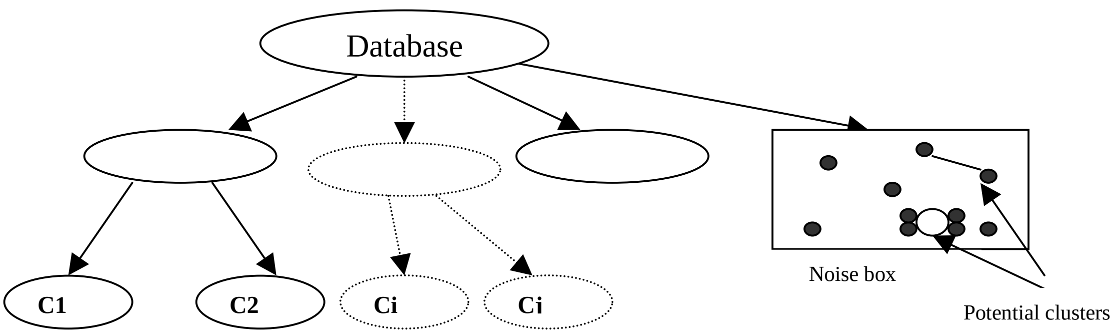



Figure 6: overview of HC-ST.

**Find the Right Cluster:** When a new image is to be inserted, we first determine which cluster it belongs to. ICC will traverse the HC-ST from the root until either a relevant cluster is found or no relevant internal node is found. To decide whether an image's ChainNet is semantically related to any node's ChainNet, we use the *List Space Model* with *MatchScale as* discussed in Section 3.2. If the sum of the similarity and matchscale is not less than the relevance threshold, the two ChainNets are semantically related. This threshold is a linear function of the query's ChainNet. The traversal will stop if an image's ChainNet is not related to any child of the current node. It's possible that an image's ChainNet is related to the current node, but not to any of its children, since the current node contain summarized information that is a combination of all its children.

**Insert:** If a right cluster is found, we insert it to that cluster and build interconnection with relevant images (i.e., all related images are linked); otherwise, we put it into the 'noise box' and check if there are any related noise or small group. If there exists related group, we build the interconnection too. Any connected group in the 'noise box' may potentially become a cluster.

**Merge:** When a new image matches two or more clusters, a merge phrase may occur. However, merging will only occur for clusters that are related (based on the list space model). If the clusters are not related, we assign the new point into the closest cluster. To perform the merging, we merge the one with a longer path into the other to avoid long path; the cluster in the longer path is then removed. This will help to keep the tree as balanced as possible. Once the cluster has been removed from its original position, its original parent should be removed too since it has only one child (as shown in Figure 7). A new cluster representative will then be constructed for the merged cluster. This is done by combining the two old clusters' ChainNets to generate their summary information (see next subsection). Finally, all their internal nodes are updated correspondingly.

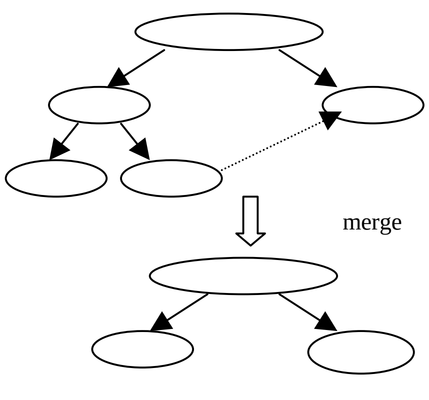



Figure 7: Illustration of the Merge operation

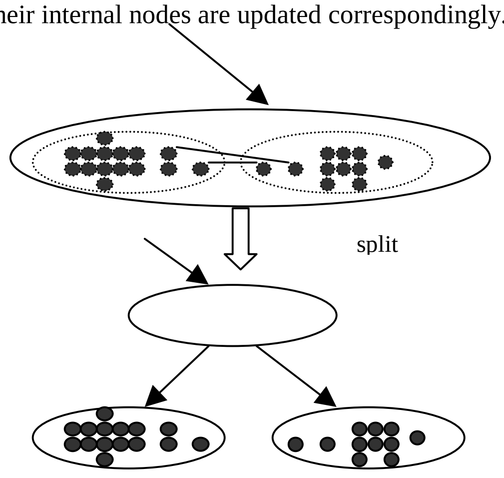



Figure 8: Illustration of the Split process.

**Split:** For a large cluster, it is possible that it may comprise a number of smaller clusters. When a cluster's size keeps increasing and reaches a certain predetermined upper bound on the size of the cluster, the cluster may be split if it has smaller clusters. To decide whether a cluster is made up of smaller clusters, we need not go through the data sets at all. The cluster's ChainNet structure can indicate this from the relevance density. The Split process breaks a large cluster node into two smaller clusters, and replaces the original node with the parent node of the two newly created smaller clusters (see Figure 8). The Split process forces the cluster's data structure in a binary hierarchical tree.

We note that there is a correlation between the number of clusters (denoted as *C*) and the upper bound on the size of a cluster (denoted as *U*). As the database keeps growing, it is better for *C* and *U* to be correspondingly increased. If we fix the cluster number, its size will be too small and too large during the early and late stage respectively. On the other hand, there are too few and too many clusters for early and late stage respectively. We found that it's not so good to define *U* as the linear function of *C*. Exponential function can express the relationship more. Hence, we define *U* as $U = \alpha * C^2$, where $\alpha$ is a scalable parameter. This formula provides a good adjustment between cluster number and cluster's upper bound. The relationship among data size, cluster number, and cluster upper bound can be shown as $N < U*C = \alpha*C^3$

**Generate new cluster**: We define the upper bound to control the cluster's size and indicate whether it's time to split a large cluster. We also define the cluster lower bound to decide when to generate a new cluster from the 'noise box'. In the 'noise box', either all points are not related, or a few points are connected to form small groups, but the number is fewer than the lower bound. These small groups are potential clusters. While small group keeps growing and reaches the lower bound, a new cluster is generated and connected to the root node in HC-ST. Here we relax the binary tree to be a tree in which each internal node has two children, except for the root node.

**Update HC-ST:** For all the operations mentioned above, updating HC-ST is necessary, from the leaf nodes to the root node. For a new image assigned into a cluster, the system will generate a new summarized ChainNet for the cluster by combining the image ChainNet and the summarized ChainNet of the cluster (see next subsection). All internal nodes' summarized information are updated in the same manner.

For merging, all the original cluster's ancestor nodes have to extract its semantics and all the newly created cluster's ancestor nodes have to add in that cluster's semantics. For splitting, it is simpler. The only thing to do is to create two new sub-clusters' representatives, and treat the original cluster's ChainNet as their parent. There is no need to update the ancestor nodes.

### 4.2 Summarization of ChainNet

Here, we present a two-step mechanism, called Vertical and Pyramidal Summarization Tree (VP-ST), that is used to generate summary data for the nodes of HC-ST. The goal here is to create compact and short, but effective representation. The difficulty is how to retain as much semantics as possible. VP-ST will summarize all the data points within one cluster and produce the cluster's representative. There are two steps: Vertical summarization, followed by Pyramidal summarization (see Figure 9).

**Vertical Summarization:** In this step, we will summarize each image's ChainNet into a summarized ChainNet. As shown in Figure 6, the original ChainNet is the node at the lowest level, and the summarized ChainNet is the middle-level node. Recall that the ChainNet comprises 6 types of LCs. Our solution is to reduce the LC length, and the number of LCs. To make a successful summarization, we will stress on these two means one after one. We shall illustrate how the LC length can be reduced using the example in Figure 4, and the initial set of LCs of the ChainNet in Table 1

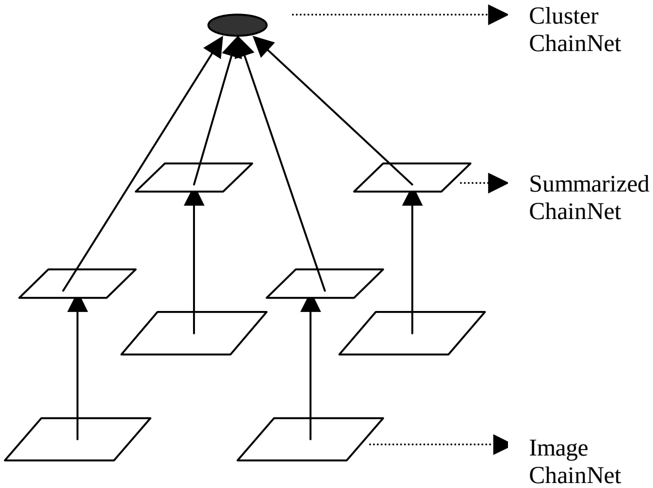



Figure 9: VP-ST structure

Recall that an LC is basically a sentence that carries certain semantics by its words. Since RSCL and CLC are generated from SLCs, the key is to reduce SLCs. It's known that in a sentence, the most important parts are subject, verb, and object. All other information is auxiliary. Therefore, we parse each LC into several parts: subject, verb, object and some auxiliary parts. Auxiliary parts can be identified by prepositions in LC. Auxiliary parts will be removed from the LC provided they are not semantically related to any other LC without considering CLC and RSLC. For instance, the second SLC can be parsed as 6 parts. The subject is “DOE officials”, the verb is “warned”, and the object is “the White House”, and there are 3 auxiliary parts: “Shortly before the summit”, “of possible nuclear spying”, and “by the Chinese”. After computing the similarity by using the list space model, ”of possible nuclear spying” and “by the Chinese” can probably be retained. This will depend on the threshold value that the user set. The last one “Shortly before the summit”, should be removed from that SLC. The same process is applied to all the other LCs. Finally, the new ChainNet LCs is shown in Table 2 after reducing the LC length, stop words, and applying word stem (we assume “by the Chinese” is removed. Nuclear word appears both in ALC and PLC).

| LC Type | Description |
|---|---|
| TLC | ap_china_nuclear_h |
| ALC | China Clinton |
| PLC | Experts Bristle Nuclear Security |
| SLC | President Clinton Chinese counterpart Jiang Zemin |
| | DOE official warn White House nuclear spy |
| RSLC | None |
| CLC | President Clinton Chinese counterpart Jiang Zemin. DOE official warn White House nuclear spying. |

Table 2: After Vertical step

The next step is to reduce the number of LCs. There are two ways to do so. The first is remove some LCs. It was shown in [33] that TLC, ALC and the first SLC play the most important roles in identifying an image’s semantics. They cannot be removed directly. If one LC is not semantically related to any other LC (except CLC), it should be removed. In this example, the second SLC has little similarity with TLC and PLC. It is likely to be removed. PLC is similarly dealt with. The other approach is to combine similar LCs. If one LC’s semantics can be identified by another similar LC, then we can merge one into the other by assigning the common words’ weight as the sum of both LCs’ weights, and move uncommon words into the other also. In this example, ALC in fact is contained completely in the first SLC. Therefore, there is no need to keep ALC. But we have to increase the common words’ weights by adding ALC’s weight. If there is only one SLC left, keeping a single copy is enough. Therefore, CLC is not needed any more. And the weight for SLC becomes the sum of both SLC and CLC. Finally, the summarized ChainNet for the example image is shown below (we assume the second SLC is removed).

| LC Type | Description |
|---|---|
| TLC | ap_china_nuclear_h |
| PLC | Experts Bristle Nuclear Security |
| SLC | President Clinton Chinese counterpart Jiang Zemin |

Table 3: The final summarized ChainNet

Comparing to the original ChainNet, some information is lost, like “white house” and “DOE official”. But by the inter-relevance, we can confirm that that information is not relatively important in determining the image semantic. By adjusting relevance threshold, we also can decide how much we want to lose. This will definitely affect the retrieval precision. The point is how much this will affect. We will show such effect in the performance study. The purpose of generating each summarized ChainNet for each image is to try to represent all the points as close as possible.

**Pyramidal Summarization:** After we have obtained all the summarized weight ChainNet for each data point in the cluster, it’s time to integrate them into one. In this phase, we want to keep as much information as possible. Every LC is very short, and important in deciding its image’s semantics. Since all images within a cluster are related, many LCs from different images must be related to each other. This provides much space for merging LCs. We apply the same methods as combining two LCs in the Pyramidal Summarization. In this step, no information is lost since all LCs are reconstructed into a more compact form.

There is an issue of a cluster’s center, or sub-clusters’ centers. In our VP-ST, they can be determined easily using the Pyramidal Summarization process: LCs that indicate a cluster’s center should be combined most often because most points are close to the center, so by looking at those most frequently combined LCs, whose words contain highest weight, we can decide where the cluster center is. If several small groups’ LCs are combined very often and there is not much interconnection in between, this most probably indicates that there are several sub-clusters within this cluster. Therefore, it’s good to split them when the size reaches the cluster upper bound.

## 4.3 Time and Space Complexity

The time complexity for list space model is O($L^2$). To compute the similarity between two ChainNets, we need O($L^2$*$D^2$), where D is the dimension. Here L and D are very small compared to *N*, the data size. To find the right cluster, we need to traverse O(ln(*C*) ) nodes. Therefore, the time to find a cluster is: O($L^2$*$D^2$ * ln (*C*)). Next is the update time. We take insert, which is the most frequent operation, as our example. Summarizing time is the same as comparing two ChainNets, which is O($L^2$*$D^2$). We need to update ln(*C*) nodes. Therefore, updating needs O($L^2$*$D^2$ * ln (*C*)) too. In fact, insert and update are two reverse direction operations. So the overall time complexity is still: O($L^2$*$D^2$ * ln (*C*)). For N data points insertion, we need time complexity $O(\boldsymbol{N} * L^2 * D^2 * ln\ (C))$. In this time complexity, there is no square on the large parameter *N*. and we believe $L^2 * D^2 * ln\ (C)$ will be much smaller than $N$ when database is very large. Therefore, we argue that our ICC is very efficiency to construct its HC-ST.

For the space usage, we only store HC-ST in main memory. There are around C*(C-1)/2 nodes, and each node is a compact ChainNet. So the overall size will be S* C*(C-1)/2, where S is the average ChainNet size. In our implementation, S is around 0.2 Kbytes. Therefore, for 1000 clusters, overall size is 0.2 * 1000 * 500 = 100 Mbytes. From the relationship between the cluster number and the cluster upper size, there could be as many as 1000*1000*1000*$\alpha$ = $\alpha$ billions data points. Comparing to this number, 100Mbytes is extremely small. Roughly it is (100*1000*10)/($\alpha$*100,000,000) = 10/$\alpha$*1,000 = (1/$\alpha$)%.

## 5. EXPERIMENTS

In this section, we present the architecture of ICICLE, a prototype system that we have implemented. ICICLE employed the Weighted ChainNet model to capture the semantics of WWW images, and the Incremental Clustering on ChainNet model to speed up the retrieval process. We also study both the effectiveness and the efficiency of the proposed methods through an extensive performance study. For the effectiveness improvement, we compared our image representation and measure models with [1]. We also compared our clustering strategy with two recently proposed clustering algorithms: CURE [18] and DBS [6].

### 5.1 Architecture of ICICLE

For purpose of testing the model, we "centralized" the image collection (instead of simply extracting the image at runtime from the various Web sites/pages in the form of a search engine). This is achieved through the design of a Web crawler that automatically searches the WWW for documents with embedded images. The crawler also extracts the image title, image ALT, page URL, page title and image caption from the HTML documents as the images' semantic content. In total, we collected 10,000 images from over 2,000 different URLs. These images are general, random and diverse enough for us to test on any query. Our overall system architecture is shown below: Whenever a new image comes, ICC module clusters it into the right partition. Whenever a query comes, Image Search Engine passes the query to ICC to allocate the right partition for comparison. We run the experiments on a 600 MHz CPU Pentium III PC with 128M RAM.

## 5.2 Experimental Setup

In our experiments, we used 12 text descriptions, as shown in the following table 4, as our queries. Given that we have over 10,000 images, it is not practical to scan all images to obtain the relevant images for each query. To determine the set of relevant images for the queries, we adopt the following realistic approach. For each query, we expand the query terms to include terms that are related. This is done using the WordNet [24]. For example, the term girl may be expanded to include the term woman. Each term is then used as a query to extract the list of images whose semantics (or rather the LCs) contain that term. The union of the results from each term form a candidate set of relevant images. We then manually examine the candidate set to eliminate those that are not semantically related to the query to get the final set of relevant images.

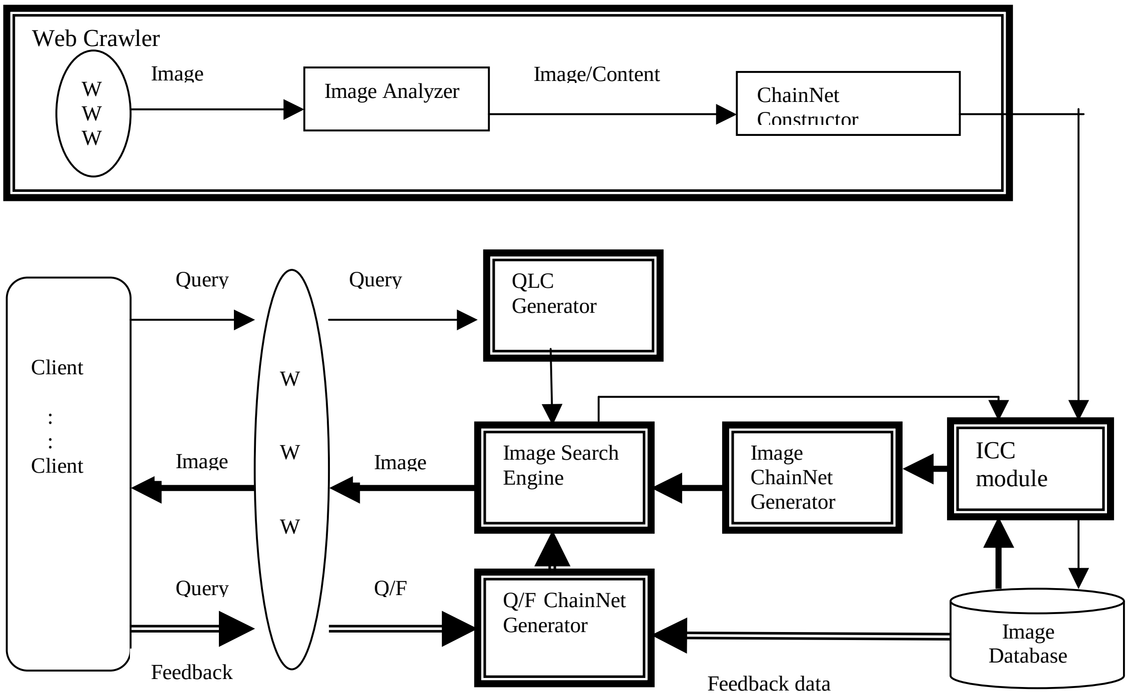



Figure10: Overall system structure in client-server form.

| Query | Query Description |
|---|---|
| Q1 | Singapore map |
| Q2 | Travel in Spain |
| Q3 | Valentine flower |
| Q4 | Island in the sky |
| Q5 | California beach girl |
| Q6 | England football league |
| Q7 | Green lizard on a red leaf |
| Q8 | Husband is kissing his wife |
| Q9 | National University of Singapore |
| Q10 | Hollywood superstar Jennifer Lopez |
| Q11 | Elephant in the beautiful national park |
| Q12 | Celebrations for new millennium of 2000 |

Table 4. Test queries.

## 5.3 Tuning the Weight ChainNet Model

In this set of experiments, we "turn off" the clustering strategy. This allows us to study the effectiveness of the Weighted ChainNet model in isolation.

### Tuning the Weights

Weight ChainNet model calls for some tuning to be performed. As mentioned earlier, there are 6 types of LCs and different LC types may have different significance in identifying the image semantics. In the first experiment, we evaluate the performance of each type of LCs exclusively to study their different impact on retrieval effectiveness. Figure 11 shows the results.

From Figure 11, we can see that for TLC, it cannot achieve >20% recall, although it has high precision. This is due to the lack of information in TLC. For similar reasons, PLC and ALC are not very effective also. For RSLC, since quite a number of images do not have RSLC, it cannot achieve high recall. Only CLC and SLC can result in high recall, but the precision is not satisfactory. TLC, ALC, PLC and RSLC can be used to improve the precision a lot. On the other hand, SLC and CLC can improve the recall. From this result, we have a rough picture of the relative importance of each type of LCs. Clearly, SLC is the most important, followed by TLC, RSLC, ALC or PLC, and finally, it is CLC.

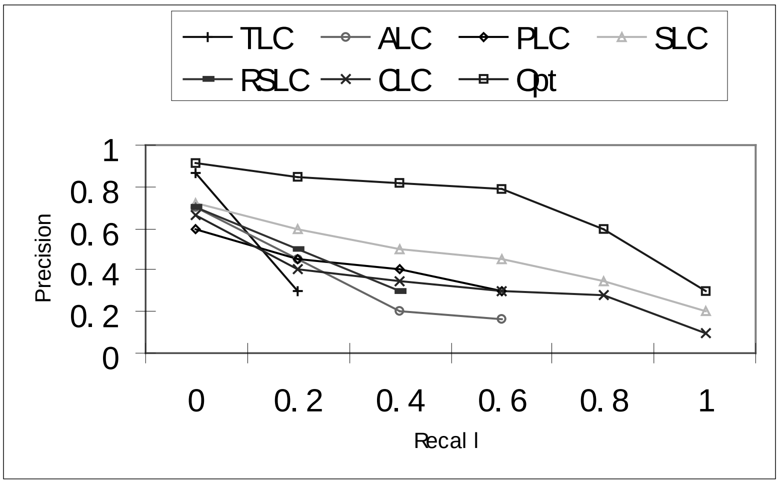



Figure 11: Utility by each Type LC alone to Represent Image

To determine the weights to be assigned to the proposed model that combines all the LCs, we tested different weight combinations from values in 0, 0.2, 0.4, 0.5, 0.6, 0.8 and 1 for each type of LCs. However, we narrowed the search space based on the result from Figure 4 by adopting some simple heuristics. For example, since SLC is the most important, we fixed its weight at 1.0. Moreover, for a LC that is more important, the weights assigned to the other less important LCs cannot be more than its weight. In total, we tested 22 combinations and obtained the following weight assignment for the various LCs: TLC (0.8), ALC(0.6), PLC(0.6), SLC(1), RSLC(0.5), CLC(0.2). In this experiment, we have fixed the scale parameter *coef* of the match level to be 0.6

(see Tuning the Match Level). We shall refer to this scheme as OPT. We also presented the result of OPT in Figure 4. As shown, we can get more than 80% precision with recall of 60%.

Though this experiment is meant to tune the proposed method, we note that it is also a comparative study among the different schemes. Clearly, the results show that using a single LC exclusively cannot provide the best performance, even though such an approach is clearly simple. Moreover, it shows that proper combinations of the various LCs can lead to very effective retrieval results. We also note that the exclusive CLC scheme can be viewed as a form of text-based system [1] without any semantic structure involved. Thus, we expect OPT to outperform existing schemes too.

### Tuning the Match Level

There is another parameter that we have to tune, the match level. Recall that the match level is the number of common terms shared by two lexical chains. It determines whether two LCs are semantically related, and then derives if two images are semantically related. In our evaluation system, only those semantically related images are returned.

One single word cannot reflect the semantic meaning of a whole query. If the match level threshold is too small, too many images may be returned to the users. On the contrary, too few images are displayed if the match level threshold is too high. Therefore, it is necessary to choose the best match level thresholds. Since the length of a query is a random variable, a fixed value for match level is not applicable to various queries. We thus define the match level as a linear function of query length:

*MatchLevel Threshold= coef * query.length()+ constant*

where the *coef* is the scale parameter we need to explore in order to get the best results in a reasonable volume. And the *constant* is just an adjustable value.

We tested those 12 queries in Table 4 in order to select the best *coef.* Figures 12a and 12b shows the relationship between precision and *coef* and recall and *coef* respectively. From Figure 12a, when *coef* is > 0.6, the precision will be greater than 85%. From Figure 12b, we can see that when *coef* is < 0.6, the recall is greater than 60% which is very satisfactory to a large image database. Therefore, observing the combined effect, we select 0.6 as the optimal value of *coef.*

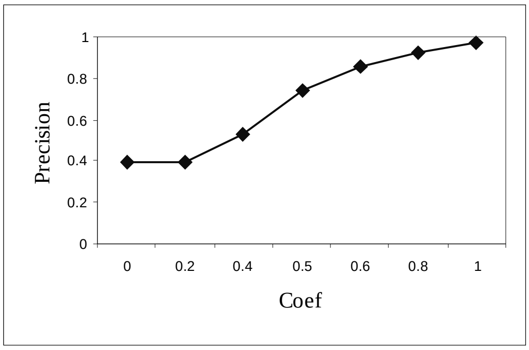



Figure 12a: Precision Vs. Coef

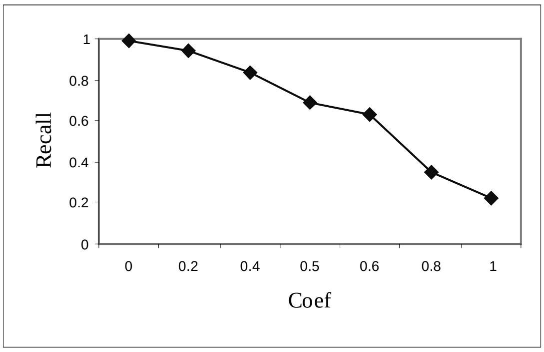
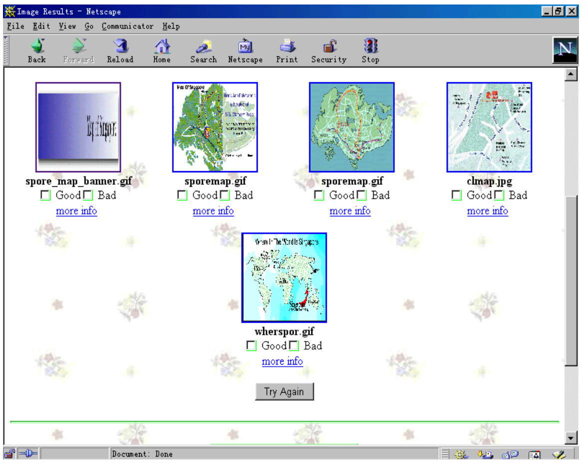
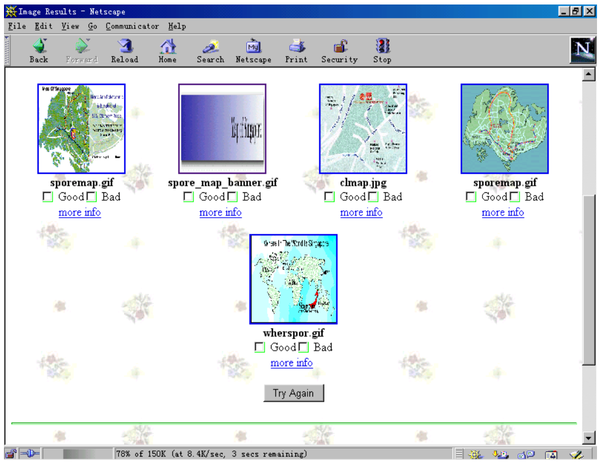



Figure 12b: Recall Vs. Coef




(A) Q1 results before Applying Match Scale




(B) Q1 Results after Applying Match Scale.

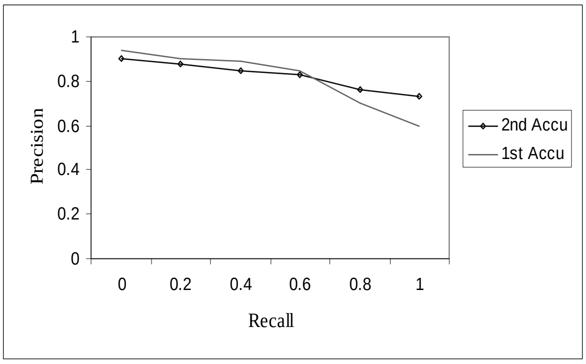
Figure 13: Effect of Match Scale.

### Impact of Match Scale

Match scale explores the importance of match order in the lexical chain. It has the effects in terms of image ranking during presentation of the returned images. Images with higher similarity measures will be returned to users ahead of images with lower similarity values. Figure 13 shows a sample results obtained from Q1. As shown, by considering the match scale, we can get more relevant images being displayed earlier.

## 5.4 Feedback Mechanisms

In this experiment, we study the effectiveness of the two proposed feedback mechanisms: *semantic accumulation* and *semantic integration and differentiantion*. Figure 14 shows the improvement by the two methods respectively. *Opt* is the basic Weight ChainNet model without feedback. *Accu* denotes the semantic accumulation method. And *I&D* represents the semantic integration and differentiation method. We note that Accu and I&D represents one application of the feedback loop after Opt returns its resultant images. To clearly see the effect of the noise that semantic accumulation brought, Figure 15 presents the results for the first feedback and second feedback by semantic accumulation. We can see that the second searching actually has a bit lower precision, but with relatively higher recall. But semantic accumulation method has the advantage that the returned images are more semantically related to the specific image selected by user - the feedback image.



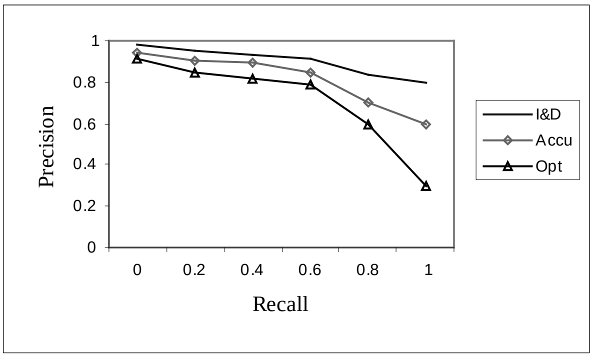
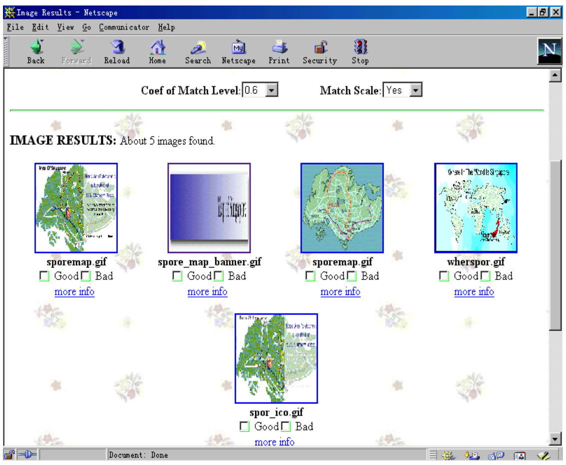




Figure 14: 1st and 2nd try by Semantic Accumulation.

Figure 15: Comparison of feedback mechanisms.

Figure 16 shows a sample feedback run of the Accu method for Q1. Compared to the results generated from OPT (the basic Weight ChainNet model without feedback), we see that the set of images retrieved are more relevant. Figure 17 shows a sample feedback run of the I&D method for Q1. From earlier results without feedback mechanism (see Figure 13), we have identified two relevant and one irrelevant image. As shown in Figure 17, the resultant images are not only more relevant than OPT and Accu approaches, the irrelevant image has also been pruned. In addition, more relevant images have been retrieved.



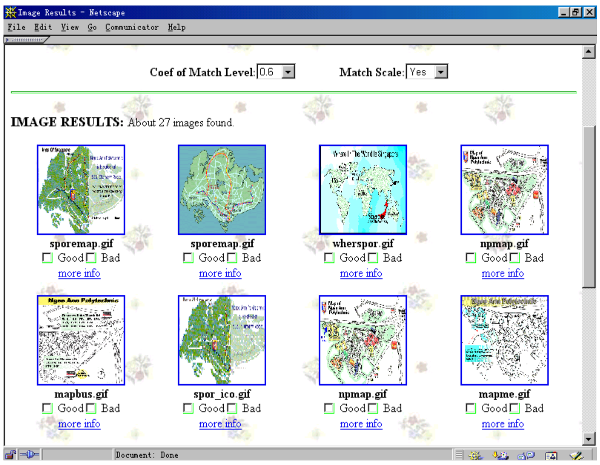




Figure 16: One-step feedback of Accu for Q1.

Figure 17: One-step feedback of I&D for Q1.

### 5.4 Comparative Study on Clustering Techniques

In this experiment, we compare ICC with two static clustering algorithms proposed recently: **CURE** (Clustering by Representatives) and **DBS** (Density Biased Sampling. For both CURE and DBS, sampling is used. We set the default sample size to 125, and the number of representative per cluster is 10. We also compared against the original ChainNet model (i.e, without ICC). Before that, let us examine the "optimal" $\alpha$ **value** to be used for ICC.

### Tune the α value

There is one important scalar parameter α, which indicates the relationship between a cluster's upper bound and and the number of clusters. As the image database keeps growing, it becomes more important to determine the optimal value for α. In this experiment, we vary α from 0.1 to 10. α is "optimal" if it can produce reasonable number of clusters and cluster's upper bound that lead to high precision with recall. Figure 18 shows the results (without applying summarization). The smaller the α, the smaller the cluster upper bound. The larger the α, the larger the cluster upper bound. From Figure 18, when α is around 2, the recall reaches 0.65, with a high precision of 0.9. This means that our clustering method can distinguish some irrelevant images and avoid inserting them into clusters. After 2, recall does not change much, while precision decreases. And before 2, the recall is not high enough. Therefore, we choose α to be 2. High recall indicates we always cluster the relevant images together and high precision shows irrelevant images are separated. There is no dramatically up or down, which confirms the successful relationship control between cluster's upper bound and the cluster number.

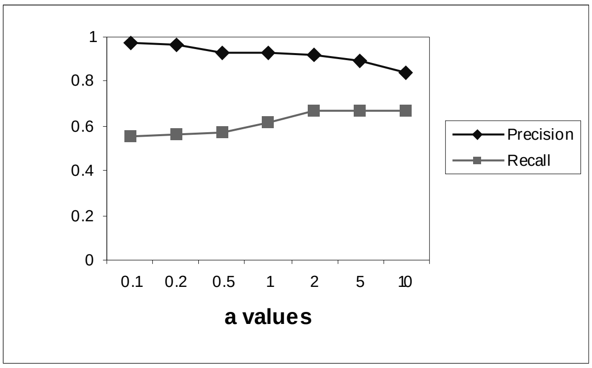



Figure 18: α effect on precision and recall.

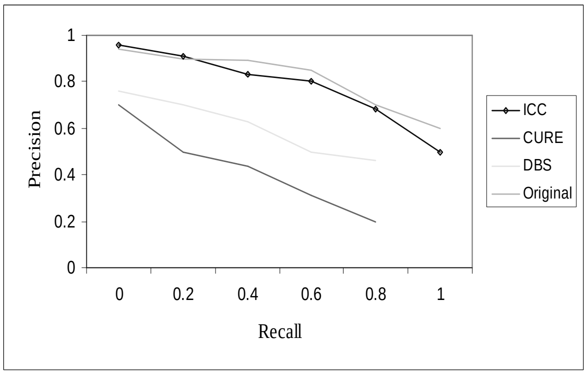



Figure 19: On retrieval effectiveness.

### Effect on Retrieval Effectiveness

We note that the retrieval effectiveness with and without ICC can be different. This is because, under ICC, only some relevant partitions will be searched. It is not impossible that some relevant images fall into a partition that is not examined during the retrieval process. Figure 19 shows the retrieval effectiveness of the various schemes. First, we note that the original scheme without clustering performs the best. This is expected since partitioning the database implies some data may be put into a partition that is not searched. Second, the proposed ICC performs nearly as well as the original scheme. We observe that the retrieval precision is slightly lower when recall is greater 0.3. Given the same precision, the original system has higher recall as it checked every single point. But the difference is not so much. At a reasonable recall of 0.6, the different is as little as 5%, with greater than 80% precision also. When recall is less 0.3, the precision is even higher. This is because ICC has the ability to separate some irrelevant new coming images when they are trying to find the right clusters. Third, ICC outperforms DBS and CURE by a wide margin. Because DBS has a better cluster's representation by density weighting than CURE, DBS achieved better performance. One possible reason for this that CURE and DBS are based on sampling and sampling techniques are not well suited for textual data. For textual data, once some information is lost, it's hard to recover by sampling points, if not impossible. Finally, CURE and DBS cannot achieve 100% recall. This means that both do not cluster well for relevant images. Some are either clustered into other cluster or still in the 'noise box'. These results clearly demonstrate that our ICC can produce quality clusters. It can cluster most similar points together and separate unrelated points way by split. By applying ICC, we still can achieve high precision with high recall.

### Effect on Retrieval Efficiency

The main goal of clustering is to improve the retrieval efficiency at the expense of retrieval effectiveness. Figure 20 shows the results on the retrieval efficiency.

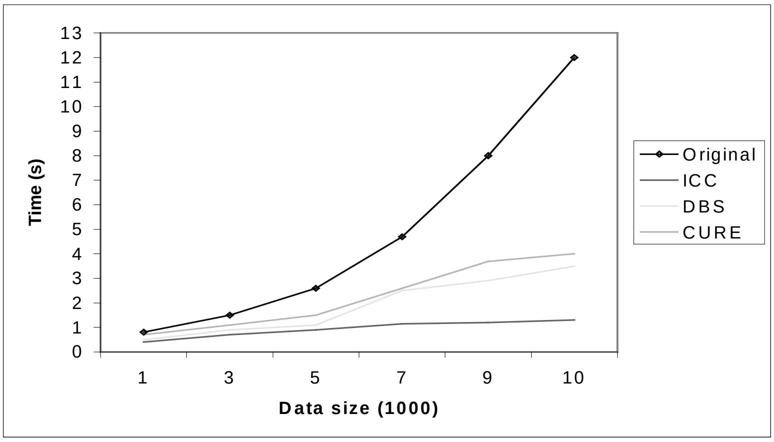



Figure 20: ICC's effect on retrieval efficiency

As shown in the figure, the original scheme (without clustering) performs poorly in terms of efficiency. This is expected since the scheme examines the entire database, while the cluster-based scheme only examines some relevant partitions. We also observe that both DBS and CURE are not as efficient as we expected. There are several possible reasons. First, both are quadratic time complexity. While for ICC, it's $O(N*L^2*D^2*\ (ln\ (C)\ +U))$. To scan the whole database, the complexity is $O(N*N*\ L^2*D^2*\ )$. With the data size increasing, the difference between ICC and the other schemes widens as $(U+ln(C))$ becomes smaller and smaller compared to $N$. Second, there is no automatic control on cluster's size and cluster number in CURE and DBS. There is a danger to cluster many images into a single cluster if they are from the same website. For example, if most images are from BBC websites, both methods may probably cluster them into a single group. In fact, this group may contain several sub- groups about sports, nature, entertainment, and so on. To confirm what we thought, we explore the inner clusters in the database resulting from these two methods. There are two extremely large clusters regarding BBC and National University of Singapore. Whenever searching these two, it takes longer time than ICC. Due to the split operation with well control on cluster's upper bound, ICC can always split extremely large clusters into smaller ones after identifying its sub-shapes. To summarize, ICC speeds up the original retrieval significantly. The larger the data set, the greater is the gain. From Figures 19 and 20, we have reason to argue that our ICC can perform very well in WWW image retrieval, both in effectiveness and efficiency.

## 6. CONCLUSIONS

In this paper, we have presented several techniques to solve the issues involved in an effective and efficient WWW image search engine. We first introduced a new model to represent the content of images embedded in WWW pages. The proposed Weight ChainNet model combines different types of lexical chains obtained from the surrounding text of an image. Our experimental study showed that the approach can be used as an effective means to represent image semantics. We also proposed two novel feedback mechanisms. In particular, the semantic integration and differentiation method returned more accurate results than semantic accumulation with higher recall.

Moreover, we have also presented a new incremental clustering algorithm ICC for the increasingly growing large database collection of WWW images. Our experiments showed that ICC can produce quality clusters, and can adapt the cluster size and cluster number dynamically. Without looking at the actual data points, ICC can identify the sub-clusters within a large cluster by checking the cluster representative's property. It can also handle temporary noise well by using a special 'noise box' from which new clusters can be generated. To improve efficiency, ICC employs a summarization step called Vertical and Pyramidal Summarization Tree. VP-ST starts from all the data points in the cluster, and finally converges all summarized points into single representative of the cluster. Our experiments indicated that this vertical and pyramidal technique provides quality representative, especially compared with random sampling. To further speed up searching right cluster, the clusters' structure is in Hierarchical ChainNet Summarization tree. A prototype system, called ICICLE, that employs the proposed models have been deployed.

## 7. REFERENCE


[1] A. F. Smeaton and I. Quigley. Experiments on Using Semantic Distances Between Words in Image Caption Retrieval. In *ACM SIGIR*, 1996.

[2] Aggarwal C. C., Wolf J. L., Yu P. S., Cecilia P. and Jong S. P. Fast algorithms for projected clustering, In *ACM SIGMOD*, 61-72, 1999.

[3] Aggrawal R., Gehrke J., Gunopulos D., and Raghavan P. Automatic subspace clustering of high dimensional data for data mining applications. In *ACM SIGMOD*, 94-105, 1998.

[4] C. Frankel, M. J. Swain, and V. Athitsos. Webseer: An image search engine for the world-wide web, *U. of Chicago Tech. rep. TR* 96-14, 1996.
[5] Chang S.F, Chen W., and Sundaram H. Semantic visual template-linking visual features to semantics. In *ICIP*, 1998
[6] Christopher R., Palmer and Christos F. Density biased sampling: an improved method for data mining and clustering, In *ACM SIGMOD*, 82 – 92, 2000.
[7] Chua T.S., Low W.C., and Chu C.X. Relevance feedback techniques for color-based image retrieval. In *Multimedia Modelling'98*, Oct 1998.
[8] Croft W.B., and Tutle H.R. Retrieval strategies for hypertext. *Information Processing & Management*, 29(3), 1993.
[9] Dunlop M.D. Multimedia information retrieval, Ph.D. Thesis. *Computing Science Department, University of Glasgow, Report* 199l/R21, 1991.
[10 Dunlop M.D., and Van Rijsbergen C.J. Hypermedia and free text retrieval, *Information Processing & Management,* 29(3), 287-298, 1993.
[11] Ellen M. Voorhees and Hou Y.W. Vector expansion in a large collection, In *TREC-1*, 1993.
[12] F. Chen, U. Gargi, L. Niles, H. Schuetze, Multi-Modal Browsing of Images in Web Documents, In *SPIE Document Recognition and Retrieval VI*, Vol 3651, 122-133, 1999.
[13] Frei H.P, and Stieger D. Making use of hypertext links when retrieving information. In *ACM-ECHT'92*, 102-111, 1992.
[14] Frisse M.E. Searching for information in a hypertext medical handbook. In *Com of the ACM*, 31(7), 880-886, 1988.
[15] Fass Adam, Eric Bier, and Eytan Adar, PicturePiper: Using a Re-configurable Pipeline to Find Images on the Web, *UIST 2000*, San Diego, CA, Nov. 5-8, 2000.
[16] G. H. Cha, C. W. Chung, An Indexing and Retrieval Mechanism for Complex Similarity Queries in Image Databases, *Journal of Visual Communication and Image Representation*, Vol. 10, No. 3, 268-290. 1999.
[17] G. Pass, Ramin Z., and Justin M.. Computing images using color coherence vectors. In *ACM MM*, 65-73. 1996
[18] Guha S,Rajeev R. and Kyuseok S. CURE:an efficient clustering algorithm for large databases. In *ACM SIGMOD*, 1998.
[19] Harmandas, Sanderson M. and Dunlop M. D. Image retrieval by hypertext links, *ACM SIGIR*, 296–303, 1997.
[20] Hinnerburg A., and KeiM D.A. An efficient approach to clustering in large multimedia databases with noise, In *Proc. 4th Int. Conf. on Knowledge Discovery and Data Mining*, 1998.
[21] M. Charikar, C. Chekuri, T. Feder, and R. Motwani, Incremental clustering and dynamic information retrieval, *In ACM Sympos. on Theory of Computing*, 626—634, 1997.
[22] Marco LaCascia, Saratendu Sethi, and Stan Sclaroff. Combining Textual and Visual Cues for Content-based Image Retrieval on the World Wide Web. *In Proceedings of IEEE Workshop on Content-Based Access of Image and Video Libraries*, 1998.
[23] Michael Ankerst, Markus M. Breunig, Hans-Peter Kriegel and Jörg Sander. OPTICS: ordering points to identify the clustering structure, In *ACM SIGMOD*, 49 – 60, 1999.
[24] Miller G.A., Beckwith R., Felbaum C., Gross D. and Miller K. Introduction to WordNet: an on-line lexical database. *Revised Version* 1993.
[25] Morris J. and Hirst G. Lexical cohesion computed by thesaural relation and an indicator of the structure of text, *Computational Linguistics*, 17(1), 22-48, 1991.
[26] Ng R.T., and Han J., Efficient and effective clustering method for spatial data mining", In *VLDB*, 144-155, 1994.
[27] Niblack W., Barber R., and Equitz W. The QIBC project: querying images by content using color, texture, and shape. *Technical report, IBM* RJ 9203(81511), 1993
[28] V. E. Ogle and M. Stonebraker. Chabot: Retrieval from a Relational Database of Images. *IEEE Computer Magazine*, 28(9):40--48, September 1995.
[29] Ooi B.C., Tan K.L., Chua T.S. and Hsu W. Fast image retrieval using color-spatial information, In *VLDB Journal*, 7(2), 115-128. 1998.
[30] Price R., Chua T.S, and Hawamdeh S.Al. Applying relevance feedback on a photo archival system. *Journal of Information Science*, 18:203-215. 1992.
[31] R.K. Srihari. Automatic indexing and content-based retrieval of captioned images. *IEEE Computer, 28(9),* 49-56, 1995.
[32] Sheikholeslami G., Chattajee S. and Zhang A., WaveCluster: a multi-resolution clustering approach for Very Large Spatial Databases, In *VLDB*, 1998.
[33] Shen H.T., Ooi B.C., AND Tan K.L. Giving meanings to WWW images. In *ACM MM*, 39-47, 2000.
[34] J. Smith and S. Chang, Image indexing and retrieval based on color histograms. In *ACM MM*, 87-98. 1996.
[35] J. Smith and S. Chang, An Image and Video Search Engine for the World-Wide Web, *Proc SPIE Storage and Retrieval for Image band Video Databases V*, Vol 3022, 84-95, 1997.
[36] J. Smith and S. Chang, Visually Searching the Web for Content. *IEEE Multimedia*, 4(3):12-20, 1997.
[37] Stan Sclaro, Leonid Taycher, and Marco La Cascia. Imagerover: A content-based image browser for the world wide web. In *Proc. IEEE Workshop on Content-Based Access of Image and Video Libraries*, 1997.
[38] Sougata M,Kyoji H, and Yoshinori H.Amore:A World Wide Web Image Retrieval Engine. *World Wide Web*, 2(3) 1999.
[39] Tan K.L., Ooi B.C. and Yee C.Y. An evaluation of color-spatial retrieval techniques for large image databases, *Multimedia Tools and Applications*, 14(2), 2001.
[40] Voorhees E.M., Query expansion using lexical-semantic relations. In *ACM SIGIR*, 1994.
[41] Yuksel Alp Aslandogan, Clement T. Yu, Evaluating Strategies and Systems for Content Based Indexing of Person Images on the Web, In *ACM MM*, 313-321, 2000.
[42] Zhang Tian, Raghu Ramakrishnan and Miron Livny. BIRCH: an efficient data clustering method for very large databases, In *ACM SIGMOD*, 103 – 114. 1996.
[43] Zheng C., Liu W., C. Hu, M. Li, and H. Zhang. iFind: A Web Image Search Engine,In *ACM SIGIR 2001*.
[44] Zheng C., Liu W., F. Zhang, M. Li, and H. Zhang. Web Mining for Web Image Retrieval, *Journal of the American Society for Information Science and technology*, Vol. 52, No. 10. 2001.