International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-4 Issue-6, August 2015

Scalable Image Search System

Ansila Henderson, Kavitha K V

***Abstract*—Several applications such as fingerprint identification, biodiversity information systems, digital libraries, crime prevention, medicine, historical research, among others uses the image search system for searching similar images. The goal of the scalable image search system is to support image retrieval based on content properties such edge and texture, encoded into feature vectors. Hashing technique is used to embed high dimensional image features into hamming space. The image search can be performed in real-time based on Hamming distance of compact hash codes. An extensive experiment on flickr image dataset demonstrates the performance of the proposed methods.**

***Index Terms*— Gray-Level Co-Occurrence Matrix (GLCM), Homogeneous Texture Descriptor (HTD, The Edge Histogram Descriptor (EHD)).**

I. INTRODUCTION

With the expansion of the Internet, and the accessibility of image capturing devices such as digital cameras, massive amounts of images are being produced every day in different areas including remote sensing, architecture, medicine, fashion, crime prevention, etc. The image search system is used for searching, browsing, and retrieving images from huge databases consisting of digital images. For some methods, metadata is used for retrieving images but some others accepts query image from users and returns the images similar to that of given image.

Today, the Text Based Image Retrieval (TBIR) is used for general-purpose web image retrieval systems. This method uses the text associated with an image to ascertain what the image contains. This text can be the image's filename, or any other part of text that is associated with the image. Google and Yahoo Image search engines are the systems with this method. These search engines are speedy and vital, but they sometimes fall short to retrieve relevant images, because the text could not fully explain the semantic content of the images.

Content Based Image Retrieval (CBIR) is a collection of techniques for retrieving semantically-relevant images from an image database based on automatically-derived image features [1]. It retrieves images based on their visual similarity to a query image given by the user. The main objective of CBIR is its effectiveness during image indexing and retrieval, so reducing the need for human intervention in the indexing process [2]. The CBIR systems extracts feature from every image based on its pixel values and defines algorithms for comparing images. The extracted features are used as the image representation to measure the similarity. The query image is compared with the images in the database by manipulating the dissimilarity of its feature vectors to other image descriptors. Several features were extracted from each image, such as colour, texture etc. Hashing techniques are used for indexing the high dimensional image features.

Manuscript published on 30 August 2015.
* Correspondence Author (s)
Ansila Henderson, Computer Science and Engineering, SCTCE Pappanamcode, Thiruvananthapuram, India
Kavitha K.V, Computer Science and Engineering, SCTCE Pappanamcode, Thiruvananthapuram, India.

II. RELATED WORKS

Recent image feature representations such as bag-of-words (BoW) [3] were similar to the bag-of-words representation of textual documents. So the methods used for document retrieval were adopted into the image retrieval task. The existing works on efficient search mechanisms can be mainly classified into three categories [4]. They are: inverted file, tree based indexing and hashing techniques.

A. Inverted File

J. Zobel proposed the inverted file to retrieve similar documents [5]. Inverted file is an index structure used for text query evaluation. The inverted file contains a collection of lists for each term which stores the identifiers of the documents containing that term. Thus it maps each terms to the documents. It has two major components:

- Search structure or vocabulary
- Set of inverted lists

The search structure stores a count and a pointer for each term. The count determines the number of documents containing the term and the pointer points to the start of the corresponding inverted list. Similarly, the word positions within the documents can also be recorded. These components provide all the information needed for the query evaluation. The textual queries usually contain very few words. But a single image may contain hundreds of visual words, which results in a large number of candidate images. So in the case of images, it needs additional verification. This makes difficult in using inverted files for large scale image search. By increasing visual vocabulary size in BoW the number of candidates can be reduced, but it will increase memory usage [6].

B. Tree-based Indexing

In order to achieve fast visual search indexing with tree-like structures [7], [8], [9] were frequently applied. For fast image descriptor matching C. Silpa-Anan and R. Hartley proposed Optimized KD-trees [10]. The high-dimensional vectors were stored as the elements in the KD-trees. At the root of the tree, the data can be split into two halves by a hyper plane orthogonal to a chosen dimension at a threshold value.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

The midpoint of the dimension with the greatest variance in the data set is chosen as the point to split. By comparing the query vector with the splitting value, it determines to which half of the data the query vector resides. Each of the two halves of the data is recursively split in the same way to generate a fully balanced binary tree. Each leaf node of the tree would be similar to a single point in the data set. In some cases, the leaf nodes may contain more than one point. The height of the tree would be $log_2 N$ where N is the number of points in the data set [10].

The KD-trees works effectively in low dimensions, but their efficiency decreases for high dimensional data. A large number of nodes would be there to search in high dimensional images. So to find the optimal solution, the KD-tree takes a lot of time to backtrack through the tree. The recent researches were aimed at keeping the backtracking within reasonable limits and increase the probability of success.

C. Hashing Techniques

Hashing has a major advantage in speed since it allows constant-time search. Using the hash codes, image similarity can be efficiently measured (using logical XOR operations) in hamming space. The binary hash codes were compactable to memory. The search can be performed efficiently using the hash table lookup or bitwise operations. Thus it satisfies time and also the memory requirements. The existing hashing methods can be divided into three categories: unsupervised methods, supervised methods, and semi-supervised [11] methods.

Most of the existing hashing techniques such as Locality Sensitive Hashing (LSH) [12], spectral hashing (SH) [13] were *unsupervised*. But the similarity in image search is not simply equivalent to the proximity of low-level visual features, but is more related to high-level image semantics (e.g., objects and scenes). Several *supervised* methods were also proposed recently to learn good hash functions [14], [15]. In [16], Wang *et al.* proposed a semi-supervised hashing algorithm that learns hash functions based on image labels. The advantage of semi-supervised hashing method is that for learning the hash functions it exploits unlabeled data as well as utilizes given labels. It is suitable for image search because it helps in the cases where only a limited number of labels are available.

III. THE PROPOSED SYSTEM

Fig.1 shows the block diagram of the Scalable Image Search System. It consists of two key components—image feature representation and search mechanism. The image features are represented using edge, colour and texture features such as wavelet decomposition, GLCM etc. The extracted features are represented by feature vectors. The feature vectors are then embedded into hash codes and stored in database. Query image is the input to the system and the output will be similar images from the database. From the given query image, features are extracted and form a feature vector. Hash codes are used for indexing and retrieving the images using hamming distance.

IV. FEATURE EXTRACTION

The three feature extraction techniques are adopted in this work. They are wavelet decomposition, homogeneous texture descriptor and edge histogram descriptor.

A. Wavelet Decomposition

In wavelet decomposition, an image is divided into four sub images by using the Haar wavelet filter function. The four sub images include LL, LH, HL and HH. The LL sub image is a smaller summary image of the original image. The LH and HL sub images highlight the horizontal and the vertical edges in the image respectively.

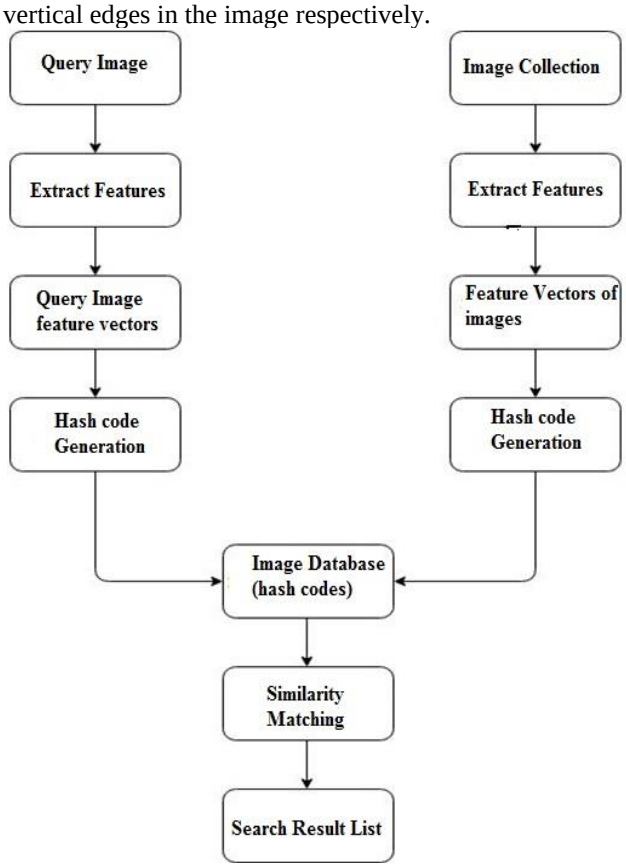


Fig.1. Proposed System

The HH sub band coefficient is less significant. Each pixel in the LL sub-band represented by the magnitude of a two-dimensional vector consists of the LH and HL coefficients. If the wavelet vector magnitude of a pixel t_1 is denoted by x_1, then:

$$x_1 = \sqrt{h^2 t_1 + v^2 t_1} \qquad (1)$$

where $t_1.h$ and $t_1.v$ represents the HL and LH sub-band coefficients. This representation ensures the invariance to rotations, since the x measures the total strength of an edge in both the vertical and horizontal directions [17], [18].

LL	**LH**
HL	**HH**

Fig.2. Wavelet decomposition

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

B. Homogeneous Texture Descriptor

The Homogeneous Texture Descriptor (HTD) can be obtained by filtering the image with the Gray-Level Co-Occurrence Matrix (GLCM) [19], [20]. The GLCM is generated with four different directions 0^0,45^0, 90^0, 135^0 and model texture as two-dimensional gray level variation. The GLCM analyses pair of horizontally adjacent pixels of image and gives the joint probability of occurrence of two grey level values at a given offset. The image is first divided into sub blocks. Then GLCM is created from each of the sub blocks. Thus multiple GLCMs created from each image are joined to get better accuracy.

The GLCM consists of four important properties for image retrieval task. They are contrast (C), energy (E), correlation (Cr) and homogeneity (H). Correlation gives the measure of the joint probability occurrence of the specified pixel pairs. Contrast represents the measure of vividness of the texture pattern. Thus, the bigger the blocks that makes up the image, the higher the contrast. Energy is the measure of textural uniformity of an image. Homogeneity gives the measure of the closeness of the distribution of elements in the GLCM to the GLCM diagonal. For each pixel pairs (a, c) in GLCM:

$$C = \sum_{a,c} (a - c)^2 \, P(a, c) \qquad (2)$$

$$Cr = \frac{\sum_{a,c} (a - m_x)(c - m_y) P(a, c)}{\sigma_x \sigma_y} \qquad (3)$$

$$E = \sum_{(a,c)} P^2(a, c) \qquad (4)$$

$$H = \sum_{(a,c)} \frac{P(a, c)}{1 + |a - c|} \qquad (5)$$

where P represents the pixel values in (a, c)th position of GLCM, and are the mean of pixel values in the horizontal and vertical values in the GLCM respectively.

C. Edge Histogram Descriptor

The Edge Histogram Descriptor (EHD) [21] represents the spatial distribution of the edges using local edge histograms. First, the image is divided into 4x4 sub-images. As shown in figure 3, the Edge Histogram Descriptor considers five types of edges: vertical, horizontal, 45% diagonal, 135% diagonal, and non-directional. Using edge filters, frequency of each type of edges are determined, and a local edge histogram with five bins is generated for each sub-image, resulting in a total of 80 histogram bins. This vector can be augmented with the histograms of global and semi-global levels, calculated by grouping the local histograms of sub-images.

For edge feature extraction, the image space is first divided into non-overlapping square blocks and then extracts edge information from each block. Regardless of the image size, the sub-image must divide into a fixed number of image-blocks. To deal with the images with different resolutions the size of the image-block must be proportional to the size of original image.

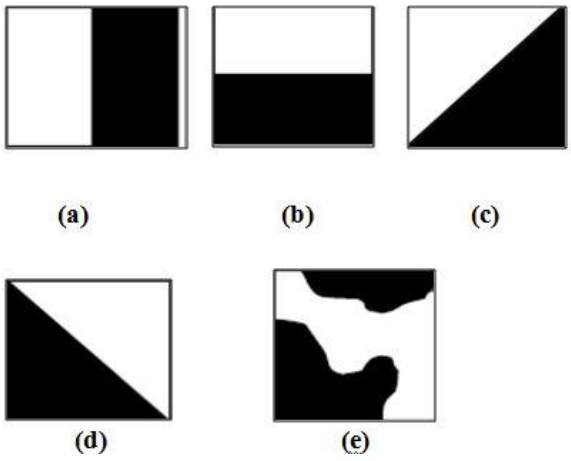


Fig.3. Five types of edges: a) vertical b) horizontal c) 45 degree d) 135 degree e) non-directional edges

V. HASH CODE GENERATION

Hash code is generated using semi-supervised sequential projection learning based hashing [23]. By iteratively updating the pair wise label matrix it implicitly incorporates bit correlation. Thus, each hash function corrects the errors made by the previous ones. For a given set of v points, $V = \{x_i\}, i = 1,2, \ldots v, x_i = \mathbb{R}^D$, a fraction of pairs are associated with two categories of label information. A neighbor-pair in which the pair of points share common class labels or neighbours in a metric space or. Similarly, a pair of points is called a non neighbour-pair if two samples have different class labels or far away in metric space. Given a vector w_k, the kth hash function is defined as

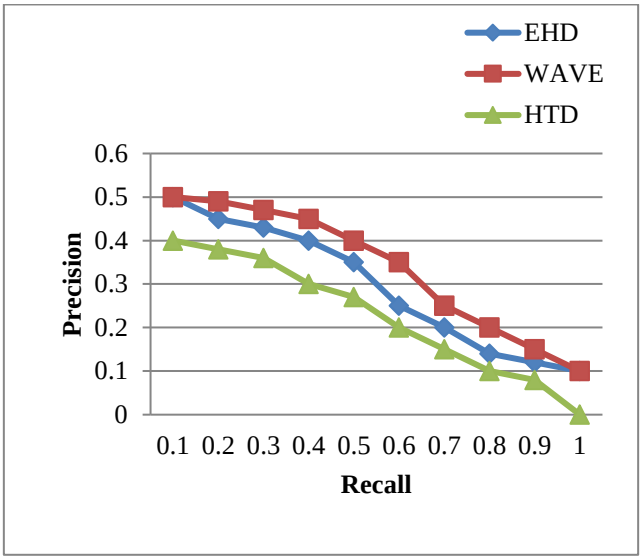
$$h_k(x_i) = sgn({w_k}^T x_i + b_k) \qquad (6)$$

where b_j the mean of projected data [24], i.e., $b_j = -\frac{1}{n}\sum_{j=1}^{n} {w_k}^T x_j = 0$ since P is zero mean. The higher weights are imposed on point pairs violated by the previous hash function. Thus the sequential process implicitly creates dependency between bits and gradually minimizes empirical error. The retrieved images are ranked based on image features.

VI. EXPERIMENT AND ANALYSIS

A set of Flickr images collected from the NUS-WIDE dataset [23] is used as the image collection. The NUS-WIDE dataset was created as a benchmark for evaluating multimedia search techniques. It contains images with much higher resolutions. First, the dataset is partitioned into two parts, 1K for query test and around 269K for training and constructing hash tables. The precision is evaluated based on whether the returned images and the query share at least one common semantic label. The performance was evaluated with code length 64-bit. With the use of semi-supervised sequential projection hashing, the system provides good performance in each feature extraction techniques.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)



The fraction of retrieved images relevant to a query is termed as precision. Recall is the fraction of the relevant images retrieved. A recall is a non-decreasing function of rank, while precision can be regarded as a function of recall rather than rank. In general, the curve closest to the top of the chart indicates the best performance.

$$Precision = x/x + y$$

$$Recall = x/x + z$$

where x =true positive, y= false positive and z= false negative. The x represents the number of relevant images retrieved, y represents the total number of irrelevant images retrieved and the term (x+z) represents the total number of relevant images in the database.

VII. CONCLUSION

With the increasing demands of multimedia applications over the Internet, the importance of image retrieval techniques has also increased. Based on the user's needs different feature extraction methods can be applied. The hashing methods are used to embed high-dimensional image features into Hamming space. Thus search can be performed in real-time based on Hamming distance of compact hash codes.

REFERENCES

1. P Maheswary, Dr. N Srivastava, "Retrieval of Remote Sensing Images Using Colour & Texture Attribute",(IJCSIS) International Journal of Computer Science and Information Security, Vol. 4,No. 1&2, 2009.
2. S. Oraintara, T. T. Nguyen, "Using Phase and Magnitude Information of the Complex directional Filter Bank for Texture Image Retrieval", IEEE International Conference on Image Processing, Vol. 4, Pages 61-64, Oct. 2007.
3. Datta. R, Joshi. D, Li.J, and Wang .J.Z, "Image retrieval: Ideas, influences, and trends of the new age," ACM Comput. Surveys, vol. 40, no.2, 2008.
4. Yu-Gang Jiang, Jun Wang, Xiangyang Xue, and Shih-Fu Chang, "Query-Adaptive Image Search with Hash Codes" IEEE transactions on multimedia, VOL. 15, NO. 2, Feb 2013.
5. J.Zobel and A. Moffat, "Inverted files for text search engines," ACM Comput. Surveys, 2006.
6. M. D. H. Jegou and C. Schmid, "Packing bag-of-features," Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2009.
7. J. L. Bentley, "Multidimensional binary search trees used for associative searching," Commun. ACM, vol. 18, no. 9, pp. 509–517, 1975.
8. M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in Proc. Int. Conf. Computer Vision Theory and Applications, 2009, pp. 331–340.
9. D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2006.
10. C. Silpa-Anan and R. Hartley, "Optimised kd-trees for fast image descriptor matching," Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2008.
11. M. Kan and S. Shan, "Semisupervised Hashing via Kernel Hyperplane Learning for Scalable Image Search," IEEE trans. circuits and systems for video technology, vol. 24, 2014.
12. P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in Proc. Symp. Theory of Computing, 1998.
13. Y.Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in Adv. Neural Inf. Process. Sys., 2008.
14. B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings," in Adv. Neural Inf. Process. Syst., 2009.
15. R.-S. Lin, D. A. Ross, and J. Yagnik, "Spec hashing: Similarity preserving algorithm for entropy-based coding," in Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2010.
16. J. Wang, S. Kumar, and S.-F. Chang, "Sequential projection learning for hashing with compact codes," in Proc. Int. Conf.Machine Learning, 2010.
17. M. M. Mushrif and Y.K. Dubey, "Texture Classification Using Cosine-modulated Wavelets," International Journal of Computer and Electrical Engineering, Vol. 4, No. 3, June 2012.
18. M. M. Mushrif and Y.K. Dubey, "Extraction of Wavelet Based Features for Classification of T2-Weighted MRI Brain Images," Signal & Image Processing: An International Journal (SIPIJ) Vol.3, No.1, February 2012.
19. G. Duan, J. Yang and Y. Yang," Content-Based Image Retrieval Research," International Conference on Physics Science and Technology, Science Direct, 2011.
20. Huang, Ying Hou, "Segmentation of Textures using PCA Fusion Based Gray-Level Co-Occurrence Matrix Features", IEEE, 2009.
21. D.K.Park, Y.S.Jeon,C.S.Won, "Efficient Use of Local Edge Histogram Descriptor", ACM Multimedia Proceedings, November 2000, pp.51-54.
22. J. Wang, S. Kumar, and S.F. Chang, "Semi-supervised hashing for large-scale search," IEEE transactions on pattern analysis and machine intelligence, Vol. 34, No. 12, December 2012.
23. T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y.T. Zheng, "Nus-Wide: A Real-World Web Image Database from National University of Singapore," Proc. ACM Conf. Image and Video Retrieval, July 2009.

Ansila Henderson is currently doing her MTech in Computer Science and Engineering at Sree Chitra Thirunal College of Engineering under University of Kerala, Trivandrum, Kerala, India. Ansila received her B Tech Degree in Computer Science and Engineering from Colege of Engineering, Adoor under CUSAT, Kerala, India in 2011. She concentrates mainly on Image Processing and Computer Vision.

Kavitha K V is working as Assistant professor at the department of computer science and engineering, Sree Chitra Thirunal College of Engineering, Trivandrum, Kerala. She did her B.Tech degree at Sree Chitra Thirunal College of engineering, Trivandrum from University of Kerala. She did her M.Tech degree at College of Engineering, Trivandrum from University of Kerala. Now she is also doing research in Machine Learning. She published her research works in many international journals.