Clustering Web Images with Multi-modal Features

Manjeet Rege Ming Dong
Machine Vision Pattern Recognition Lab
Jing Hua
Graphics and Imaging Lab
Department of Computer Science, Wayne State University
Detroit, MI 48202, USA
{rege, mdong, jinghua}@wayne.edu

ABSTRACT

Web image clustering has drawn significant attention in the research community recently. However, not much work has been done in using multi-modal information for clustering Web images. In this paper, we address the problem of Web image clustering by simultaneous integration of visual and textual features from a graph partitioning perspective. In particular, we modelled visual features, images, and words from the surrounding text of the images using a tripartite graph. This graph is actually considered as a fusion of two bipartite graphs that are partitioned simultaneously by the proposed Consistent Isoperimetric High-order Co-clustering (CIHC) framework. Although a similar approach has been adopted before, the main contribution of this work lies in the computational efficiency, quality in Web image clustering and scalability to large image repositories that CIHC is able to achieve. We demonstrate this through experimental results performed on real Web images.

Categories and Subject Descriptors

I.5.3 [**Pattern Recognition**]: Clustering-*algorithms*.

General Terms

Algorithms, Performance, Experimentation.

1. INTRODUCTION

With the explosive growth of Web and the recent development in digital media technology, the number of images on the Web has grown tremendously. Consequently, Web image clustering has emerged as an important application. For example, properly grouped Web images can provide a very neat bird's eye view of the retrieved images to users.

The initial efforts on image clustering were solely based on low-level visual features of images and hence prone to the semantic gap problem. A low-tech and a naive solution adopted by search engines to overcome this problem to a certain extent has been to treat image clustering as a text clustering problem. Web images are represented using textual features in terms of the surrounding texts and captions. Images clustered based on these textual features are then retrieved accordingly. But, since images are not actually text documents, this approach is hardly a solution to the problem at hand. A better approach is to use multi-modal features that incorporate both visual and textual features together. [1] and [2] combined the textual and visual features into a global vector by performing Latent Semantic Indexing. In both these works, the two different kinds of image representations were simply combined together in a rather rigid way without any theoretical basis. [3] first clustered images into different semantic groups by employing the textual and link features and then performed visual feature-based clustering in each semantic group. A problem in this two-step process is that an erroneous first step results can propagate to the second step leading to poor image clustering. [4] applied an iterative algorithm to combine the co-clustering between images and surrounding text and the one-sided clustering of images based on visual features. The convergence property of this algorithm was not proven and the kind of combination is unsymmetrical according to the status of visual and textual features. From the above discussion, it is clear that the earlier efforts on using multi-modal features for Web image clustering were along the lines of combining the information from visual and textual features instead of integrating them together synchronously under a sound theoretical framework.

In this paper, we propose the Consistent Isoperimetric High-Order Co-clustering (CIHC) framework for simultaneous integration of visual and textual features for efficient Web image clustering under a graph theoretical approach. Specifically, visual features, images and textual features are modelled using a tripartite graph, as shown in Figure 1. This

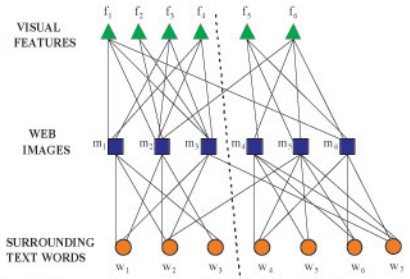

Figure 1: Tripartite graph of visual features, Web images and surrounding words of the images.

tripartite graph is treated as two bipartite graphs of visual features & images and that of images & textual features from the surrounding texts of the image. Co-clustering is then achieved by simultaneously partitioning these two bipartite graphs together such that the information from both the kinds of features is optimally utilized. Note that the simultaneous partitioning of the two bipartite graphs is performed in such a way that the local clustering of each graph need not be optimal under the constraint that the fusion of the two results yields optimum image clustering. Actually, a similar concept was presented by [5] where the Consistent Bipartite Graph Co-partitioning (CBGC) was proposed where the two bipartite graphs were partitioned using spectral co-clustering [6]. An iterative algorithm using semi-definite programming (SDP) [7] was used to partition the tripartite graph which is computationally expensive and does not work well on large data sets. On the other hand, the proposed methodology requires a simple solution to a sparse system of overdetermined linear equations. Moreover, in the CIHC framework, we partition the two bipartite graphs simultaneously using Isoperimetric Co-clustering Algorithm (ICA) which has been shown to achieve superior results than the spectral approach in terms of the quality, efficiency and stability of the partition [8, 9]. Experimental results performed on images extracted from real Websites demonstrate the advantages of CIHC over CBGC in clustering Web images.

2. CIHC FRAMEWORK FOR WEB IMAGE CLUSTERING

The tripartite graph $G = \{F, M, W, E\}$, has three sets of vertices, viz., F, M and W representing visual features, images and words, respectively with E being the set of edges. Let $\mathbf{A}$ and $\mathbf{B}$ represent the weight matrices for *feature-image* and *image-word* bipartite graphs respectively. Every entry in $\mathbf{A}$ and $\mathbf{B}$ represents the importance of a particular feature and word for that image, respectively. For $\mathbf{B}$, word frequency in the surrounding text of the images is used.

We partition the tripartite graph by applying ICA to the two bipartite graphs: *feature-image* and *image-word*. ICA has been motivated from the combinatorial formulation of the classic isoperimetric problem [10, 11, 8]: *For a fixed area, find the shape with minimum perimeter*. It provides polynomial time heuristic for the NP-hard problem of finding a region with minimum perimeter for a fixed area. Let $V = \{M \bigcup W\}$ be the set of vertices of the *image-word* bipartite graph. ICA partitions V into sets S and S^c, such that $S \bigcup S^c = V$ and $S \bigcap S^c = \phi$. Like other graph partitioning algorithms, ICA achieves optimum partitioning by finding S and S^c so that isoperimetric ratio of the graph h_G defined as,

$$h_G = \frac{|\Delta S|}{Vol_S} \tag{1}$$

is minimized. The numerator and denominator represent the boundary area and the volume of S, respectively. The boundary of S is defined as,
$\Delta S = \{e_{ij} |$ edges between a vertex in S and $S^c\}$.
Consequently,

$$|\triangle S| = \sum_{e_{ij} \in \triangle S} w(e_{ij}) \tag{2}$$

where $w(.)$ represents the edge weight. The combinatorial volume can be defined as,

$$Vol_S = |S| \tag{3}$$

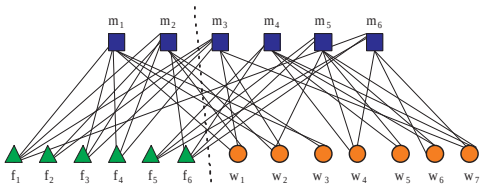

Figure 2: Traditional extension of bipartite graph partitioning algorithm ICA is unable to partition the *visual feature-image-word* tripartite graph.

After a few mathematical deductions, ICA achieves the minimization of equation (1) by solving a sparse system of linear equations as,

$$\mathbf{L} \begin{bmatrix} \mathbf{m} \\ \mathbf{w} \end{bmatrix} = \mathbf{e} \tag{4}$$

where $\mathbf{L}$ is the Laplacian matrix of the *image-word* bipartite graph, $[\mathbf{m}\ \mathbf{w}]^T$ is the indicator vector to indicate partitioning of the two types of vertices, and $\mathbf{e}$ is a vector of ones of size $|M| + |W|$. Solving this system of equations results in a real valued $[\mathbf{m}\ \mathbf{w}]^T$. In order to get partitions, this solution is *cut* using a *splitting value* s into the set of vertices having indicator vector value $\leq s$ and $> s$.

To partition the *visual feature-image-word* tripartite graph, intuitively it might seem obvious to perform traditional extension of ICA by solving a similar system of linear equations as,

$$\mathbf{L} \begin{bmatrix} \mathbf{f} \\ \mathbf{m} \\ \mathbf{w} \end{bmatrix} = \mathbf{e} \tag{5}$$

where $\mathbf{L}$, the indicator vector $[\mathbf{f}\ \mathbf{m}\ \mathbf{w}]^T$ and $\mathbf{e}$ are similarly defined for the tripartite graph. However, by doing so the *visual feature-image-word* tripartite graph actually ends up being a bipartite graph of images and visual features & words. This can be seen by shifting the *visual feature* vertices on to the side of the *word* vertices as illustrated in Figure 2. Due to this, we will be unable to distinguish between cutting a *visual feature-image* edge and an *image-word* edge and is thus a conceptual misrepresentation of the basic structure of visual features, images and words in Figure 1.

To overcome the ill-partitioning of Figure 2, we partition the *visual feature-image-word* tripartite graph by considering it as two bipartite graphs coupled together. It is easy to see that applying ICA separately on the two bipartite graphs will result in two different partitioning results on the images. In order to achieve consistent results, we need to partition the two bipartite graphs simultaneously. That is, *visual feature-image* bipartite graph needs to be partitioned under the constraints enforced on images by words while, the partitioning of *image-words* bipartite graph has to be under the constraints enforced on images by the visual features. In other words, we achieve consistent partitioning of images under the constraints that the partitioning of *visual feature-image* or *image-word* need not be optimal. By doing so, we consistently integrate the visual and textual features simultaneously for clustering the Web images.

Applying ICA to the *visual feature-image* bipartite graph, we get,

$$\mathbf{L}^{(fm)} \begin{bmatrix} \mathbf{f} \\ \mathbf{m} \end{bmatrix} = \mathbf{e}^{(fm)} \tag{6}$$

Similarly, *image-word* bipartite graph yields us,

$$\mathbf{L}^{(mw)} \begin{bmatrix} \mathbf{m} \\ \mathbf{w} \end{bmatrix} = \mathbf{e}^{(mw)} \tag{7}$$

Table 1: Image categories used in the experiments

Category Name	Size	Category Name	Size
Owls	71	Snow Mountains	82
Flowers	64	Flying Eagle	67
Lions	56	Dusk	58
Elephants	85	Plants	79
Horses	76	Railways	77

We combine the above two system of linear equations as,

$$\begin{bmatrix} \mathbf{L}^{(fm)} & \mathbf{0} \\ \mathbf{0} & \mathbf{L}^{(mw)} \end{bmatrix} \begin{bmatrix} \mathbf{f} \\ \mathbf{m} \\ \mathbf{w} \end{bmatrix} = \begin{bmatrix} \mathbf{e}^{(fm)} \\ \mathbf{e}^{(mw)} \end{bmatrix} \quad (8)$$

$$\mathbf{F}\,\mathbf{r} = \mathbf{v}$$

where $\mathbf{v}$ is a vector of ones of size $|F| + 2|M| + |W|$. Note that $\mathbf{F}$ is not a square matrix, i.e. this is an overdetermined system of linear equations where the number of equations is more than the number of variables, which needs to be solved using a least squares method [12]. Due to its simplicity and efficiency, we used the QR decomposition method to solve equation (8). However, any other method can be employed as well. Amongst the common methods for *cutting* the indicator vector $\mathbf{r}$ are the median cut and the ratio cut. Median cut uses the median of the indicator vector $\mathbf{r}$ as the *splitting value* to produce equally sized partitions while ratio cut chooses one such that the resulting partitions have the lowest isoperimetric ratio indicating optimal partitioning. As our goal is not to necessarily produce equally sized clusters, we employ the ratio cut.

The main steps of CIHC can be summarized as follows:

1. Using weight matrix $\mathbf{A}$ of the *visual feature-image* bipartite graph, construct the Laplacian matrix $\mathbf{L}^{(fm)}$.
2. Similarly, using weight matrix $\mathbf{B}$ of the *image-word* bipartite graph, construct $\mathbf{L}^{(mw)}$.
3. Using equation (8), construct $\mathbf{F}$, $\mathbf{r}$ and $\mathbf{v}$, and solve the system of linear equations $\mathbf{F}\,\mathbf{r} = \mathbf{v}$.
4. Employ ratio cut on $\mathbf{r}$ to get the partitions.

3. EXPERIMENTS AND RESULTS

For dataset preparation we followed the same approach as in [5]. A web crawler was first sent out to crawl some of the Webpages in the Yahoo directory[1] to extract images and their surrounding texts. The image database consists of 15,000 images which have been manually assigned to 42 categories. To conduct the experiments, we randomly selected 10 of these categories shown in Table 1. The words left over after removal of stop words were considered to be the textual features of the image. For the visual features, we used the PCA-SIFT image descriptors [13].

We have compared CIHC with CBGC in terms of the image clustering results, scalability and computational speed. CBGC is a very parameter-dependent framework which relies heavily on the values of θ_1, θ_2 and β. β is the weighting parameter that decides which bipartite graph is given more emphasis in clustering. θ_1 and θ_2 are parameters used to put constraints on the SDP bound controllers. To decide on the values for these 3 parameters, we followed the approach suggested by the authors [5]. We randomly selected two image categories (*Elephants* and *SnowMoutains*) and varied the values of all the parameters between 0 and 1 for the partitioning. Values that gave best clustering result were chosen and used for the rest of the categories. The problem with CBGC is that these values are category specific and have to be retuned for the other categories. This is demonstrated in the results in Figures 3 (a)-(d). The vertical dotted line separates the image categories indexed on the X-axis. Embedding value in the indicator vector is plotted along the Y-axis. The two pattern plots ($*$ and $\triangle$) represent the two clusters obtained. Figures 3(a) and (b) show the performance of CBGC and CIHC in partitioning *Elephants* and *SnowMoutains*. Based on the parameter values set, CBGC is able to get perfect clustering results. CIHC also gets similar results. In another case involving *Flying Eagle* and *Lions* shown in Figures 3(c) and (d), CBGC has misclassified a number of *Lions* images. CIHC on the other hand was able to generate very good clusters. We observed similar results in the clustering of other categories. Hence, this approach for tuning the CBGC parameters works on only those few categories and performs poorly on the other datasets. In the real world application for Web image clustering, it is impossible to know what parameter values are to be chosen for clustering the images retrieved. On the other hand, CIHC is a completely parameter-less approach and does not require *a priori* specification of any parameters.

To evaluate the image clustering performance of CIHC and CBGC across all the categories, we have used the cross-accuracy metric [5]. If two image categories having n_1 and n_2 images respectively are mixed, then the ground truth Boolean vector $\mathbf{rt}$ can be written as, $\mathbf{rt}$ = (1,1,...,1,0,0,...,0) where the first n_1 elements are set to 1 and the rest n_2 elements are set to 0. The image clustering results can be represented as a Boolean vector $\mathbf{rc}$ having the same ordering of elements as $\mathbf{rt}$. Cross-accuracy is defined as follows,

$$accuracy = \max\left\{ \frac{\sum_i (rt_i \oplus rc_i)}{n_1+n_2}, 1 - \frac{\sum_i (rt_i \oplus rc_i)}{n_1+n_2} \right\}$$

where $\oplus$ represents the exclusive-OR operation. We mixed every image category with the rest of the category and measured the accuracy of the clustering. In Figure 3(e), we have plotted the accuracy of CIHC (Y-axis) vs CBGC (X-axis). Each circle in the plot represents a possible image category pair. It can be seen that most of the circles fall in the upper part of the diagonal. This indicates that in the clustering of most of the image category pairs, CIHC outperforms CBGC. The few circles on the lower part of the diagonal are the category pairs for which CBGC has been properly tuned with the parameter values. In Table 2, we show the mean accuracy between each category and all other categories for both the algorithms. CIHC has a higher mean accuracy than CBGC on all the categories.

We now compare the computational speed of CIHC with CBGC. Since the time required to cut the indicator vector is the same for both algorithms, we compare on the basis of the time required to calculate the indicator vector. The algorithms were implemented using MATLAB on a machine with a 3 GHz Intel P4 processor with 1 GB RAM. In Figure 3(f), we plot the time required by the algorithms as the number of vertices in the fully connected tripartite graph increases. Time for CIHC gradually increases with the number of vertices. For the maximum number of vertices we increased to,

[1]http://dir.yahoo.com/Arts/Visual_Arts/Photography/

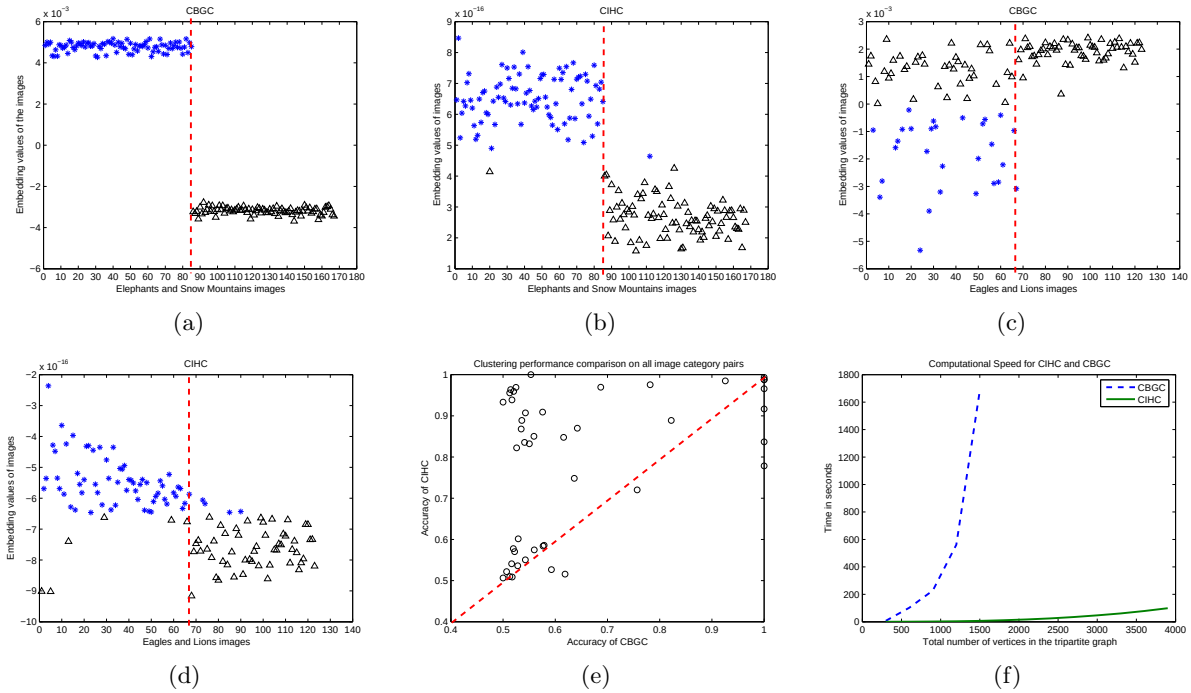

Figure 3: (a)&(b) In the clustering of *Elephants* and *Snow Moutains*, CBGC is able to get perfect clustering due to the fact that the values used for the parameters β, θ_1 and θ_2 are suitable for separating these two categories. CIHC is devoid of any parameters and is able to get comparable results on the same dataset. (c)&(d) Parameters previously set for CBGC are unable to separate *Flying Eagles* and *Lions*. A number of *Lions* images are misclassified. On the other hand, CIHC performs well. (e) Web image clustering comparison of CBGC and CIHC on all image category pairs. (f) Computational speed comparison of CIHC with CBGC.

Table 2: Average clustering performance

Category Name	CBGC	CIHC
Owls	0.5898	0.8241
Flowers	0.6248	0.8342
Lions	0.6544	0.8132
Elephants	0.8706	0.8941
Horses	0.6363	0.8244
Snow Mountains	0.6050	0.7746
Flying Eagle	0.6728	0.8608
Dusk	0.5412	0.5614
Plants	0.6386	0.6543
Railways	0.6631	0.8070

about almost $4,000$, CIHC required about 98 seconds only. CBGC on the other hand, was unable to keep up with CIHC. As can be seen, the time required by CBGC really shoots up for a few hundred vertices and hence is unable to handle larger sized datasets. This experiment clearly demonstrates the computational efficiency of CIHC and the potential for applicability in large-scale real-world Web image clustering applications.

4. REFERENCES

[1] M. Cascia, S. Sethi, and S. Sclaroff. Combining textual and visual cues for content-based image retrieval on the world wide web. In *proc. of IEEE Workshop on Content-Based Access of Image and Video Libraries*, 1998.

[2] R. Zhao and W. I. Grosky. Narrowing the semantic gap - improved text-based web document retrieval using visual features. *IEEE Trans. on Multimedia*, 4(2):189–200, 2002.

[3] D. Cai, X. He, Z. Li, W-Y. Ma, and J-R. Wen. Hierarchical clustering of WWW image search results using visual, textual and link information. In *proc. of ACM Multimedia*, 2004.

[4] Z. Li, G. Xu, M. Li, W-Y. Ma, and H-J. Zhang. Grouping www image search results by novel inhomogeneous clustering method. In *proc. of MMM*, 2005.

[5] B. Gao, T-Y. Liu, T. Qin, X. Zheng, Q-S. Cheng, and W-Y. Ma. Web image clustering by consistent utilization of visual features and surrounding texts. In *proc. of ACM Multimedia*, 2005.

[6] I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *proc. of KDD*, 2001.

[7] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[8] L. Grady and E. L. Schwartz. Isoperimetric graph partitioning for image segmentation. *IEEE Trans. on PAMI*, 28(3):469–475, 2006.

[9] M. Rege, M. Dong, and F. Fotouhi. Co-clustering documents and words using bipartite isoperimetric graph partitioning. In *proc. of ICDM*, 2006.

[10] J. Dodziuk. Difference equations, isoperimetric inequality and the transience of certain random walks. *Transactions of the American Mathematical Society*, 284:787–794, 1984.

[11] L. Grady and E. L. Schwartz. Isoperimetric partitioning: A new algorithm for graph partitioning. *SIAM Journal on Scientific Computing*, 27(6):1844–1866, 2006.

[12] C. L. Lawson and R. J. Hanson. *Solving Least Squares Problems*. Soc for Industrial and Applied Math, 1995.

[13] Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *proc. of CVPR*, 2004.