VIDEO SIMILARITY DETECTION WITH VIDEO SIGNATURE CLUSTERING

Sen-ching S. Cheung and Avideh Zakhor

Department of Electrical Engineering and Computer Sciences
University of California, Berkeley, CA 94720
{cheungsc, avz}@eecs.berkeley.edu

ABSTRACT

The proliferation of video content on the web makes similarity detection an indispensable tool in web data management, searching, and navigation. We have previously proposed a compact representation of video clips, called video signature, for retrieving similar video clips in large databases. In this paper, we propose a new signature clustering algorithm to further improve retrieval performance. The algorithm treats all the signatures as an abstract threshold graph, where the threshold is determined based on local data statistics. Similar clusters are identified as highly connected regions in the graph. This algorithm outperforms simple thresholding and hierarchical clustering techniques in identifying a set of manually-determined similar clusters from a dataset of 46,356 web video clips. At 95% precision, our algorithm attains 85% recall while simple thresholding and complete-link hierarchical scheme attain 67% and 75% recall respectively. Applying our algorithm to the entire dataset, 6,900 similar clusters are identified, with an average cluster size of 2.81 video clips. The distribution of cluster sizes follows a power-law distribution, which has been shown to describe many web phenomena.

1. INTRODUCTION

With ever more popularity of video web-publishing, popular content is being mirrored, reformatted, modified and republished, resulting in excessive content duplication. An efficient algorithm to identify similar video clips can therefore be beneficial to many web retrieval applications, such as presenting uncluttered search results, and providing alternatives in the case of expired links or network outages. By similar video clips, we mean video clips with almost identical content but possibly compressed at different qualities, reformatted to different sizes and frame-rates, or undergone minor editing in either spatial or temporal domain. In [3, 4], we propose a compact and robust representation of video clips called video signature for similarity measurement. Specifically, simple thresholding on distances between signatures is used to identify similar video clips. In this paper, we investigate clustering methods to further improve retrieval performance.

A clustering structure can provide an efficient organization of data which allows users to rapidly focus on relevant information. For example, clustering is extensively used in the areas of browsing and navigation[8]. Another benefit of clustering, which will be the focus of this paper, is its potential gain in retrieval performance over simple-thresholding or ranking similarity search techniques. Assume we have a distance function that measures similarity. When presented with a query, a simple-thresholding search retrieves all items from the database that are within a certain distance threshold away from the query; alternatively, ranking search techniques retrieve a fixed number of items closest to the query. In either case, the relationship among individual data items in the database is completely ignored. On the other hand, a system that employs clustering will first group the entire database into clusters of similar items. When presented with a query, the system will return the cluster closest to the query. Even though the actual performance depends on the clustering algorithm used, by considering the totality of the data, clustering can typically reduce imprecision from the distance measurement, and discover hidden similar items. Recently, clustering techniques have been applied to multimedia information systems to improve retrieval performance[1, 6].

There is a myriad of clustering algorithms in the literature[11]. Our application of clustering signatures imposes a number of requirements on the clustering algorithm. First, the number of clusters is very large and unknown a priori. Second, the clustering algorithm should be able to adapt to local statistics in order to handle video clips with very diverse content. Third, there is a small probability for the signature distance to be large even when the two video clips are very similar[3]. Hence, the clustering algorithm must be robust enough to discard such erroneous measurements. To this end, a class of clustering algorithms based on graph-theoretical concepts seem particularly suitable[1]. This class of algorithms treats data items as vertices in a graph and connects potentially similar items with edges. The goal is to identify highly connected regions which are indicative of the presence of similar clusters. Another class of popular clustering algorithms, called the agglomerative hierarchical clustering[11], can also be used. This class of algorithms seeks a hierarchical clustering structure by successively combining clusters which are close to each other. Single-link and complete-link clusterings are the two most commonly used hierarchical algorithms. They differ from each other on how they measure the distance between two clusters. Single-link defines the distance based on the two closest elements from the two clusters, while complete-link uses the elements farthest apart.

In this paper, we propose a new graph-theoretical clustering algorithm for signatures. Our algorithm addresses threshold adaptation and signature uncertainties by considering connected regions at many different distance thresholds. The details of our algorithm are described in Section 2. Section 3 compares results of our algorithm to simple thresholding, and two hierarchical clustering schemes; statistics of the clustering structure are also presented. We conclude with discussions on future work in Section 4.

2. VIDEO SIGNATURE CLUSTERING

We begin with a brief review of video signature[3, 4]. A video signature is a compact representation of a video clip which supports a randomized algorithm in measuring video similarity. We assume that every frame in a video clip is represented by a high-dimensional feature vector with a distance metric $d_f()$. In our experiments, a quadrant HSV color histogram with l_1-metric is used. A set of M seed feature vectors $s = \{s[i], i = 1 \ldots M\}$ are first randomly generated. Then, for every video clip v in the dataset, a corresponding signature $v_s = \{v[i], i = 1 \ldots M\}$ is computed, where $v[i]$ is the frame in v most similar to the seed vector $s[i]$. We have shown previously that using M as small as nine is adequate for successfully detecting similarities in web video clips[4]. We denote the set of signatures in our dataset as $\boldsymbol{V}$. The distance between two signatures v_s and w_s is defined as

$$d(v_s, w_s) \triangleq \text{median}\{d_f(v[i], w[i]), i = 1 \ldots M\}.$$

The median operator is used to remove outliers due to possible sampling error in choosing similar signature frames. As $d()$ fails to satisfy the triangle inequality, it is not a metric in the mathematical sense. In general, the larger $d(v_s, w_s)$ is, the less likely video clips v and w are similar to each other. It is thus reasonable to assume that all pairs of similar video clips, except for those rare cases when sampling errors occur, have signature distances smaller than some small predefined $\mu > 0$.

In order to describe our clustering algorithm in a general framework, we will treat the set of signatures and the distance relationship as a *graph*. A graph $\boldsymbol{G}$ has a set of vertices $V(\boldsymbol{G})$ and a set of edges $E(\boldsymbol{G}) \subset V(\boldsymbol{G}) \times V(\boldsymbol{G})$. All edges we consider are undirected, i.e. $(u_s, v_s) = (v_s, u_s)$. In many occasions, we only consider a portion of a graph. Thus, we need the notion of a *subgraph* – $\boldsymbol{G'}$ is a subgraph of $\boldsymbol{G}$ if $V(\boldsymbol{G'}) \subset V(\boldsymbol{G})$ and $E(\boldsymbol{G'}) \subset E(\boldsymbol{G})$. If $E(\boldsymbol{G'})$ contains all the edges in $\boldsymbol{G}$ between vertices in $V(\boldsymbol{G'})$, $\boldsymbol{G'}$ is called an *induced subgraph*. In our application, each signature is a vertex in a graph. Ideally, we link two signatures with an edge if the two corresponding video clips are truly similar. However, this is unknown a priori. In fact, one of the goals of a clustering algorithm is to search all possible placement of edges so as to arrive at the most reasonable graphical structure to describe all similar video clips. Since the space of graphs with all possible edge placements is too large to search in practice, our algorithm only considers a special subset of graphs called the signature *threshold graphs*, or simply, threshold graphs. A threshold graph $P(\boldsymbol{V}, \rho)$ has a vertex set $\boldsymbol{V}$, and there is an edge between any two vertices whose corresponding signature distance is less than a fixed distance threshold ρ with $\rho \leq \mu$. We use the signature distance to represent the length of the edge.

Since the majority of the video clips on the web are not similar to each other, even the largest threshold graph considered, i.e. $P(\boldsymbol{V}, \mu)$, is likely to contain many disjoint *connected components*. A connected component, or CC, of a graph $\boldsymbol{G}$ is an induced subgraph of $\boldsymbol{G}$ in which all vertices are reachable from each other, but completely disconnected from the rest of $\boldsymbol{G}$. CCs in our signature threshold graphs are candidates for similar clusters. All signatures in a connected component $\boldsymbol{C}$ of a threshold graph $P(\boldsymbol{V}, \rho)$ are at least distance ρ away from the rest of $P(\boldsymbol{V}, \rho)$. If there is an edge between every pair of vertices in $\boldsymbol{C}$, in which case we call $\boldsymbol{C}$ a *complete* subgraph, then all signatures in $\boldsymbol{C}$ are less than distance ρ away from each other. Thus, a complete CC matches intuitively to what a similar cluster should be : all video clips in it are close to each other but far away from the rest of the dataset. In practice, full completeness is too stringent of a requirement to impose because the randomness in video signatures may erroneously amplify the distance between some similar video clips. Thus, as long as $\boldsymbol{C}$ is close to a complete subgraph, it is reasonable to assume that it represents a similar cluster. To this end, we need a metric to measure edge density of a CC. Let $|\cdot|$ denote the cardinality of a set. We notice that $\boldsymbol{C}$ has at least $|V(\boldsymbol{C})| - 1$ edges as it is connected, and at most $|V(\boldsymbol{C})| \cdot (|V(\boldsymbol{C})| - 1)/2$ edges if it is complete. We can define the *edge density* $\Gamma(\boldsymbol{C})$ as follows:

$$\Gamma(\boldsymbol{C}) = \begin{cases} \frac{|E(\boldsymbol{C})| - (|V(\boldsymbol{C})| - 1)}{|V(\boldsymbol{C})| \cdot \frac{(|V(\boldsymbol{C})| - 1)}{2} - (|V(\boldsymbol{C})| - 1)} & \text{if } |V(\boldsymbol{C})| > 2 \\ 1 & \text{otherwise,} \end{cases}$$

$\Gamma(\boldsymbol{C})$ is properly crafted such that it evaluates to 0 when $\boldsymbol{C}$ is barely connected, and to 1 when $\boldsymbol{C}$ is complete. We define a *similar cluster* to be a CC whose edge density exceeds a fixed threshold $\gamma \in (0, 1]$.

We are now ready to describe our clustering algorithm: we start by considering each connected component $\boldsymbol{C}$ in the largest threshold graph $P(\boldsymbol{V}, \mu)$. If its edge density exceeds γ, it is declared as a similar cluster and removed from the graph. Otherwise, it is likely that $\boldsymbol{C}$ contains a number of distinct similar clusters joined loosely to each other. In such case, we start trimming edges in decreasing order of their lengths, with edges of the same length removed simultaneously, until some similar clusters emerge. This is reasonable as signatures joined by longer edges are less likely to be similar. This process of edge trimming is equivalent to lowering the distance threshold ρ until the graph is partitioned into multiple CCs. We then compute the edge density for each newly-formed CC to see if it is a similar cluster. This process of checking edge density and lowering threshold is repeated until we exhaust all CCs.

The key step of the above algorithm is to find the appropriate distance threshold to partition a connected component $\boldsymbol{C}$ once we find that it is not a similar cluster. A naive approach is to first sort all the edges based on their lengths, and then remove them in decreasing order until $\boldsymbol{C}$ becomes disconnected. The desired distance threshold is then the length of the last edge removed. The drawback of this approach is that we need to check connectedness after removing every edge. A simpler method is to make use of a *minimum spanning tree* or MST. A MST of a connected graph is a subgraph that connects all vertices with the least sum of edge lengths. It can be shown that the length of the longest edge in the MST, $\boldsymbol{T}$, of $\boldsymbol{C}$ is the correct distance threshold to partition $\boldsymbol{C}$[5]. After partitioning $\boldsymbol{C}$ into CCs, the subtree $\boldsymbol{T'}$ in each newly-formed connected component, $\boldsymbol{C'}$, is a MST for $\boldsymbol{C'}$. Thus, if we need to further partition $\boldsymbol{C'}$, we can set the distance threshold to the length of the longest edge of $\boldsymbol{T'}$. In other words, all the distance thresholds used in our clustering procedure are restricted only to the length of the edges of the MST. This approach is computationally less complex than the naive approach of examining every edge and checking for connectedness.

The implementation of our clustering algorithm consists of two phases: MST construction and cluster formation. In the first phase, we construct the MST based on a modified version of the popular Kruskal algorithm[5]. In Kruskal algorithm, the MST of a graph is constructed in two steps. First, all the edges in the graph are sorted in increasing length. Second, the MST is incrementally constructed by testing if each edge can be included in the tree. We modify the second step to also compute the edge densities of the CCs attached to the two nodes of every newly-added edge to the MST. This computation can be easily done by keeping track of the number of all the nodes and edges examined so far. This extra step does not change the order of complexity of Kruskal algorithm, which is $O(e \log e)$ where $e = |E(P(\boldsymbol{V}, \mu))|$.

In the second phase of our algorithm, we identify similar clusters by repeatedly setting the thresholds to the length of the edges of the MST, and checking the pre-computed edge density for each newly-formed CC. For each threshold tested, all the information required to identify similar clusters is pre-computed and the complexity is simply $O(1)$. Thus, the computational complexity of the second phase is $O(|\boldsymbol{V}|)$, the same order as the maximum number of edges in the MST. The combined complexity of the two phases in the implementation is thus $O(e \log e) + O(|\boldsymbol{V}|) \approx O(e \log e)$.

3. EXPERIMENTAL RESULTS

In this section, we first describe our test dataset, and then compare the retrieval performance of our proposed algorithm with simple thresholding, single-link, and complete-link clustering. Statistics of the clustering structure will also be presented.

3.1. Test dataset and Ground-truth

In all our experiments, the retrieval performance is measured by how well an algorithm can identify a set of manually-selected similar video clusters, or the ground-truth set, in a large test dataset. The test dataset is a collection of 46,356 video clips, crawled from the web between August and December, 1999[3]. We expand our prior effort on constructing the ground-truth[4] and adopt a statistical approach to identifying a number of similar clusters from the test dataset. This approach is similar to the pooling method[7] used in establishing ground-truth for text retrieval systems.

The basic idea of pooling is to send the same queries to different automatic retrieval systems, whose top-ranked results are pooled together and examined by human experts to identify the truly relevant ones. For our system, the first step is to use meta-data terms to identify the initial ground-truth clusters. Meta-data terms are extracted from the URL addresses and other auxiliary information for each video in the test dataset [4]. All video clips containing at least one of the top one thousand most frequently used meta-data terms are manually examined and grouped into similar clusters. Clusters which are significantly larger than others are removed to prevent bias. The remaining 107 clusters form our initial ground-truth clusters.

However, there might still be some video clips in the test dataset that are similar to the ground-truth clusters. We first examine all video clips in the dataset that share at least one meta-data term with the video clips already in the ground-truth set. Any additional similar video clips are added to the clusters. Afterwards, a video signature database is constructed for the test dataset with one hundred frames per signature. This provides a robust retrieval system as the signature size is far larger than the nine signature frame used in our system for testing. For each video in the ground-truth set, the closest one hundred video clips in the video signature database are manually examined. Again, any additional similar video clips are added to the ground-truth set. As a result, we obtain a ground-truth set consisting of 443 video clips in 107 clusters. The cluster size ranges from two to twenty, with average size equal to 4.1.

3.2. Retrieval Performance

The performance metrics used in our experiments are recall and precision. Their definitions are described below. Let $\boldsymbol{S}$ denote the ground-truth set. $\boldsymbol{S}$ is composed of $N = 107$ non-overlapping clusters $\boldsymbol{C_i}, i = 1 \ldots N$ with $\cup_{i=1}^{N} \boldsymbol{C_i} = \boldsymbol{S}$. For any video $q \in \boldsymbol{S}$, we define the relevant set to q as $\text{rel}(q) = \boldsymbol{C_i} \setminus \{q\}$ where $q \in \boldsymbol{C_i}$ and $\setminus$ denotes set exclusion. For a retrieval algorithm, we define the return set of q, $\text{ret}(q, \lambda)$, to be the set of video clips returned by the algorithm when presenting q as a query. λ is the set of parameter(s) used by the algorithm. The precise definition of $\text{ret}(q, \lambda)$ depends on the details of the algorithm. Four algorithms are tested: simple thresholding used in our previous work[3, 4], our proposed clustering algorithm, and two hierarchical clustering schemes: single-link and complete-link[11]. For clustering algorithms where signatures are grouped into clusters, $\text{ret}(q, \lambda)$ is the set of the video clips in the returned cluster, except for q itself. For simple thresholding, $\text{ret}(q, \lambda) = \{x | d(x_s, q_s) < \lambda\} \setminus \{q\}$. Using the notions of relevant and return sets, we define recall and precision of an algorithm as follows:

$$\begin{aligned} \text{Recall}(\lambda) &\triangleq \frac{\sum_{q \in \boldsymbol{S}} |\text{rel}(q) \cap \text{ret}(q, \lambda)|}{\sum_{q \in \boldsymbol{S}} |\text{rel}(q)|} \\ \text{Precision}(\lambda) &\triangleq \frac{\sum_{q \in \boldsymbol{S}} |\text{rel}(q) \cap \text{ret}(q, \lambda)|}{\sum_{q \in \boldsymbol{S}} |\text{ret}(q, \lambda)|}. \end{aligned}$$

Thus, recall computes the fraction of all relevant video clips in the dataset that are returned by the algorithm. Precision measures the fraction returned by the algorithm that are relevant.

In our experiments, we consider pairwise signature distances up to $\mu = 4.0$, which is half of the maximum possible distance. For our dataset, around 1.2 million pairs of signatures satisfy this criterion. This number is significantly less than the total number of possible pairs which is $\binom{46356}{2} \approx 1$ billion. Each algorithm is tested with a range of parameters to demonstrate the whole spectrum of precision/recall trade-off. For the two hierarchical schemes and simple thresholding, the parameter used is a distance threshold. For a given query, more video clips will be classified as similar as the distance threshold increases. Thus, the recall value typically increases with the distance threshold. The parameter used by our proposed algorithm is the edge density threshold γ. The recall value increases as γ is reduced since larger clusters can be formed.

Figure 1(a) shows precision versus recall for the four algorithms. The single-link algorithm results in the worst performance. The reason is that as the distance threshold exceeds a certain limit, long chains of non-similar video clips are erroneously identified as clusters significantly lowering the precision. Simple thresholding results in better performance but the single threshold used is inadequate to cater to similar video clips at different distances. The complete-link algorithm performs quite well at low recall values. Unlike single-link clustering, the complete-link algorithm guarantees that all the video clips in a cluster are within the distance threshold. Thus, the clustering is more reliable leading to higher precision values. As the threshold increases to capture similar video clips that are far apart from each other, the algorithm simultaneously groups some non-similar video clips together which are relatively close to each other. This situation occurs because a single distance threshold is used in forming clusters. As a result, the precision drops as the threshold becomes too large. The curve terminates at 80% precision and 80% recall where the largest possible threshold of 4.0 is used. Our proposed algorithm provides the best performance among all the four schemes and achieves its peak performance of 85% recall and 95% precision with $\gamma = 0.3$. The precision and recall stay around the same level until γ decreases beyond 0.03. Precision then starts to drop as large loosely-connected regions are identified as clusters.

In terms of computational complexity, both simple thresholding and single-link clustering take $O(e)$ to complete – simple thresholding sequentially examines each edge while single-link identifies all CCs as clusters. Our implementation of the complete-link algorithm runs at $O(|\boldsymbol{V}|^3)$ but there are many $O(|\boldsymbol{V}|^2)$ implementations [9]. Since the number of edges in a graph, e, is always upper-bounded by $|\boldsymbol{V}|^2$, it might seem that the complexity of our proposed algorithm, $O(e \log e)$, is larger than that of the others. However in practice, the largest threshold graph, $P(\boldsymbol{V}, \mu)$, is quite sparse, i.e., $e \ll |\boldsymbol{V}|^2$, making it difficult to compare the relative complexities of the algorithms. Furthermore, there exist faster MST algorithms that run close to $O(e)$[10], and as such, we are currently investigating the feasibility of combining them with our clustering step.

Using $\gamma = 0.3$ for clustering our entire dataset, a non-optimized implementation with Perl and BerkeleyDB for memory management takes roughly 9 hours to complete on a Sun Ultra60 running SunOS 5.7. This does not include the time required for computing all the signature distances. A total of 6,900 clusters are produced and the average cluster size is 2.81. In other words, about 42% of the video clips have at least one similar clip. Figure 1(b) shows the distribution of cluster sizes. After examining some of the clusters, we notice that many of them consist of video clips from the same web-pages, and many of them are the same clips at different bitrates and formats; others are system messages from the same server. These similar clips are generated intentionally by the content creators and provide little information on how video clips are copied and modified by different web users. They constitute about half of the 42% that have at least one similar clip. On the other hand, for those clusters with video clips coming from diverse locations, the largest few are particularly interesting as they reflect the popularity of the corresponding video clips. The largest four are labeled in Figure 1(b). A small number of video clips in these clusters are mis-classified and the error percentages are shown in parentheses.

We also observe that the distribution of the cluster sizes can be well-modeled by a family of distributions called the *power-law* distributions. In a power-law distribution, $\text{Probability}(x) \propto 1/x^i$ for some positive exponent i. The best power-law fit of our distribution is shown in the log-log plot of Figure 1(c) with $i = 2.39$. It is experimentally demonstrated that many web phenomena, such as the degrees of hyper-linkage of web pages, access statistics for web pages and web surfing patterns, can be modeled using the power-law[2]. Interestingly, our estimated exponent is quite close to those estimated in other web phenomena. This is a good indication that the clusters produced by our algorithm have captured the underlying pattern of similar video clips on the web.

4. FUTURE WORK

We are currently exploring a number of issues regarding the clustering algorithm. As a web database is constantly being updated, it is inconceivable to cluster for every single change made to the database. We are currently investigating dynamic update algorithms which maintain consistent clustering structure. In addition, we are constructing statistical models to incorporate the measured uncertainties from the video signatures into the clustering algorithm.

5. REFERENCES

[1] S. Aksoy, R.M. Haralick, "Graph-theoretic clustering for image grouping and retrieval," *Proc. IEEE CVPR.*, vol. 1, p. 63–8, June 1999.

[2] A.Z. Broder, et al., "Graph Structure in the web," *9th Inter. WWW Conf.*, May 2000.

[3] S.-C. Cheung, A. Zakhor, "Estimation of web video multiplicity," *Proc. SPIE*, vol. 3964, p. 34–6, Jan. 2000.

[4] S.-C. Cheung, A. Zakhor, "Efficient video similarity measurement and search," *ICIP 2000*, vol. I, p. 85–9, Sept. 2000.

[5] T. Cormen, et al., *Introduction to Algorithms*, the MIT Press, 1992.

[6] G. Iyengar, A.B. Lippman, "Distributional clustering for efficient content-based retrieval of images and video," *ICIP 2000*, vol. I, p. 81–4, Sept. 2000.

[7] K.S. Jones, C. van Rijsbergen, "Report on the need for and provision of an "ideal" information retrieval test collection," *British Library Research and Development Report 5266*, 1975.

[8] S. Krishnamachari, M. Abdel-Mottaleb, "Image browsing using hierarchical clustering," *Proc. IEEE Int. Sym. on Comp. and Comm.*, p. 301–7, July 1999.

[9] F. Murtagh, "Comments on "Parallel Algorithms for Hierarchical Clustering and Cluster Validity," IEEE Trans. on PAMI., vol. 14, no. 10, p. 1056–8, October 1992.

[10] S. Pettie, V. Ramachandran, "An Optimal Minimum Spanning Tree Algorithm," Proc. 27th Int. Colloq. on Automata, Languages and Programming, LNCS Volume 1853, p. 49-60, July 2000.

[11] S. Theodoridis, K. Koutroumbas, *Pattern Recognition*, Academic Press, 1999.

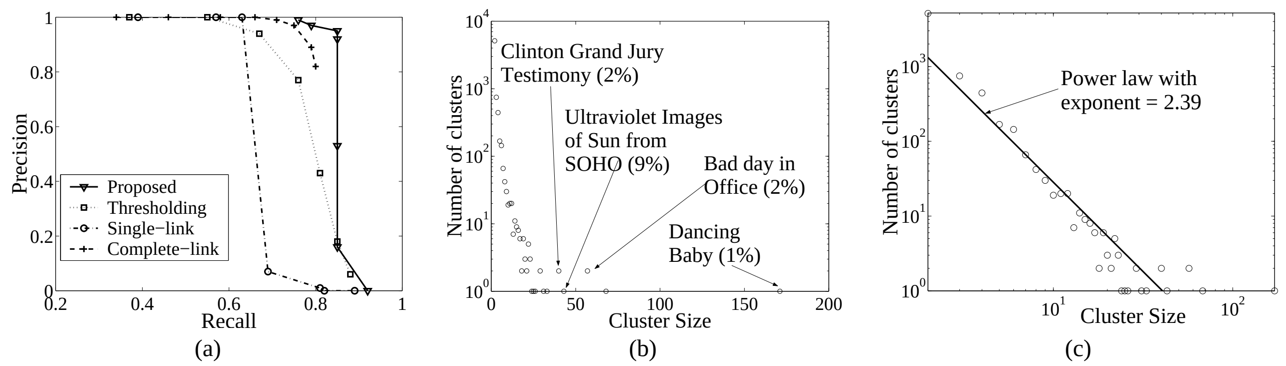

Fig. 1. *(a) Precision versus recall for different clustering algorithms and simple thresholding. (b) Distribution of cluster sizes in log-linear scale, with the four largest clusters of video clips from diverse locations labeled. (c) Distribution of cluster sizes in log-log scale with the best power-law fit.*