Making a Scene: Alignment of Complete Sets of Clips Based on Pairwise Audio Match

Kai Su
Rutgers University
kais@rutgers.edu

Mor Naaman
Rutgers University
mor@rutgers.edu

Avadhut Gurjar
Rutgers University
aag137@eden.rutgers.edu

Mohsin Patel
Rutgers University
map433@eden.rutgers.edu

Daniel P. W. Ellis
Columbia University
dpwe@ee.columbia.edu

ABSTRACT

As the amount of social video content captured at physical-world events, and shared online, is rapidly increasing, there is a growing need for robust methods for organization and presentation of the captured content. In this work, we significantly extend prior work that examined automatic detection of videos from events that were captured at the same time, i.e. "overlapping". We go beyond finding pairwise matches between video clips and describe the construction of scenes, or sets of multiple overlapping videos, each scene presenting a coherent moment in the event. We test multiple strategies for scene construction, using a greedy algorithm to create a mapping of videos into scenes, and a clustering refinement step to increase the precision of each scene. We evaluate the strategies in multiple settings and show that a greedy and clustering approach results in best possible balance between recall and precision for all settings.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

algorithms, human factors

Keywords

audio fingerprinting, video, social media, synchronization

1. INTRODUCTION

Increasingly, video content is captured by individuals attending or witnessing live events, from a MGMT music concert to Arab Spring demonstrations; from a New York City hurricane to the Chinese New Year's parade in San Francisco. These immense volumes of social video content generated around events can serve multiple purposes, including allowing individuals to review, re-live and share the experience. Unfortunately, the social content is fragmented and scattered across the Web, with no service that generates a coherent consumption experience for this content. We build on previous work [8] to study the automatic detection of event scenes, each including multiple overlapping videos. The scene structure can support better viewing, understanding and sensemaking for the event using social video content. A robust scene structure generation will help build on the information social videos carry about an event, to provide for better organization of information, improved playback, redundancy removal, highlighting of key content, generation of summaries, and more [8, 13].

Since reliable temporal metadata is not available for videos, finding overlapping videos and synchronizing them is a difficult task, even when starting with a set of videos taken at a single event [8]. Past work had used audio fingerprinting to find overlapping and synchronized social videos [8, 12] within videos taken at an event. As Kennedy and Naaman show [8], pairwise matching is often noisy and inaccurate, demonstrating low recall-precision tradeoff – about 30% pairwise recall for acceptable levels of precisions. However, that work only studied pairwise matching and synchronizing of clips. In other words, the research examined all possible pairs of clips and whether or not they match, and are detected correctly, ignoring the information available in transitive overlap data: we can leverage knowledge about the overlap of video clips (A,B) and video clips (B,C), to reason about the overlap of clips A and C.

Matching and clustering video data is related, but distinct, from applications to other documents such as text and images [4, 7]. Whereas the matching of local text shingles or image interest points can often lead to an unambiguous association between documents, the wide range in soundtrack quality between videos of the same event mean that there are always ambiguous cases where it is impossible to decide if a particular pair, taken alone, constitute a match. In this work, we use clustering to leverage the transitive similarity within a group of matches to improve over local, pairwise decisions. Secondly, the temporal dimension of videos mean that it is important to distinguish between the case where two videos show weak similarity because they are not related, as opposed to a low count of matching features simply because they have a small proportion of time overlap – even if the similarity within that overlap is strong. The existence of edited videos, whose timebase does not preserve a uniform

ICMR '12, June 5-8, Hong Kong, China

alignment to other versions, exacerbates this issue.

In this work, we go beyond pairwise match, to study the performance of scene-generation algorithms that automatically create a complete scene structure from pairwise match scores of a set of clips. We use the pairwise audio fingerprinting-based match score between each pair of clips to generate an initial mapping of the event videos into scenes (i.e. sets of clips when each clip overlaps with at least one other in the set). We test various settings for generating the scene mapping, and investigate post-mapping clustering approach for refining the resultant scenes, to improve the precision of the system (proportion of true overlapping clips that are correctly detected as overlapping) while keeping high levels of recall (proportion of clips that are identified as overlapping out of the complete set of true overlapping clips).

2. BACKGROUND AND RELATED WORK

Much work has been done around events, in particular (but not only) music events. For example, research addressed retrieving videos (and other media) taken at a given event [3, 9], which could serve as input for our system. Other efforts looked at classifying and analyzing event content (e.g., [14]). Recent work showed how the output of video synchronization can be used for auto-summarization and presentation of content [13, 17] or even 3-D rendering of multi-camera sources [2].

In this section we focus on work that serves as technical foundations for our study. We describe the process of generating audio fingerprints for video clips, and the process of finding the strength of the match between two potentially-overlapping clips. We build on this previous work, and expand on it in the next sections, after highlighting the potential issues and complications with current approaches.

2.1 Generating Fingerprints

We compute audio fingerprints for audio segments by taking the short-time Fourier transform of a given audio segment and identifying "landmarks", as proposed by Wang [16], defined to be pairs of the onsets of local frequency peaks. Each landmark consists of two adjacent onsets' frequency and time value, and is hashed into a fixed-length value of a 3-tuple: 1st onset frequency, time difference, and frequency difference.

We follow the settings and parameters described in [6] in this work, including the frequency and time binning, and the size of hashes. In our work, the frequency, the time difference and the frequency difference are quantized into 8, 6 and 6 bits, respectively. Therefore, each hash value consists of a concatenation of these three parameters, yielding a 20-bit hash with 1,048,576 possible values.

The set of fingerprints hashes for clip i is defined as H_i, with each single $h_{i,t} \in H_i$ is associated with a time offset t from the beginning of the video clip.

2.2 Synchronizing Clips

The result of the fingerprinting process is a large set of time-stamped fingerprints for each clip i, where the fingerprints are simply a set of timestamped hash values, H_i. The synchronization process determines whether a pair i, j of clips are likely to include a recording of the same audio source, and what is the difference in time between the two recordings. We perform this detection task by finding all occurrences of matching hash values between the two clips, as described in [8]. Since there is a fixed vocabulary for hash values, many spurious hash matches will be found between different points in the pair of clips; in other words, that exist many hash values such that $h_{i,t_n} = h_{j,t_m}$. However, two clips that are overlapping are likely to have a larger number of matching hash values occurring at identical offsets in each of the two clips. We detect a potential match between two clips using the presence of a unique offset o where the two clips demonstrate a large number of matching hash values. In other words, we are looking for a series of hash values $h_{i,n_1}, h_{i,n_2}, h_{i,n_3}, \ldots$ and $h_{j,m_1}, h_{j,m_2}, h_{j,m_3}, \ldots$, such that $n_1 - m_1 = n_2 - m_2 = n_3 - m_3 =, \ldots = o$. The size of the set of matches at maximal offset o is denoted $M_{i,j}(o)$. As in the work of Kennedy and Naaman [8], the offsets of the found hash matches are grouped into 50ms bins (representing a rate of 20 frames per second).

Next, we describe how pairwise matches are decided based on the set of matches, and discuss the limitations of the pairwise alignment information.

2.3 Challenges with Pairwise Matching

The pairwise approach leads to clear limitations, trading off precision and recall for pairwise matches according to the set threshold.

A decision about whether or not the two clips are an *initial* match is made by checking whether the match between the clips $M_{i,j}(o)$ at the maximal offset exceeds a threshold. Other options for thresholds are based on the number of matches per seconds of overlap between the clips at the maximum-match offset. Regardless, setting a high threshold on the matching score will result in a large number of false negatives (truly overlapping videos that are not recognized as overlapping). Setting a low threshold will result in many false positives (videos that do not overlap but are recognized as overlapping). In both cases, the outcome would be a less than ideal experience for a user browsing these videos: the set of videos shown is likely to either be smaller than it could be, or include spurious matches of videos that "do not belong".

Worse, the transitive nature of clip overlap can add further noise and degrade the results. For example, Figure 1 shows a hypothetical set of four clips numbered $1 \ldots 4$, and lines that represents the hash matches between the clips at the maximum offset. Further assume that clips 1 and 2 are overlapping, and so are clips 3 and 4, but the matching landmarks between clips 2 and 3 are spurious. If we use the raw value of $M_{i,j}(o)$ to decide the relationship of tracks, setting a threshold of four matches would fail to identify clips 3 and 4 as overlapping. Using a threshold of two, however, will not only wrongly identify overlap between clips 2 and 3, but will transitively, and wrongly, suggest an overlap between clips 2 and 4 as well.

3. FROM PAIRWISE MATCHES TO COMPLETE SCENES

In this section, we describe a process that creates robust groups of overlapping clips – *scenes* – given pairwise information about clip matches. Each scene will be a set of clips that captured a continuous, coherent moment in the show. Note that an underlying assumption here is that we have distinct, separate scenes captured by the audience (i.e., no

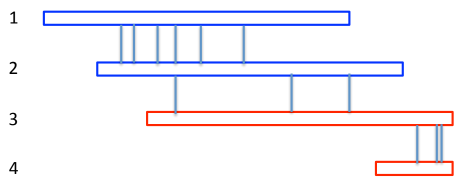


Figure 1: Potential complications of transitive overlap data

large component of videos that together cover a large portion of the event). This limitation can be controlled for by trimming videos or scenes longer than a threshold to shorter parts.

We use the co-information between pairs of matching clips to improve the match data when creating the scenes. Assume clips 1, 2, 3 in Figure 1 are each 2 minutes long. If Clip 1 initially matches Clip 2 with an offset of 20 seconds (Clip 2 starts 20 seconds after before Clip 1), and Clip 2 initially matches Clip 3 with an offset of 20 seconds, that means Clip 1 and Clip 3 should overlap with an offset of 40 seconds. In this case, given evidence that clips (1,2) match and (2,3) match, we'd expect evidence that (1,3) match as well. Existence of evidence for a (1,3) match will be consistent with the other two matches and increase our "trust" in them. Lack of such evidence might mean that one or more of these two matches is wrong.

We take a greedy approach to generate initial scenes of overlapping video clips, as described next (Section 3.1). The greedy algorithm is threshold based. Thus, setting a low threshold will generate overloaded scenes with many spurious matches (e.g., all the clips in Figure 1 will be wrongly included in one scene). Setting a low threshold will generate over-segmented scenes (e.g., Clip 3 and Clip 4 in Figure 1 will wrongly appear in different scenes).

To improve on the results of the greedy assignment, we refine the scenes generated by the greedy algorithm using a clustering step (section 3.2–3.3). We present an evaluation of the methods and their combination for different thresholds in Section 4.

3.1 Initial Overlap Computation

This first step in our process uses a greedy threshold-based algorithm to generate a complete set of scenes from a large set of pairwise matches of clips. The algorithm sorts all pairwise matches i, j according to the number of hash matches between the clips at the maximal-match offset o. Scenes are then created by iteratively adding videos from the sorted queue to scenes when a) the video matches another video(s) already in that scene, and b) the match offsets across all videos in the scene are consistent (such that no conflicting information about clip alignment is possible).

More precisely, the steps of computing this initial alignment are:

1. Compute the pairwise matches of all clips and choose for each pair of clips i, j a "best offset" o such that the number of hash matches between i and j at offset o, $M_{i,j}(o)$ is greater than all other offsets o.

2. Remove all pairwise matches (i, j) such that $M_{i,j} < threshold$.

3. Sort all remaining pairwise matches (i, j) by $M_{i,j}$ into a queue Q.

4. Use a greedy procedure to generate an initial set of overlapping clips, or scenes, as described in Algorithm 1.

Algorithm 1 Greedy algorithm used to create tentative alignment of all clips

```
Q ← sort({M_{i,j}}) a queue of pairwise matches p
𝕊 ← empty set of scenes
p ← pop(Q)
Add the match of i, j, o represented by p to a new scene 𝒮 in 𝕊
while Q not empty do
  p ← pop(Q)
  for every scene 𝒮_i in 𝕊 do
    if One of the clips of p is in 𝒮_i then
      if The offset information of p agrees with the information already in 𝒮_i then
        Add the match p to 𝒮_i
      end if
    end if
  end for
  if the clips in p are not in any scene then
    Add the match represented by p to a new scene 𝒮
  end if
end while
for 𝒮_i, 𝒮_i in 𝕊 do
  if 𝒮_i and 𝒮_i share one or more common clips then
    Merge the two scenes.
  end if
end for
```

The output of Algorithm 1 is sets of clips, or initial scenes. For each scene, we have the synchronization (offset) between all clips in the scene, and those offsets are guaranteed to be consistent. If Clip 3 is at 20 seconds offset from Clip 2, and Clip 2 is at 20 seconds offset from Clip 1, then Clip 3 would be at 40 seconds offset from Clip 1 in the algorithm's assignment. Note that in each scene, it is not required that all clips temporally overlap. Referring back to Figure 1, clips 1 and 4 could be included in the same scene because of their overlap with other clips in the scene, but even if the matching is correct, there would be no actual overlap between clips 1 and 4. Also note, as mentioned above, since the input to the algorithm is noisy, the output is also likely to include spurious matches. For example, it could be that Clip 1 and Clip 3 are overlapping in the output but do not actually capture the same moment according to the ground truth.

A simple representation for the output of Algorithm 1 is a set of scenes where each scene $\mathcal{S}_I$ (we used capital letters to mark scene indices) is a set of clips $c_{I,1}, c_{I,2}, \ldots, c_{I,m}$ and offsets $o_1, o_2, \ldots, o_m$ such that $o_i \geq 0$ capturing the start time of each clip in respect to the earliest clip in the cluster (for which $o_i = 0$). Note that these offsets, in each scene, define the offset between each pair of clips in the scene.

3.2 Scene Refinement

In this work we explore two approaches for achieving high precision as well as and high precision-recall tradeoff in scene assignments. One approach is simply setting a high threshold for the scene alignment in Algorithm 1. The other approach, detailed in the remainder of this section, is to set a low threshold for Algorithm 1, and refining the resultant scenes using a clustering step. The clustering step examines each scene computed as described above, and decides how to split it to final "scenes" that are likely to include more precise assignments. If the threshold for Algorithm 1 is low, the scenes generated will achieve high recall, perhaps at the expense of precision: the output is likely to be complete, but not accurate. In other words, if clips c_i and c_j are indeed synchronous, they are likely to be in the same scene in Algorithm 1's output. The clustering step can only further divide each scene in the output.

For each scene $\mathcal{S}_I$, we use a score $s_{i,j}$ of match likelihood at the offset decided by the greedy match for each two clips in the scene. Note that some pairs of clips in the scene may not have any overlap at all, in which case the match score between them can be assigned a special value (e.g., 0 or N/A). At the end, for each two clips in the scene, we have a score that represents the likelihood these clips to overlap at the offset suggested by Algorithm 1.

In this work, we consider two types of scores $s_{i,j}$ for the goodness of the pairwise match between clips i, j. We use these scores as described below, and evaluate the performance of the system based on the scoring function used. The first score we use is the raw number of hash matches, $s_{i,j}^{raw} = M_{i,j}$. This score is defined in respect to an offset o between the two clip as decided by Algorithm 1.

The second score we use is a direct estimate of the posterior probability that the two streams contain the same content in their region of overlap, based on the comparison between the number of matches and what would be expected for that duration of overlap. Assume that a random video will include a particular landmark at a particular time offset with a uniform probability ϵ, which is small, and that two video shots of the same event will share any given hash with a probability q. We assume q is constant, although in reality it depends on the relative quality and noise levels of the two recordings. We further assume that we have found the best alignment between Clip i and Clip j at an offset of o seconds, giving an overlap duration of t_{ov} seconds and $M_{i,j}$ shared landmarks.

In that same region of overlap, there are a total of $L_{ov}(i)$ landmarks from Clip i, which should be approximately $\frac{t_{ov}}{duration(i)} L(i)$, where $L(i)$ is the total number of landmarks in Clip i; similarly, $L_{ov}(j) \approx \frac{t_{ov}}{duration(j)} L(j)$. Because we are implicitly assuming comparable conditions in the two videos, we expect $L_{ov}(i)$ and $L_{ov}(j)$ to be about the same. Let $L_{max} = min(L_{ov}(i), L_{ov}(j))$, which places an upper bound on the number of matching landmarks.

Our simple model predicts that we would expect $M_{i,j}$ to be about $q \cdot L_{max}$. We can estimate q by dividing $M_{i,j}$ by L_{max} over a set of "true" match regions obtained via manual annotation. At the same time, we would expect to see landmarks from a chance alignment give $M_{i,j} = \epsilon \cdot L_{max}$, and thus we can also estimate ϵ from the best scores between non-true-matching overlaps.

Assuming each landmark in the overlap has an independent probability of occurring in the the other video, we would expect the distribution of $M_{i,j}$ given a true match (tm) to be Binomial distributed:

$$Pr(M_{i,j}|\mathrm{tm}) = \binom{L_{max}}{M_{i,j}} q^{M_{i,j}} (1-q)^{(L_{max}-M_{i,j})} \quad (1)$$

The probability of a chance match is given by the same equation, with ϵ replacing both instances of q.

To obtain the posterior, $Pr(\mathrm{tm}|M_{i,j})$, as distinct from a chance match (cm), we use Bayes rule:

$$Pr(\mathrm{tm}|M_{i,j}) = \frac{Pr(M_{i,j}|\mathrm{tm}) \cdot Pr(\mathrm{tm})}{Pr(M_{i,j}|\mathrm{tm}) \cdot Pr(\mathrm{tm}) + Pr(M_{i,j}|\mathrm{cm}) \cdot Pr(\mathrm{cm})} \quad (2)$$

which we call our matching likelihood score. To compute this score we need to estimate: q = average$(M_{i,j}/L_{max})$ over true match regions; ϵ = average$(M_{i,j}/L_{max})$ over chance match regions (where the averages are weighted in proportion to L_{max} to reduce the variance arising from short overlaps); and $Pr(\mathrm{tm})$ = proportion of video pairs with true matches out of all pairs within the set of videos. For now, we estimate $Pr(\mathrm{tm})$ from our ground truth, representing the proportion of all pairs of videos in the collection actually have a true overlap. Using statistics of past events and their ground truth would be one way to find good values to use for the priors. Alternatively, we could calculate a prior based on the length of the event and the amount of content captured.

3.3 Refinement Procedure

In this section, we assume we have the complete pairwise data for clips and offsets in one scene $\mathcal{S}_I$ output by Algorithm 1. We cast this as a clustering problem: is there a subdivision of the set of videos in one scene to two or more clusters that results in a more "coherent" overall match? We will drop the scene index and refer to it simply as scene $\mathcal{S}$. Subclusters of scene $\mathcal{S}$ will be denoted $\mathbb{C}_1, \mathbb{C}_2, \ldots \mathbb{C}_n$.

For example, if Algorithm 1 output the set of all four clips from Figure 1 as a single scene, the output of the current clustering step would strive to determine that this scene is really two different scenes, $\mathbb{C}_1$ with clips 1 and 2, and $\mathbb{C}_2$ with clips 3 and 4.

As a clustering/allocation problem, this can be addressed using any procedure that maximizes the ratio of overall intra-cluster similarity to inter-cluster similarity over a set of subclusters $\mathbb{C}_1, \mathbb{C}_2, \ldots \mathbb{C}_n$. Such a clustering algorithm finds an allocation of items (clips) to clusters such that the similarity between clips in the same cluster is high, and the similarity between clips in different clusters is low. We use a scoring method based on the Davies-Bouldin index [5]. The question becomes how to define the similarity between clips inside a cluster (intra-cluster similarity) and between clips in different clusters (inter-cluster similarity). One way to do so is to use the following:

Define $s(i, j)$ as the similarity score of clips c_i, c_j using the offset computed by Algorithm 1 in one of the ways listed above. Define $Sim(\mathbb{C}_h)$, the intra-cluster similarity of clips in a subcluster $\mathbb{C}_h$ of $\mathcal{S}$, as:

$$Sim(\mathbb{C}_h) = \frac{\sum_{i,j \in \mathbb{C}_h, i<j} s(i,j)}{|\mathbb{C}_h| \cdot (|\mathbb{C}_h| - 1)} \quad (3)$$

In other words, $Sim(\mathbb{C}_h)$ is the average pairwise similarity of clips in $\mathbb{C}_h$.

Similarly define inter-cluster similarity of two clusters $\mathbb{C}_h, \mathbb{C}_\ell$:

$$Sim(\mathbb{C}_h, \mathbb{C}_\ell) = \max_{i \in \mathbb{C}_h, j \in \mathbb{C}_\ell} s(i,j) \quad (4)$$

In other words, $Sim(\mathbb{C}_h, \mathbb{C}_\ell)$ is the maximum pairwise similarity between pairs of clips where each clip belongs to one of the proposed clusters.

An overall score for a subcluster $\mathbb{C}_h$ will capture the minimum distance between it and another subcluster – in other words, the score is based on the similarity of the subcluster to the most similar (closest) other subcluster:

$$R(\mathbb{C}_h) = \min_{\ell \neq h} \frac{Sim(\mathbb{C}_\ell) + Sim(\mathbb{C}_h)}{Sim(\mathbb{C}_\ell, \mathbb{C}_h))} \tag{5}$$

The final goodness score of the clustering process that produces n subclusters for scene $\mathcal{S}$ is then the average of the single subclusters' distance scores:

$$\sum_{h=1\ldots n} \frac{R(\mathbb{C}_h)}{n} \tag{6}$$

A higher value for the goodness score is better – it would mean that all subclusters are sufficiently distant from all other subclusters. Note that $R(\mathbb{C}_h)$ is not defined when $\mathbb{C}_h$ is the only subcluster for the scene (when $n = 1$). If $n = 1$, we subtitute the inter-cluster similarity with a value T, where T is set to be equal to or slightly larger than the least pairwise match amongst clips in $\mathcal{S}$. Therefore, the final clustering goodness for the case $n = 1$ is $\frac{Sim(\mathbb{C}_1)}{T}$.

A procedure to decide on the final cluster assignment and the number of clusters for each initial scene $\mathcal{S}$ does the following:

1. Compute a possible split of cluster $\mathcal{S}$ to k clusters, where $k = 1, 2, \ldots |\mathcal{S}|$.
2. For each k, compute the goodness score for the clustering of $\mathcal{S}$ for that number of clusters.
3. Pick the k that maximizes the goodness score, and use the computed clusters as the final outcome of the process.

We used Spectral Clustering [11] for the clustering step. The results of the clustering procedure for all scenes are "flattened" from the scenes-and-clusters hierarchy to a non hierarchical set of final scenes. For example, if Algorithm 1 creates initial scenes $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_3$, and the clustering breaks the scenes to three, two and one (no split) clusters respectively, $\mathbb{C}_{1_1}$, $\mathbb{C}_{1_2}$, $\mathbb{C}_{1_3}, \mathbb{C}_{2_1}, \mathbb{C}_{2_2}, \mathbb{C}_{3_1}$, we treat the results as six distinct final scenes: $\mathbb{S} = S1, S2, S3, S4, S5, S6$.

In our running example in Figure 1, this process would ideally result in $k = 2$ different scenes, $\mathcal{S}_1$ with clips 1 and 2, and $\mathcal{S}_2$ with clips 3 and 4.

4. EVALUATION

The goals of the evaluation were to measure the performance of the various strategies on content from real-world events, as well as understanding the effect of the threshold set for the greedy algorithm has on the outcome.

For the evaluation, we used videos captured at 10 different music events, listed in Table 1. For each event, we generated the ground truth of scenes as described below. We tested our scene alignment using the techniques described above using multiple thresholds (for Algorithm 1) and the two possible scoring functions for the optional clustering step.

For each event, and each video, we extract the audio fingerprints, and compute the maximum match offset between each two clips as described above. We then run the greedy Algorithm 1 with a low threshold=5 to get an initial scene alignment for all clips that is likely to include all true matches (as well as others as discussed above). We call this scene alignment *Greedy5*. The *Greedy5* alignment is then used to generate the ground truth.

	Event	Date	Num of Clips	Total Duration
1	LCD Soundsystem in New york	Apr 2 2011	174	1024
2	Linkin Park in Boston	Feb 1 2011	156	482
3	The Decemberists In San Francisco	Aug 14 2011	44	195
4	Daft Punk in Berkeley	Jul 27 2007	122	268
5	Taylor Swift in Dallas	Oct 8 2011	124	476
6	MGMT in LA	July 16 2010	32	148
7	Beyonce In Greece	Aug 11 2009	74	211
8	Radiohead in Rio de Janeiro	Mar 20 2009	325	1161
9	Coldplay in Glastonbury	Jun 27 2011	25	109
10	Jay-Z in Nashville	Nov 13 2009	30	92

Table 1: A list of events used for our evaluation, with the number of video clips for each and the total duration of the clips in minutes.

The ground truth is generated by marking, for every pair of clips in a scene computed by Algorithm 1, whether the two clips are a correctly marked as overlapping or whether the match is wrong. Note that we do not check for potential alignment of clips in different scenes, as it would be prohibitive to do so for every pair of clips and every possible offset. Instead, the scenes created by Algorithm 1 (with threshold=5) serve as the basis for the ground truth annotation. While this method might result in over-estimation of the recall (i.e., as we might miss clips that should overlap but were not detected), we estimate that this problem is not likely given the low match threshold used.

For each event, two human judges viewed all pairs of clips in each scene in Algorithm 1's output, marking the pair as true or false match as described above. The correct pairwise matching was then used to reconstruct ground truth scenes $\mathbb{G} = \{G_1, \ldots G_N\}$ using in a process similar to Algorithm 1. Note again that because of the way ground truth was generated, each ground truth scene is a subset of one of *Greedy5*'s scenes.

While *Greedy5* served as the baseline for computing the ground truth, we evaluated three different strategies using set of threshold values. The strategies included:

- **Greedy**: Using Algorithm 1's output directly.
- **Greedy+clustering** Performing a clustering step on the scenes as output from Algorithm 1 using scoring by number of matches.
- **Greedy+likelihood** Performing a clustering step on the scenes as output from Algorithm 1 using scoring by likelihood of overlap as described above.

4.1 Evaluation Metrics

We use two sets of metrics: standard clustering metrics, and pairwise match metrics. Because we create scenes, or a grouping of clips into sets (each set is a scene), we can evaluate this grouping using clustering quality metrics. However, the overlap between clips is not necessarily transitive (e.g., clips 1 and 4 in Figure 1 do not overlap even if they belong to the same scene). We therefore also evaluate the pairwise recall and precision of each alignment: how many of pairs that overlap are correctly identified by our scene construction.

Although several clustering quality metrics exist (see [1]), in this work we use on Normalized Mutual Information (NMI) [10, 15] and B-Cubed [1]. NMI is an information-theoretic metric that measures how much information is shared between the actual "ground truth" assignment of scenes, each with an associated set of clips, and the assignment made by our system under the different parameters. Specifically, for a set of computed scenes $\mathbb{S} = \{\mathcal{S}_1, \ldots, \mathcal{S}_N\}$ and ground-truth scenes $\mathbb{G} = \{G_1, \ldots G_M\}$, where each $\mathcal{S}_j$ and G_k is a set of clips, and n is the total number of clips, $NMI(\mathbb{S}, \mathbb{G}) = \frac{I(\mathbb{S},\mathbb{G})}{(H(\mathbb{S})+H(\mathbb{G}))/2}$, where: $I(\mathbb{S}, \mathbb{G}) = \sum_k \sum_j \frac{|G_k \cap \mathcal{S}_j|}{n} \log \frac{n \cdot |G_k \cap \mathcal{S}_j|}{|G_k| \cdot |\mathcal{S}_j|}$, $H(\mathbb{S}) = -\sum_j \frac{|\mathcal{S}_j|}{n} \log \frac{|\mathcal{S}_j|}{n}$, and $H(\mathbb{G}) = -\sum_k \frac{|G_k|}{n} \log \frac{|G_k|}{n}$.

B-Cubed is a measure that balances the the precision and recall associated with matches in the dataset, where for precision P and recall R values it computes $B\text{-}Cubed = \frac{2 \cdot P \cdot R}{P+R}$. For each event, precision is defined as the proportion of clip pairs in the event that overlap in both the computed scenes and the ground truth, out of all the pairs in the computed scenes. Recall is defined as the proportion of clip pairs in the event cluster that overlap in both the computed scenes and the ground truth, out of all the pairs in the ground truth. For example, say the system output all clips in Figure 1 as belonging to a single scene. The pairs of detected overlapping clips for this scene are (1,2), (1,3), (2,3), (2,4) and (3,4), but not (1,4) as these clips do not actually overlap. Further assume that the ground truth is in fact two scenes, one with clips 1 and 2 and another with 3 and 4: the ground truth set is (1,2), (3,4). The precision of the system is thus 2/5, the recall is 1, and the B-Cubed score is 0.57.

Both NMI and B-Cubed balance the clustering properties that we would like to have in our final solution: maximizing the homogeneity of clips within each scene, and minimizing the number of scenes that clips for each ground-truth scene are spread across. In addition to NMI and B-Cubed we will also directly examine the recall and precision values, as defined above, so we can reason about their trade off for the different strategies.

5. RESULTS

We evaluate the performance of the different strategies listed above, for different threshold values. We present aggregate results over all events and thresholds, then show results for individual events for a single threshold value. First, though, to illustrate the results, we present the specific outcome of the scene creation for event 3.

The constructed scenes for event 3 using two strategies, Greedy, and Greedy+Clustering (both with threshold 5) is shown in Figure 2, with Greedy results shown on top. The x-Axis captures time and the y-Axis is clip ID, organized by scene. Since scenes are not overlapping, time progress is stacked: a scene is marked as starting at the time offset where the previous scene ended. For each clip, a dot shown where there is a hash match with other clips in the same scene. For example, the Greedy algorithm (top) computed a total of eight scenes for the event, the first one being the largest with most clips. The arrows indicate two Greedy scenes that were broken into finer scenes by the clustering step. Indeed, the results below show that for event 3, the Greedy strategy had low precision, while the Greedy+Clustering (bottom) approach had almost perfect precision and recall. Figure 2 illustrates the effect that errors in threshold-based methods may have on results, which we study in more detail next.

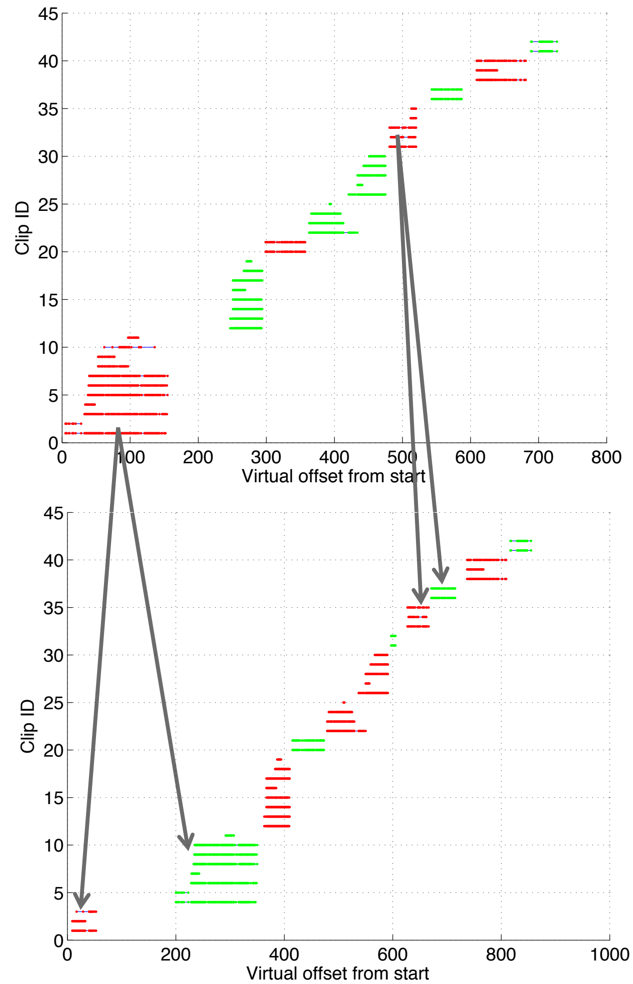
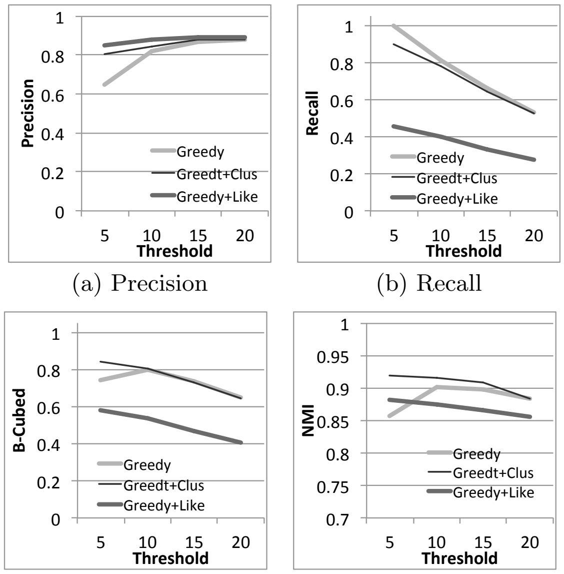


Figure 2: Results (clip overlap and scene structure) using two strategies for event 3; Greedy (top) and Greedy+Clustering (bottom)

Figure 3 shows the precision, recall, B-Cubed and NMI results, averaged over all events, for each of our three strategies. For example, in Figure 3(d), the score for the Greedy+ Clustering strategy with threshold 5 is 0.92 (top left). The figure shows that Greedy+Clustering performs best across all threshold values in almost all measures. In terms of recall (Figure 3(b)), the Greedy output serves as an upper bound for all strategies, due to the fact that the clustering can only sub-divide Algorithm 1's computed scenes. Still, the recall results for Greedy+Clustering are very close to the upper bound. In addition, considering the experience of a person viewing the results of the alignment, precision should probably be emphasized over recall.

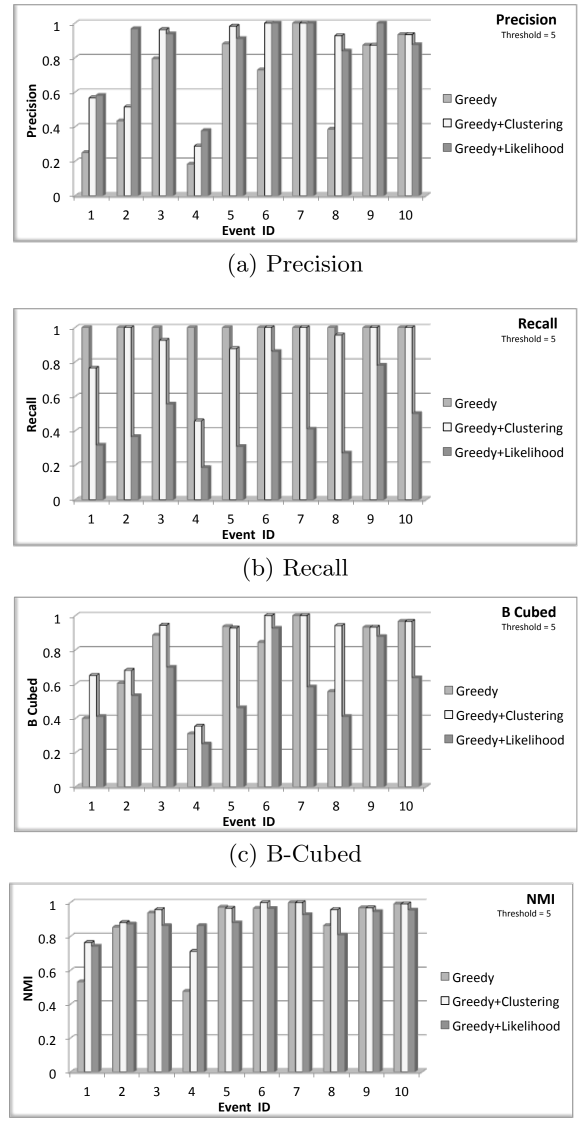
Figures 4 and 5 show the detailed results for the different strategies across all events, with two different threshold values, 5 (Figure 4) and 15 (Figure 5). For example, Figure 4(a) shows that for Event 1, all strategies resulted in low



(a) Precision (b) Recall

(c) B-Cubed (d) NMI

Figure 3: Precision, recall, B-Cubed and NMI results for different strategies across different threshold values



(a) Precision

(b) Recall

(c) B-Cubed

(d) NMI

Figure 4: Performance across all events, using three different strategies, with threshold 5

precision when using threshold 5, with Greedy+Likelihood performing best at around 0.58. On the other hand, Figure 5(a) shows that the precision for the same event goes up, for all strategies, reaching over 0.75. This rise is of course expected (and reflected in Figure 3 above) and correlated with lower recall as Algorithm 1 will consider much fewer matches when the threshold is higher. Note that the recall for Greedy in Figure 4(b) is 1 for all events, as the ground truth is defined in respect to Greedy5, and matches not detected by this strategy (if any) were ignored.

Figures 4 and 5 highlight the fact that, by and large, with threshold 15, all events demonstrates high level of precision across all categories, with the exception of the events with the largest numbers of videos (see Table 1). Recall levels, however, vary. However, it does seem like the mixed strategies were more effective for large events. For example, for events 1, 2, and 8 in Figure 4, Greedy+Clustering achieves high levels of both precision and recall compared to other strategies and compared to the higher threshold in Figure 5.

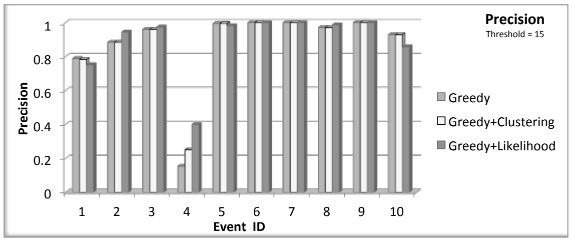
6. CONCLUSIONS

We have investigated multiple strategies that aim to construct scenes, or sets of overlapping videos taken by users at events. We tested our methods on social videos from 10 different events of various scale. We showed the effect of the match threshold on the recall and precision and resultant scene matches. A hybrid strategy, with a greedy match of clips followed by a clustering step to refine the scene, performed best across multiple threshold settings. The hybrid Greedy+Clustering strategy performed better for large events, consisting of many captured videos. In these events, due to the redundancy of multiple synchronous camera shots, precision – where Greedy+Clustering has the advantage – is more critical than recall.

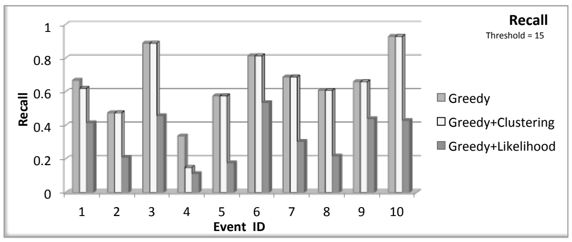
One limitation is the robustness of the results. Since results significantly varied between events, we cannot conclusively determine that one strategy is overwhelmingly better than others; more data and test are required that will allow for a statistical test of the significance of the results. In addition, we are currently estimating prior likelihoods by averaging the results of the ground truth data. In future work we would also like to address integrating other signals, such as text and time information from the videos that do provide it. The text, for example, can provide indicators about common content for overlapping clips, as hinted by previous work [8]. Finally, many events contain social videos that were edited by the user prior to upload. These remixes would serve as a unique challenge, potentially overlapping with multiple videos in a non-linear fashion, that we would like to address in future work.

7. ACKNOWLEDGMENTS

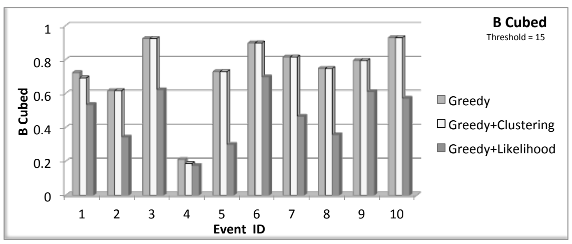
This work was partially supported by the National Science Foundation award IIS-1017845.



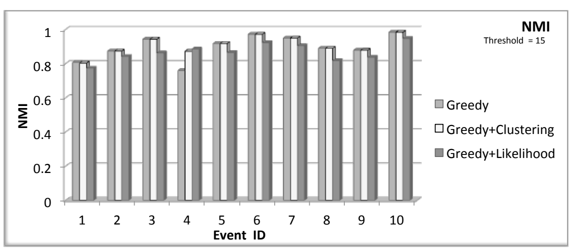
(a) Precision



(b) Recall



(c) B-Cubed



(d) NMI

Figure 5: Performance across all events, using three different strategies, with threshold 15

8. REFERENCES

[1] E. Amigó, J. Gonzalo, J. Artiles, and F. Verdejo. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 2008.

[2] L. Ballan, G. J. Brostow, J. Puwein, and M. Pollefeys. Unstructured video-based rendering: interactive exploration of casually captured videos. *ACM Trans. Graph.*, 29:87:1–87:11, July 2010.

[3] H. Becker, D. Iter, M. Naaman, and L. Gravano. Identifying content for planned events across social media sites. In *Proceedings of the fourth ACM international conference on Web search and data mining*, WSDM '12, New York, NY, USA, 2011. ACM.

[4] A. Broder, S. Glassman, M. Manasse, and G. Zweig. Syntactic clustering of the web. *Computer Networks and ISDN Systems*, 29(8-13):1157–1166, 1997.

[5] D. L. Davies and D. W. Bouldin. A Cluster Separation Measure . In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 224 – 227, April 1979.

[6] D. Ellis. Robust Landmark-Based Audio Fingerprinting. web resource, `http://labrosa.ee.columbia.edu/matlab/fingerprint/`, 2009.

[7] J. Foo, J. Zobel, and R. Sinha. Clustering near-duplicate images in large collections. In *Proceedings of the international workshop on Workshop on multimedia information retrieval*, pages 21–30. ACM, 2007.

[8] L. Kennedy and M. Naaman. Less talk, more rock: automated organization of community-contributed collections of concert videos. In *WWW '09: Proceeding of the 18th international conference on World Wide Web*, pages 311–320, New York, NY, USA, 2009. ACM.

[9] X. Liu, R. Troncy, and B. Huet. Finding media illustrating events. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, ICMR '11, pages 58:1–58:8, New York, NY, USA, 2011. ACM.

[10] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge Univ. Press, 2008.

[11] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances In Neural Information Processing Systems*, pages 849–856. MIT Press, 2001.

[12] P. Shrestha, M. Barbieri, and H. Weda. Synchronization of multi-camera video recordings based on audio. In *Proceedings of the 15th international conference on Multimedia*, MM '07, pages 545–548. ACM Press, 2007.

[13] P. Shrestha, P. H. de With, H. Weda, M. Barbieri, and E. H. Aarts. Automatic mashup generation from multiple-camera concert recordings. In *Proceedings of the international conference on Multimedia*, MM '10, pages 541–550, New York, NY, USA, 2010. ACM.

[14] C. G. Snoek, B. Freiburg, J. Oomen, and R. Ordelman. Crowdsourcing rock n' roll multimedia retrieval. In *Proceedings of the international conference on Multimedia*, MM '10, pages 1535–1538, New York, NY, USA, 2010. ACM.

[15] A. Strehl, J. Ghosh, and C. Cardie. Cluster ensembles–A knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, 2002.

[16] A. Wang. An Industrial Strength Audio Search Algorithm. In *Proceedings of the International Conference on Music Information Retrieval*, 2003.

[17] V. Zsombori, M. Frantzis, R. L. Guimaraes, M. F. Ursu, P. Cesar, I. Kegel, R. Craigie, and D. C. Bulterman. Automatic generation of video narratives from shared ugc. In *Proceedings of the 22nd ACM conference on Hypertext and hypermedia*, HT '11, pages 325–334, New York, NY, USA, 2011. ACM.