Large-Scale Video Summarization Using Web-Image Priors

Aditya Khosla[†] Raffay Hamid[‡] Chih-Jen Lin[*] Neel Sundaresan[‡]
[†]Masschusetts Institute of Technology, Cambridge MA 02139
[‡]eBay Research Labs, San Jose CA 95032
[*]National Taiwan University, Taipei 10617, Taiwan
khosla@csail.mit.edu, {rhamid, nsundaresan}@ebay.com, cjlin@csie.ntu.edu.tw

Abstract

Given the enormous growth in user-generated videos, it is becoming increasingly important to be able to navigate them efficiently. As these videos are generally of poor quality, summarization methods designed for well-produced videos do not generalize to them. To address this challenge, we propose to use web-images as a prior to facilitate summarization of user-generated videos. Our main intuition is that people tend to take pictures of objects to capture them in a maximally informative way. Such images could therefore be used as prior information to summarize videos containing a similar set of objects. In this work, we apply our novel insight to develop a summarization algorithm that uses the web-image based prior information in an unsupervised manner. Moreover, to automatically evaluate summarization algorithms on a large scale, we propose a framework that relies on multiple summaries obtained through crowdsourcing. We demonstrate the effectiveness of our evaluation framework by comparing its performance to that of multiple human evaluators. Finally, we present results for our framework tested on hundreds of user-generated videos.

1. Introduction

Over the years, there has been a tremendous growth in the amount of user-generated video data [5]. These videos are extremely diverse in their content, and can vary in length from a few minutes to a few hours. It is therefore becoming increasingly important to automatically extract a brief yet informative summary of these videos in order to enable a more efficient and engaging viewing experience. In this work, we focus on the problem of automatic summarization and evaluation of user-generated videos.

Summarizing user-generated videos is different from well-produced videos in two important ways. First, user-generated videos are usually of poor quality, with erratic camera motion, variable illumination conditions, and scene clutter. This makes it difficult to rely solely on the low-level appearance and motion cues to find important key-frames, as done by a majority of the previous approaches [13] [26].

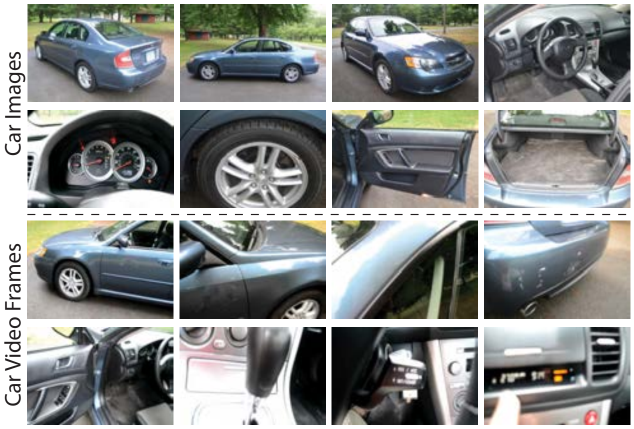

Figure 1: Consider a user listing of an automobile containing both images and videos of a particular object (*e.g.*, car). The top two rows show the images, while the bottom two rows show eight uniformly sampled frames from the corresponding user-generated video. Unlike the individual images, that are taken from different *canonical viewpoints* to capture the car in a maximally informative way, very few of the sampled video-frames are as informative.

Secondly, most of the user-generated videos contain only a small fraction of frames where some interesting event is happening. Due to this content sparsity, techniques that attempt to find representative frames with high appearance differences amongst them usually do not produce semantically meaningful results [22] [15].

The main contribution of this work is the idea of using web-images as a prior to facilitate the process of creating summaries of user-generated videos. Our intuition is that people tend to take pictures of objects and events from a few *canonical viewpoints* in order to capture them in a maximally informative way. On the other hand, as shown in Figure 1, user-generated videos taken by hand-held cameras often contain many uninformative frames captured while transitioning between the various canonical viewpoints.

We therefore hypothesize that images of objects and events present on the web contain information that could be used as a prior for building semantically meaningful

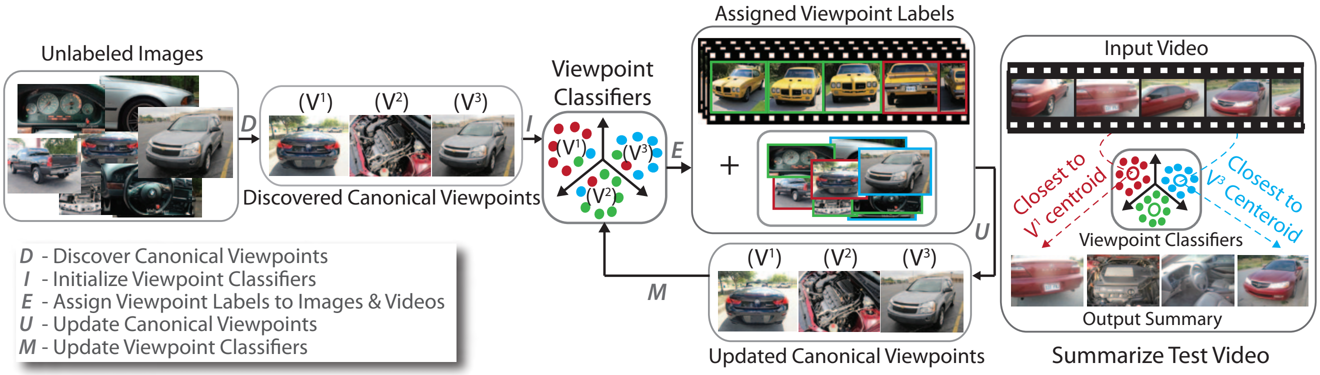
Unlabeled Images
D
(V¹) (V²) (V³)
Discovered Canonical Viewpoints
I
Viewpoint Classifiers
(V¹) (V³) (V²)
E
Assigned Viewpoint Labels
+
U
M
(V¹) (V²) (V³)
Updated Canonical Viewpoints
Input Video
Closest to V¹ centroid
Closest to V³ Centeroid
Viewpoint Classifiers
Output Summary
Summarize Test Video
D - Discover Canonical Viewpoints
I - Initialize Viewpoint Classifiers
E - Assign Viewpoint Labels to Images & Videos
U - Update Canonical Viewpoints
M - Update Viewpoint Classifiers

Figure 2: We start with an unlabeled corpus of images belonging to some class (*e.g.*, automobiles), and cluster them into subclasses, where each discovered subclass corresponds to a "canonical viewpoint". We also learn a classifier for each of the discovered subclass. These two operations are done iteratively by optimizing Equation 1. To improve our subclass models, we also use the unlabeled video data by first assigning each video frame to a subclass, and then repeating the optimization procedure from Section 2.1 with both video frames and images. Given a test video, we assign its frames to the learned subclasses, and compute the average decision score of the test examples assigned to each subclass. We use this score to rank the subclasses for the given test video. Finally to generate the output summary with k representative frames, we select the k frames from the test video, each of which is closest to the centroid the top k ranked subclasses.

summaries of user-generated videos in a scene-independent manner. In this work, we apply our novel intuition to propose a summarization algorithm that incorporates this image-based prior to automatically select the maximally informative frames from a video.

An important related question we explore in this work is the evaluation of video summarization algorithms in a large-scale setting. A majority of the previous work on video summarization uses expert opinion to evaluate their results. However, this evaluation methodology does not scale well with the size of video data. Moreover, videos considered by these methods are generally well-produced with their content following a strict cinematographic structure. This makes it possible to streamline the rules using which the experts could perform evaluation. Since user-generated videos are produced at scale, and do not follow strict structure, using expert opinion for their evaluation is infeasible.

To this end, we propose to rely on crowd-sourcing to obtain multiple candidate summaries of user-generated videos, to get a general sense of what their summaries should look like. We cast the question of matching the output of a summarization algorithm and the crowd-sourced summarization as a graph-theoretic problem. We compare the performance of our automatic evaluation framework to that of expert evaluation, demonstrating its effectiveness in capturing the characteristics of a good summary.

The main contributions of our work are:

- A novel intuition to incorporate information from web images as a prior to automatically select maximally informative frames from user-generated videos without using any human annotated summaries for training.
- A crowd-sourcing based automatic evaluation framework to evaluate the results of multiple video summarization algorithms.
- An analysis of our summarization mechanism tested over a large set of user-generated videos.

In the following, we start by presenting the details of our proposed algorithm in Section 2. We then explain how we automatically evaluate different summarization results using a crowd-sourcing platform in Section 3, and present a comparative analysis of our experiments and results in Section 4. We discuss related work in Section 5 and conclude this paper by summarizing our main findings and highlighting potential directions for future research in Section 6.

2. Using Web Images for Summarization

Our goal is to discover a set of "canonical viewpoints" in web-images related to an object class (*e.g.*, automobiles). Using these discovered viewpoints, we want to learn a discriminative model to identify similar frames in a video capturing a different instance of the object class. These keyframes can be used to represent the video and summarize its content. To this end, we propose to use a Support Vector Machine (SVM) framework (Section 2.1) that jointly discovers the canonical viewpoints (also called subclasses) as well as learns their discriminative decision boundaries.

Since many user-generated videos are captured from hand-held mobile devices, they contain a lot of variation in the viewpoints from which they capture an object. As people transition between canonical views of an object, they tend to capture the object in atypical and often uninformative views. Therefore, frames from these videos can be used as difficult-to-classify negative examples to further improve models of the canonical viewpoints. To this end, we use the pre-trained viewpoint classifiers learned only from web-images to initialize a second round of training, where the labels for both web-images and video frames are consid-

Algorithm 1 Video summarization algorithm (explained in Section 2). J denotes the total number of iterations and $\mathbf{w}_i^{(j)}$ represents the weight vector for class i, at iteration j.

```
Input: Unlabeled images $\mathbf{x}^D$ and videos $\mathbf{x}^V$, number of subclasses $K$.
Output: Learned weight vector $\mathbf{w}_y^{(J)} \forall y \in \{1 \dots K\}$.
Initialization: Initialize image labels $\hat{y}^D$ using k-means.
Refine image labels $\hat{y}^D$ and learn $\mathbf{w}^{(0)}$ using Equation 1.
for: $j = 1 \dots J$, do
    $\hat{y}_i^V = \arg\max_y(\mathbf{w}_y^{(j-1)} \cdot \mathbf{x}_i^V) \forall i$
    $\hat{y}_i^D = \arg\max_y(\mathbf{w}_y^{(j-1)} \cdot \mathbf{x}_i^D) \forall i$
    Learn $\mathbf{w}^{(j)}$ using $(\mathbf{x}^V, \hat{y}^V)$ and $(\mathbf{x}^D, \hat{y}^D)$
end for
```

ered latent (Section 2.2). Our overall approach is listed in Algorithm 1, and is illustrated in Figure 2.

2.1. Identifying Canonical Viewpoints

In order to discover the canonical viewpoints, we want to identify visually similar images in a corpus of web images. Furthermore, we want to reliably identify these viewpoints in a previously unseen set of video frames. We therefore have two main challenges:

- Clustering images into canonical viewpoints, and
- Learning a discriminative classifier for each viewpoint.

To achieve both these objectives, we iterate between them based on the following optimization problem:

$$\begin{aligned} &\min_{\mathbf{w},\xi\geq 0,\hat{y}} \quad \frac{1}{2}\sum_{k=1}^{K}||\mathbf{w}_k||^2 + C\sum_{i=1}^{N}\xi_i \\ &\text{s.t.} \quad \mathbf{w}_{\hat{y}_i}\cdot\mathbf{x}_i - \mathbf{w}_{\hat{y}}\cdot\mathbf{x}_i \geq \Delta(\hat{y}_i,\hat{y}) - \xi_i, \quad \forall \hat{y} \in \{1\dots K\} \end{aligned} \tag{1}$$

where $\Delta(\hat{y}_i, \hat{y}) = 1$ if $\hat{y}_i \neq \hat{y}$ and 0 otherwise, ξ is the slack variable and C is a hyperparameter. Equation 1 is a variant of a multi-class SVM [6] where the class labels of the examples are unknown. Specifically, given N images, we want to find K clusters and further learn K decision boundaries, one for each cluster. We represent the features of image i by $\mathbf{x}_i \in \mathbb{R}^n$ where n is the dimensionality of the feature space, and their unknown labels by $\hat{y}_i \in \{1 \dots K\}$.

Objective function optimization: Note that the objective function in Equation 1 is non-convex as the labels $\hat{y}$ are unknown. However, when $\hat{y}$ are known, the problem reduces to a multi-class SVM, which is convex. We therefore use an iterative procedure where we first learn the cluster labels, and then keeping these labels fixed, learn the SVM parameters. This process is repeated over multiple iterations.

We initialize the cluster labels using k-means clustering and update them at each iteration as: $\hat{y}_i = \arg\max_y(\mathbf{w}_y \cdot \mathbf{x}_i)$. While this optimization procedure does not guarantee a global optima, it does guarantee a local optima, since each iteration always reduces the objective function (for analytical proof, please see the supplementary material).

Implementation details: We use $K = 100$ subclasses with the hyperparameter $C = 30$ and perform 10 iterations of the algorithm. Furthermore, we add a class $K + 1$ that contains only negative examples[1] to identify frames that do not contain the object of interest.

2.2. Using Unlabeled Videos for Training

In this section, we assume that we have a classifier for one canonical viewpoint and we want to identify additional examples from the videos from the same viewpoint. We break the videos into frames and treat all the frames as independent examples. We denote the features and labels of the M training video examples as $\mathbf{x}_i^V$ and $\hat{y}_i^V \in \{-1, 1\}$ respectively, and similarly for the N training images, as $\mathbf{x}_i^D$ and $y_i^D \in \{-1, 1\}$. Note that in this case we have already discovered and assigned the labels y^D for the images, and our goal is to learn the latent labels for the video examples, $\hat{y}^V$. Again, we use an SVM framework, and define our objective function as:

$$\begin{aligned} &\min_{\mathbf{w},\xi\geq 0,\rho\geq 0,\hat{y}^V} \quad \frac{1}{2}||\mathbf{w}||^2 + C_1\sum_{i=1}^{N}\xi_i + C_2\sum_{i=1}^{M}\rho_i \\ &\text{s.t.} \quad y_i^D \mathbf{w}\cdot\mathbf{x}_i^D \geq 1 - \xi_i, \quad i = 1\dots N \\ &\qquad\; \hat{y}_i^V \mathbf{w}\cdot\mathbf{x}_i^V \geq 1 - \rho_i, \quad i = 1\dots M \end{aligned} \tag{2}$$

where ξ and ρ are slack variables, and C_1 and C_2 are hyperparameters. In this case, we define two hyperparameters so we can control the relative importance of image and video examples. The optimization procedure for this objective function is very similar to the one described in Section 2.1. In our experiments, we set $C_2 = C_1/10$ to give more importance to images as they are considered to be more reliable prior information, as compared to video frames.

2.3. Overall Algorithm

The overall algorithm consists of combining the methods from Section 2.1 and Section 2.2. We do this by first learning the image subclasses using the iterative optimization method described in Section 2.1. We then assign each frame from all videos to a subclass using the equation $\hat{y}_i^V = \arg\max_y(\mathbf{w}_y \cdot \mathbf{x}_i^V)$ where the weights learned from the images are used for the video frame subclass assignment. We repeat the optimization from Section 2.1 with both video frames and images for a few iterations.

Given a test video, we assign its frames to the different subclasses using their learned classifiers, and compute the average decision score of the positive examples from each subclass. We use this score to rank the subclasses in terms

[1] Details about negative examples are given in Section 4.2.1.

of which ones are most pertinent to the test video. To generate the output summary with k representative frames, we select the k frames from the test video, each of which is closest to the centroid of any one of the top k ranked subclasses. We find that it is better to select the example closest to the subclass centroid compared to the highest scoring example, as that may not be representative of the subclass.

3. Large-scale Annotation and Evaluation

In this section, we describe the process of obtaining annotations through crowdsourcing on Amazon Mechanical Turk (AMT), and how these annotations can be used for automatic evaluation using the notion of average precision. Overall, the process consists of obtaining multiple summaries of a single video via AMT, and later comparing those summaries against the ones obtained by applying different algorithms. This results in precision-recall curves, which can be used to compare algorithms against one another.

3.1. Obtaining Annotation using Mechanical Turk

Summarizing a video is a subjective task, and summaries produced by different people are often different, even when done by experts. Thus, it is beneficial to obtain multiple summaries of a single video as ground truth to evaluate the performance of different algorithms. This method has been shown to be promising when using expert annotators, but it is not easily scalable. One way to overcome this barrier is to use AMT. As the workers are typically not highly-skilled and we cannot explain the task in person to ensure it is well understood, we have to design a system to identify workers capable of performing this task.

Similar to the work in [1], we first extract keyframes using [10], which are shown to the workers on AMT (called *turkers*) for selection. A turker must select at least 3 and at most 25 frames that he believes adequately summarize the content of the frames shown. While instructions are provided for this task, we found that it is important to have a test to ensure that they are well understood.[2] Further, to ensure that instructions are being followed, we randomly sample 5% of the images from the video and include duplicates with slight perturbations (crop and rescale). A worker that selects both the original and the perturbed frame is blocked, and all his previous annotations are dropped. On average, it cost about $0.20 per summary per video, and we obtain a total of 10 summaries per video, for a total of 155 videos.

3.2. Evaluation using Average Precision

Since the number of frames to use for a summary is application dependent, we propose to evaluate a variable number of frames from a ranked list (similar to [14]). For a particular video, given a summary $H_1, \ldots, H_n$ by a turker, we want to evaluate the average precision of a rank-ordered summary $S_1, \ldots, S_n$ proposed by an algorithm, where S_1 is the highest-scoring frame. Assume that we require a ranked list of size n from the algorithm for a reference summary of size n. Thus, we can iteratively evaluate the precision and recall of using 1 frame ($\{S_1\}$), 2 frames ($\{S_1, S_2\}$) and so on to plot a precision recall curve. Below we describe how we find the precision and recall for k frames ($\{S_1, \ldots, S_k\}$).

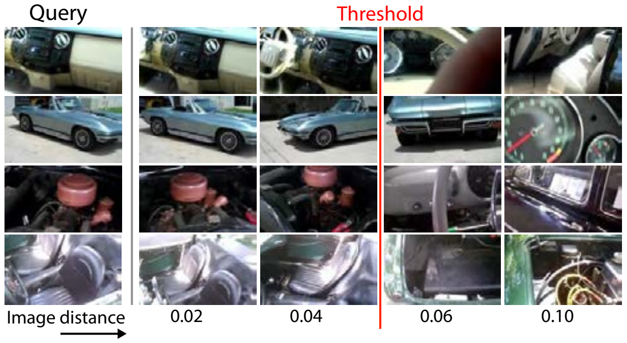

Figure 3: Sample query images and their respective matches when using SIFTFlow to compute d. As shown, SIFTFlow is quite robust to small changes in viewpoint resulting in semantically meaningful matches. We threshold d such that frames with larger distances are set to have $d = 1$.

Precision refers to the fraction of retrieved instances that are relevant while recall refers to the fraction of relevant instances that are retrieved. Thus, to compute precision, we want to find how well all the retrieved frames match with the reference frames, while to compute recall we want to find how many, and how accurately, are the reference frames returned in the retrieval result. In order to do this, we first need to define the notion of "how well does a frame match with another frame". This distance is denoted as d.

Defining frame distance: Given two frames, F_1 and F_2, we compute the SIFTFlow [20] between them and find the warped image for one of the two frames F_1^w. We define d to be the sum-squared pixel-wise distance between F_1^w and F_2 *i.e.*, $d = |F_1^w - F_2|^2/P$ where P is the number of pixels such that $0 \leq d \leq 1$. Figure 3 shows some query images and their respective matches using SIFTFlow to compute d. As shown, SIFTFlow is quite robust to small changes in viewpoint, and is quite consistent in assigning distance values to images in accordance to their variations. For frames with d greater than a threshold, we set it equal to 1.

Bipartite frame matching: Based on the above distance metric, we construct a distance pair-wise matrix, $D \in \mathbb{R}^{nxk}$, between all of the retrieved and reference frames. We want to find a matching between the two sets of frames such that there is one reference frame corresponding to each retrieved frame. This corresponds to finding a bipartite matching between the two sets of frames as illustrated in Figure 4. We apply the Hungarian algorithm [11] to D to find the

[2] The set of instructions, and a sample test is provided in the supplementary document.

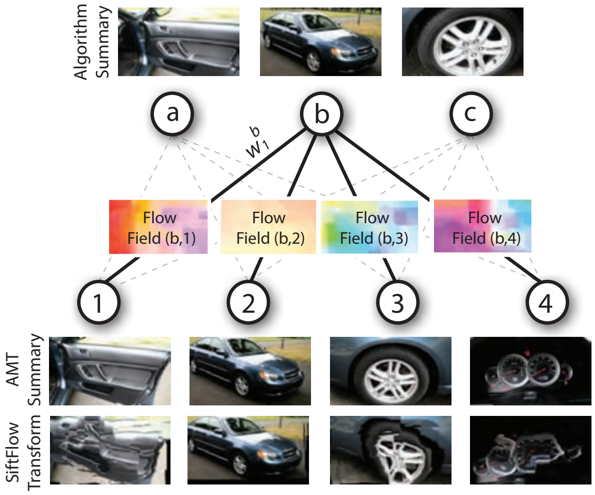

Figure 4: Bipartite matching to assign frames between a video summary given by an AMT worker, and a summarization algorithm. Edges from node b in the top partite to all nodes in the bottom partite are drawn in solid. Also shown are the SIFTFlow [20] fields found while transforming frame b to all frames in the AMT-summary. The transformed frames are shown in the bottom row. Edge weights (*e.g.*, w_1^b) are equal to the average pixel-wise distance of the transformed image and the original image.

lowest cost bipartite matching between the frames. Using the resulting frame correspondence, we compute the precision as $\sum_{i=1}^{k}(1 - d_i)/k$ where d_i are the distances of the best matches, and similarly recall is computed as $\sum_{i=1}^{n}(1-d_i)/n$. We find the area under the precision-recall curve to obtain the average precision.

4. Experiments

4.1. Dataset

For this work, we focus on the "Cars and Trucks" class of objects, since it is one of the most popular categories where users upload both images and videos to ecommerce websites. In order to collect our image corpus, we crawled several popular ecommerce websites and downloaded about $300,000$ images of cars and trucks that users had uploaded to their listings. To collect video data, we searched for all the user listings with a "youtube.com" URL in their product description. For each of these 180 listings, we downloaded their corresponding videos from youtube.com. We randomly selected 25 videos for training, and the rest (155) for evaluation. We ensured that images from listings containing test videos were not included in our training data.

4.2. Setup and Baselines

4.2.1 Setup

We used our downloaded vehicle image corpus as the positive set, and images from the PASCAL VOC 2007 dataset [8] that do not contain cars or trucks as the negative set. For the 25 training videos, we extracted 2 uniformly spaced frames per second. The images and video frames were resized to have a maximum dimension of 500 pixels (preserving aspect ratio). We extracted SURF [4] descriptors in a dense grid with grid spacing of 6 and learn a codebook of size 256 using k-means clustering. The descriptors are assigned to the codebook using Locality-Constrained Linear Coding [31] and combined in a 2-level spatial pyramid representation [16] resulting in a feature dimension of $1,280$. We used LIBLINEAR [9] to train the subclass models of Equation 1.

4.2.2 Baselines

For comparison, we use four baseline algorithms that do not require training summaries: random sampling, uniform sampling, k-means and spectral clustering. Assume that n frames are returned by the algorithm. Random sampling is the simplest baseline where we randomly select n frames from the video. In uniform sampling we split the video into $n + 1$ equal segments where the last frame from the first n segments is selected. In k-means and spectral clustering, we create n clusters and select the image closest to each cluster centroid. We report the average of 20 runs for random sampling, k-means clustering and spectral clustering.

4.3. Automatic Evaluation

We performed automatic evaluation using the method described in Section 3. Apart from the baseline methods described in Section 4.2 and our method, we also compute the average precision (AP) when reference summaries from the AMT workers are used for evaluation. This provides a pseudo-upper bound for this task, and thus we also report normalized AP scores by rescaling the AP of AMT workers to 100%. The results are reported in Table 1.

We observe that our algorithm significantly outperforms all baseline methods to achieve a normalized AP of 59.5%. For the baselines, k-means and spectral clustering outperform uniform and random sampling. Figure 5 shows the summaries produced by different methods for three example videos to give the reader a visual sense of the summaries produced by different algorithms. Note that our algorithm produces summaries most similar to the ones generated by human annotators. Figure 6 shows the improvement of our method over the baseline (k-means) for individual videos. We observe that our method consistently improves the result across a large proportion of the videos.

4.4. Human Evaluation

With 15 human judges, we performed human evaluation of the retrieved summaries from different algorithms to verify the results obtained from the automatic evaluation. Each expert was shown a set of 25 randomly sampled videos

Method	Baselines				Our method			AMT
	Uniform	Random	k-means	Spectral	Init	Img only	Img + Vid	
AP	0.0719	0.0705±0.004	0.0764±0.003	0.0742±0.003	0.08557	0.0897	**0.0934**	0.1571
AP$_i$/AP$_{AMT}$	45.8%	44.9%±2.4	48.6%±1.9	47.3%±1.9	54.5%	57.1%	**59.5%**	100%

Table 1: **Automatic Evaluation**: Average precision (AP) results. The second row rescales the AP of AMT to 100% for comparison, since that is a pseudo upper-bound on the results.

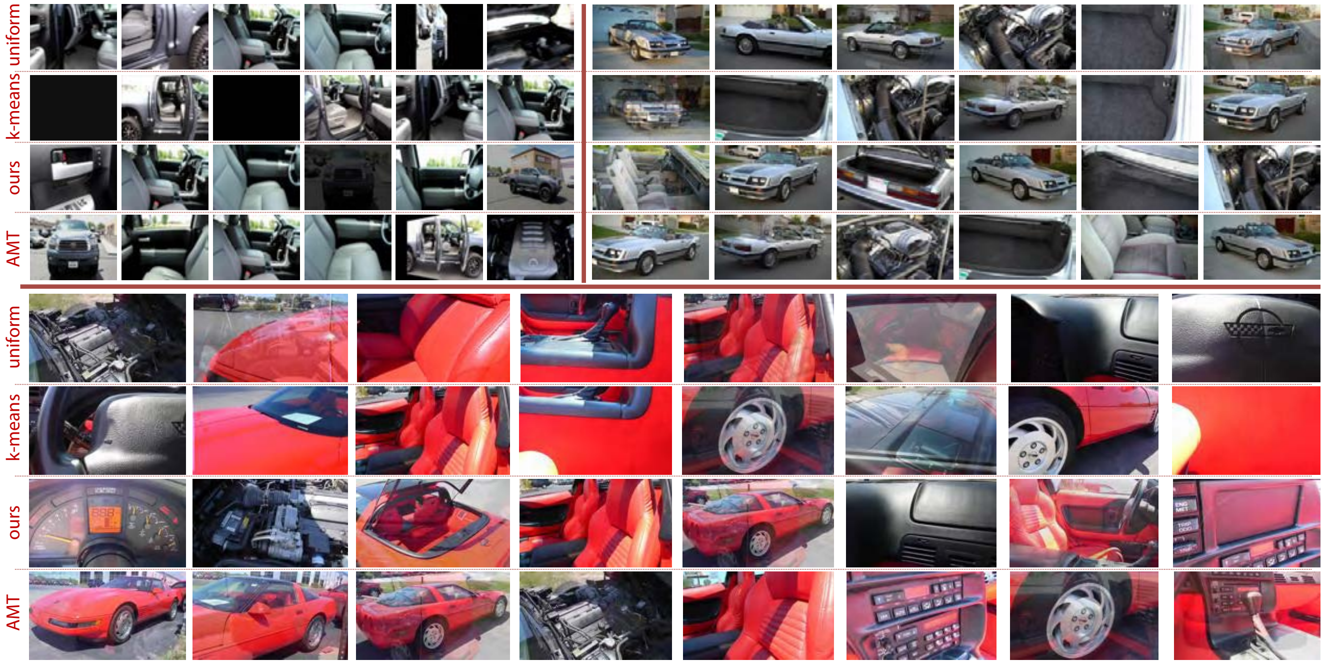

Figure 5: Visual comparison of results from various methods (*i.e.*, uniform, k-means, ours and AMT), showing different number of frames for 3 different videos to give a sense of the visual quality of summaries obtained using the different summarization algorithms. The corresponding videos are included in the supplementary material.

from our “Cars and Trucks” dataset. They watched the video at 3x speed and were then shown 4 sets of summaries constructed using different methods: uniform sampling, k-means clustering, our proposed algorithm (Section 2), and a reference summary from AMT. They were asked to rate the overall quality of each summary by assigning a rating from 1 to 10. The results are summarized in Table 2.

Similar to the result of automatic evaluation, our algorithm significantly outperforms other methods (uniform, k-means). Furthermore, we note that the relative rank of the different algorithms is largely preserved in the human evaluation as compared to the automatic evaluation (Table 1). In addition, we found a Spearman’s rank correlation coefficient of 0.70 between the scores assigned to each video by human evaluators and our automatic method. This suggests the efficacy of our method for performing evaluations automatically. Finally, the high performance of AMT summaries in both human and automatic evaluation illustrates that our method to obtain summaries using crowdsourcing is effective, allowing us to evaluate video summarization results in a large-scale setting, while keeping costs low.

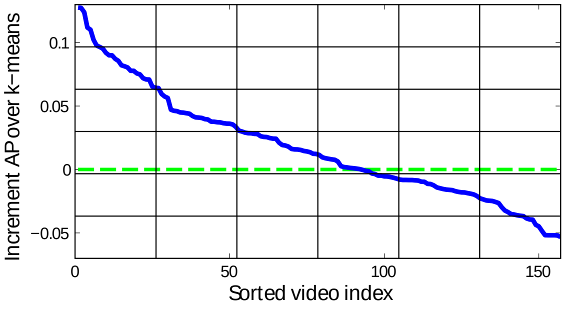

Figure 6: Improvement of our algorithm over baseline (k-means) for individual videos sorted by the amount of improvement. We observe that there is consistent improvement across a large proportion of the videos as compared to the baseline.

4.5. Discussion & Analysis

Based on our experiments and results, we now discuss some of the insights from our work. Our main intuition is that people tend to take pictures of objects from select viewpoints in order to capture them in a maximally informative way. We therefore hypothesized that images of objects could be used to create summaries of user-generated videos

Method	Uniform	K-means	Ours	AMT
Avg. Score	3.89	3.59	**4.58**	6.58

Table 2: **Human Evaluation**– 15 human judges evaluated the summaries from different algorithms on a scale of 1 to 10 to verify the results of our automatic evaluation scheme. Our algorithm significantly outperforms other automatic methods.

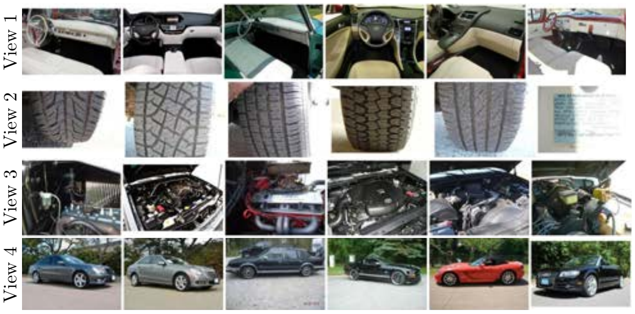

Figure 7: Examples of learned *canonical viewpoints*. These are the highest scoring images in each cluster arranged in a descending order from left to right.

capturing those objects. As shown in Table 1, the results of our experiments confirm our hypothesis, where the average precision we obtain while using web-image prior to summarize videos is significantly better (54.5%) than all the baseline methods that do not incorporate this prior information (46.7% on average).

We also posit that since user-generated videos have a lot of variation in the viewpoints from which they capture an object, frames from these videos could be used in addition to the image based prior information to further improve the summarization performance. This hypothesis is confirmed by the results in Table 1, where using video frames and images together performs better (59.5%) than using images only (57.1%). This is because combining images and video frames results in viewpoint clusters that are largely coherent (see Figure 7 for example clusters). This in turn enables our learned models to do accurate frame retrieval during testing.

The competence of our framework is also corroborated by the human evaluation (Table 2), where *all* 15 judges consistently ranked our proposed algorithm next to the AMT summaries. This indicates that using image based priors for summarizing user-generated videos captures what humans consider good summaries. Furthermore, based on the feedback from the judges we learned that users generally position their cameras at the start and end of recording the videos such that the first and last frames of the videos are usually more informative than a randomly selected frame. Adding this information in our summarization algorithm is likely to improve our overall performance.

Testing framework limitations: We also applied our algorithm on a more challenging dataset to explore the limitations of our framework. Specifically, we collected 20 user-generated cooking videos from Youtube and downloaded close to $10,000$ images from Flickr for similar queries, all of which related to the activity of making a salad. The clustering result of our algorithm is shown in Figure 8.

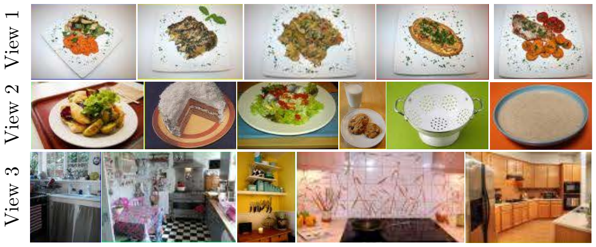

Figure 8: Examples of learned *canonical viewpoints* in the cooking dataset (Section 4.5). These are the highest scoring images in each cluster arranged in a descending order from left to right.

While generally the clusters look reasonable, the overall consistency of the clusters and the number of consistent clusters is smaller than the "Cars and Trucks" dataset. This is because the images in our second dataset have more noise, such as focusing mostly on the cook instead of the food items. Moreover, the large variety and inconsistency in appearance of the food items is hard to capture given the limitations of the state-of-the-art visual features. For these reasons, we found the returned summaries of our algorithm and uniform sampling to be largely similar. Furthermore, there are challenges of domain adaptation [33] when training on images in one setting, and testing the learned models on videos in a different setting. While our current approach has been designed to be unsupervised, it could be made more robust to noise by using more informative visual features, and could benefit from some supervision of training images for more challenging datasets.

5. Related Work

Video summarization has been looked at from multiple perspectives [28]. While the representation used for the summary might be key-frames [34] [12], image montages [3], or short glimpses [26] [25], the goal of video summarization is nevertheless to produce a compact visual summary that encapsulates the most informative parts of a video.

Most of the previous summarization techniques are designed for well-produced videos, and rely on low-level appearance and motion cues [22] [15]. However, they usually do not perform well for user-generated videos, given their generally poor quality and sparse content.

To address these challenges, there have been some recent approaches that take into account interesting objects [17], events [32], user preferences [2], and attention models [21] to summarize videos. Our current work is another step in this general direction of content-aware summarization, where unlike previous approaches, we use web-images as a prior to facilitate summarization of user-generated videos.

The lack of an agreed upon notion of the "optimal" summary of a video can make summary evaluation a key challenge for video summarization. Similar challenges exist in other domains, such as machine translation [24] and text summarization [19], where previous methods have tried to combine several human-generated candidate summaries to infer a final answer which in expectation is better than any of the individual candidate results. Following this approach, there has been previous work in the field of video summarization that also attempts to aggregate multiple summaries of a video to infer a final answer [18] [29] [7]. However they have mostly focused on small to medium scale problems, using expert annotators to label data.

More recently, there has been an interest in the problem of evaluating video summarization results at a large scale [2] [23]. However, these approaches use multiple expert summaries which is an expensive and time-consuming exercise. To this end, methods that rely on crowd-sourcing [27] have mostly focused on the problem of object annotation with either sparse [35] or dense labeling [30]. In this work however, we show how to use a crowd-sourcing model to get multiple summarization labels specifically for user-generated videos.

6. Conclusion & Future Work

In this work, we focused on automatically summarizing user-generated videos. We demonstrated that web images could be used as a prior to summarize videos that capture objects similar to those present in the image corpus. We also focused on the related problem of large-scale automatic evaluation of summarization algorithms. We proposed an evaluation framework that uses multiple summaries obtained by crowd-sourcing, and compared the performance of our framework to that of multiple expert users.

Our main intuition regarding people taking pictures of objects to capture them in an informative way is applicable to videos of events and activities as well. Going forward, we would like to apply our approach on videos of wedding receptions, birthday parties and graduation ceremonies. Moreover, we would like to test if by using this image-based prior, we could identify important viewpoints of an object or event that might have been missed.

References

[1] W. Abd-Almageed. Online, simultaneous shot boundary detection and key frame extraction for sports videos using rank tracing. In *ICIP*, pages 3200–3203. IEEE, 2008. 4

[2] J. Almeida, N. Leite, and R. Torres. Vison: Video summarization for online applications. *Pattern Recognition Letters*, 2011. 7, 8

[3] A. Aner and J. Kender. Video summaries through mosaic-based shot and scene clustering. *ECCV*, pages 45–49, 2006. 7

[4] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. *ECCV*, pages 404–417, 2006. 5

[5] M. Cha, H. Kwak, P. Rodriguez, Y. Ahn, and S. Moon. I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system. In *ACM SIGCOMM*, 2007. 1

[6] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *JMLR*, 2:265–292, 2002. 3

[7] E. Dumont and B. Mérialdo. Automatic evaluation method for rushes summary content. In *ICME*, pages 666–669. IEEE, 2009. 8

[8] M. Everingham, A. Zisserman, C. Williams, and L. Van Gool. The Pascal visual object classes challenge. 2006. 5

[9] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. Liblinear: A library for large linear classification. *JMLR*, 9:1871–1874, 2008. 5

[10] C. Gianluigi and S. Raimondo. An innovative algorithm for key frame extraction in video summarization. *JRTIP*, 2006. 4

[11] F. Glover. Maximum matching in a convex bipartite graph. *Naval Research Logistics Quarterly*, 14(3):313–316, 1967. 4

[12] D. Goldman, B. Curless, D. Salesin, and S. Seitz. Schematic storyboarding for video visualization and editing. *ACM TOG*, 2006. 7

[13] Y. Gong and X. Liu. Video summarization using singular value decomposition. In *CVPR*, volume 2, pages 174–180. IEEE, 2000. 1

[14] M. Huang, A. Mahajan, and D. DeMenthon. Automatic performance evaluation for video summarization. Technical report, 2004. 4

[15] R. Jiang, A. Sadka, and D. Crookes. Hierarchical video summarization in reference subspace. *Consumer Electronics*, 2009. 1, 7

[16] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, volume 2, pages 2169–2178. IEEE, 2006. 5

[17] Y. Lee, J. Ghosh, and K. Grauman. Discovering important people and objects for egocentric video summarization. CVPR, 2012. 7

[18] Y. Li and B. Merialdo. Vert: automatic evaluation of video summaries. In *Multimedia*, pages 851–854. ACM, 2010. 8

[19] C. Lin. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: ACL Workshop*, 2004. 8

[20] C. Liu, J. Yuen, and A. Torralba. Sift flow: Dense correspondence across scenes and its applications. *PAMI*, pages 978–994, 2011. 4, 5

[21] Y. Ma, L. Lu, H. Zhang, and M. Li. A user attention model for video summarization. In *ACM MM*, pages 533–542, 2002. 7

[22] C. Ngo, Y. Ma, and H. Zhang. Video summarization and scene detection by graph modeling. *IEEE CSVT*, 2005. 1, 7

[23] P. Over, A. Smeaton, and P. Kelly. The trecvid 2007 bbc rushes summarization evaluation pilot. In *TRECVID Workshop*, 2007. 8

[24] K. Papineni, S. Roukos, T. Ward, and W. Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, 2002. 8

[25] Y. Pritch, A. Rav-Acha, A. Gutman, and S. Peleg. Webcam synopsis: Peeking around the world. In *ICCV*. IEEE, 2007. 7

[26] A. Rav-Acha, Y. Pritch, and S. Peleg. Making a long video short: Dynamic video synopsis. In *CVPR*. IEEE, 2006. 1, 7

[27] A. Sorokin and D. Forsyth. Utility data annotation with amazon mechanical turk. In *CVPR*. IEEE, 2008. 8

[28] B. Truong and S. Venkatesh. Video abstraction: A systematic review and classification. *TOMCCAP*, 3(1):3, 2007. 7

[29] V. Valdes and J. Martinez. Automatic evaluation of video summaries. *ACM TOMCCAP*, 8(3):25, 2012. 8

[30] C. Vondrick, D. Ramanan, and D. Patterson. Efficiently scaling up video annotation with crowdsourced marketplaces. *ECCV*, 2010. 8

[31] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. *CVPR*, 2010. 5

[32] M. Wang, R. Hong, G. Li, Z. Zha, S. Yan, and T. Chua. Event driven web video summarization by tag localization and key-shot identification. *Multimedia, IEEE Transactions on*, (99), 2011. 7

[33] X. Wang, G. Hua, and T. X. Han. Detection by detections: Non-parametric detector adaptation for a video. In *CVPR*, 2012. 7

[34] W. Wolf. Key frame selection by motion analysis. *ICASSP*, 1996. 7

[35] J. Yuen, B. Russell, C. Liu, and A. Torralba. Labelme video: Building a video database with human annotations. In *CVPR*, 2009. 8