Automatic Video Summaraisation of Casually Captured Videos

Dissertation report submitted in partial fulfillment of the requirements for the award of

Master of Sciences in Mathematics
(with a specialization in Computer Science)
by
Varshaneya V
(Regd.No : 15013)

supervised by
Dr.S. Balasubramanian

DEPARTMENT OF MATHEMATICS & COMPUTER SCIENCE
Sri Sathya Sai Institute Of Higher Learning, Prasanthi Nilayam
25^{th} **March 2017**

Dedicated to The Lotus Feet of my Guru, Eternal Guide, Master and Friend.....

Sri Sathya Sai Institute of Higher Learning

(Deemed University)

DEPARTMENT OF MATHEMATICS & COMPUTER SCIENCE

Certificate

This is to certify that this project report entitled **"Automatic Video Summarisation of Casually Captured Videos"** being submitted by Sri. **Varshaneya V** in partial fulfillment of the requirements for the award of the degree **Master of Sciences in Mathematics (with a specialization in Computer Science)** is a record of bonafide research work carried out by him under my supervision and guidance during the academic year 2016-17 in the Department of Mathematics and Computer Science, Sri Sathya Sai Institute Of Higher Learning, Prasanthi Nilayam campus. To the best of my knowledge, the results embodied in this project have not formed the basis of any work submitted to any other University or Institute for the award of any Diploma or Degree.

Dr. Pallav Kumar Baruah
HOD, D.M.A.C.S,
Sri Sathya Sai Institute of Higher Learning,
Prasanthi Nilayam.

Dr.S.Balasubramanian
D.M.A.C.S,
Sri Sathya Sai Institute of Higher Learning,
Prasanthi Nilayam.

Place: Prasanthi Nilayam
Date: 25^{th} March 2017

Declaration of Authorship

Sri Sathya Sai Institute of Higher Learning

(Deemed University)

DEPARTMENT OF MATHEMATICS & COMPUTER SCIENCE

DECLARATION

The work embodied in this thesis was carried out by me under the supervision of Dr.S.Balasubramanian, in the Department of Mathematics and Computer Science, Sri Sathya Sai Institute of Higher Learning, Prasanthi Nilayam. The work is original and the results embodied in this thesis have not been submitted in part or full to any other University or Institute for the award of any Diploma or Degree.

Varshaneya V

Regd. No. 15013

Date: 25^{th} March 2017.

SRI SATHYA SAI INSTITUE OF HIGHER LEARNING, PRASANTHI NILAYAM

Abstract

Video summarisation is the method of giving compact portrayals of video content through a blend of pictures, video fragments, graphical portrayals and literary descriptors. An ideal video summary should capture the remarkable events and astounding occasions that constitute the video. Summaries are produced by analysing the actual content of the video, along with other associated information and compiling this sequence of keyframes or video-segments that represent essence of the content of the video. Video summarisation techniques can broadly be classified into keyframe based and skims based.

In keyframe based methods, certain frames of the video are selected to summarise the videos. No motion information is available here. Skims based methods produce short video clips to summarise the videos. In this work we have studied and implemented two keyframe based methods and two skims based method for video summarisation. For keyframe based methods, we have focused on clustering techniques including kmeans and Delaunay clustering. For skims based method, we have studied submodular functions and how they can be used to model video summarisation. Further, with respect to skims based summary, we also explored ways of representing videos particularly to capture interestingness and representativeness of the video. These include aesthetic, attention, presence of object features as well as deep features. Our implementation of skims based summary provide meaningful summaries of videos.

Acknowledgements

First and formeost I wish to express my gratitude to Bhagawan Sri Sathya Sai Baba for providing me with a set of amazing teachers and world-class equipments. He is also the source of my inspiration and my *internal* guide. I have crossed many of my hurdles in this endeavour through His guidance. I express my heartfelt gratitude to my mother, who with her tireless efforts, made me what I am today.

This dissertation would not be possible if not for the constant support and guidance of Dr.S.Balasubramanian. He is always ready to give me a patient hearing whenever I approached him for guidance. He was very encouraging and motivating all through this journey. I am very much inspired by his diligence and meticulousness. I would like to express my heartfelt gratitude to him.

My sincere thanks to Dr. Pallav Kumar Baruah, HOD D.M.A.C.S for providing me with ample opportunities, facilities and the ambience in the department to carry out this dissertation. Special thanks to the system administrators Sri. C.Uday Kiran and Sri. Srivarun Vallampatla for their tireless efforts in ensuring the computing facilities of the department go on smoothly.

My thanks are due to my class teacher Sri. Sai Shyam Sharma for his constant support and encouragement. Last but not the least I would like to thank all my classmates for extending their timely help and support to help me complete my thesis.

Contents

Bibliography **44**

List of Figures

List of Tables

Chapter 1

Introduction

Video summarisation is the method of giving compact portrayals of video content through a blend of pictures, video fragments, graphical portrayals and literary descriptors. An ideal video summary should capture the remarkable events and astounding occasions that constitute the video. Summaries are produced by analysing the actual content of the video, along with other associated information and compiling this sequence of keyframes or video-segments that represent essence of the content of the video.

1.1 Aim

Aim is to study the various techniques of video summarisation of casually captured videos and to compare them based on the summaries that they generate. The emphasis is on automatic video summarisation without any parameter tuning or user intervention.

1.2 Motivation

A large amount of video data is generated due to the widespread of tools for video capturing like mobile phones, CCTV cameras, Google Glass etc becoming cheaper. The amount of videos generated to record special occasions and to record day-to-day activities, have increased. Mentality of people which is increasingly popular in the current scenario is "*capture first, filter later*". So it becomes very necessary to get the gist from the videos and remove the unwanted parts.

Also many popular sites like YouTube, Facebook, Twitter, Metacafe, Daily Motion etc. have sprung up. They are encouraging users to upload and share their videos with others. In this context the amount of video data in the internet is increasing exponentially. In order to enable user to access relevant videos easily without spending a lot of time and bandwidth, video summarisation provide a condensed version of the video capturing the important and the pertinent aspects in the video.

Video summarisation can also be integrated with other applications like interactive browsing and searching systems. Video summarisation also form an integral part of video cataloguing where a huge repository of videos can be categorised and classified based on the summaries generated which makes video retrieval easier.

Video summarisation is very useful to those users on low bandwidth network connections. The multimedia data communicated over the network must be more towards information-oriented rather than just sending the actual content itself. This will give the end users a chance to choose their preferred video for complete watching without wasting much of their time in buffering.

1.3 Scope of work done

The summarisation techniques can be broadly categorised into three, namely shot based, keyframe based and skims based. The emphasis here is on keyframe based and skims based. Under the keyframe based technique two methods have been studied and implemented: one uses clustering based on user defined parameter [1] and the other uses clustering without any parameter[2]. Two skims based methods are discussed. The first one [3] selects video segments which are *interesting* based on interstingness score and implements a 0-1 knapsack algorithm to generate summary out of the interesting segments for a fixed budget. The second one [4] is based on optimising mixtures of submodular objectives which are *interestingness*, *representativeness* and *uniformity*. The *interestingness* factor for [3, 4] is learnt from the ground truth prepared by 15 users for a set of 25 videos. Our focus is mainly on perception-based and feature-based summary generation.

This work is divided into 5 chapters and 3 appendices. Introduction is given in the 1^{st} chapter, the 2^{nd} chapter presents a broad survey of variety of techniques available.

The keyframe based techniques and the results that we have obtained are discussed in 3^{rd} chapter and skims based techniques and the results are discussed in 4^{th} chapter. Conclusions and future work are presented in 5^{th} chapter. Appendices A-C consists of a brief discussion on the methods used in skims based summaries.

Chapter 2

Survey of available techniques

The survey paper[5] outlines some of the conceptual frame-work for video summarisation from the literature. Video contents have a three-stage life cycle:

- *Capture*: The video is recorded.
- *Production*: The video is edited to eliminate unwanted and unclear parts.
- *Transformation*: The video is transformed so that it conveys the content or message correctly, and can be viewed by a target audience.

Techniques are broadly classified into 3 categories:

- *Internal methods*: These methods examine the information directly obtained from the video.
- *External methods*: These methods examine the information indirectly obtained from the video.
- *Hybrid methods*: These methods examine a combination of both internal and external information.

Video summarisation assists with the development of compact versions of the complete video through the recognition of its most pertinent content. These summaries can then be added-on to various applications, like interactive surfing and search systems. This offers the users easier methods of handling and accessing digital video content effeciently.

They incorporate variety of cues like key-frames (images from the video), video segment cues (dynamic extension of key-frame cues with audio and motion elements), graphical cues (using visual cues and syntax to supplement other cues) and textual cues (like text caption and subtitles) to get a compact and well represented essence of the video stream. The authors feel that a good summary should have all of these contributing to it. The difficulty of summarising a video is due to the wide spectrum of semantics embedded and latent in a video which can be expressed in various ways, like sounds, images, skims and text. This makes it a challenging area for research. The techniques mentioned perform a domain specific and a non-domain specific analysis. Domain specific analysis refer to techniques that pertain to the specific domain of work and non-domain specific analysis refer to those techniques that are common across domains.

2.1 Internal methods

Internal summarisation techniques are used for videos at the production level of the video lifecycle. These strategies routinely examine raw image, audio and text features of the video directly to analyse semantics suitable for a video summary. Internal summarisation techniques are quite commonly used and have three basic assumptions:

- The analysis is disconnected in space and time from the capture to viewing stages of the video life cycle.
- All analysis and abstraction must be taken from video stream alone.
- The analysis is done automatically thereby avoiding the necessity to rely on user participation.

Image features consists of changes in texture, colour, shape and motion of object taken directly from video. These are used for segmenting a video into shots by identifying the shot boundaries, which include cuts and fades. *Cuts* are represented by sharp changes in image features and *fades* are represented by slower changes in image features. Audio features appear in the audio stream of the video. They include speech, music, sounds and silence. These are quite helpful in figuring out which segments are to be a part of video summary. Speech recognition techniques can be used for identifying and spotting specific semantic detail, which are used to convert snippets of dialogue into textual

representations. Textual features that appear in the video may be in the form of subtitles or text captions. These are embedded in the image stream of the video so they are cues for detecting events.

2.2 External methods

External summarisation methods analyse and utilise information that are not directly part of the video which include user-based information and contextual information. User-based information is the information given by the user. This could be their interactions with, behaviour towards and their understanding of the content of a video stream at the different stages of the video life cycle. This also includes the preference of users. User-based information is obtrusively sourced i.e. user has provided that vital information to aid the summarisation process, for e.g. user sourced information could be in the form of manual annotations of the video as given in SumMe dataset. Contextual information is additional knowledge that is not picked up directly from the user or the video and are unobtrusively sourced without any conscious input of user. For e.g. contextual information gathering devices include GPS (global positioning system) sensors or pressure based floor sensors attached to the video camera to get additional information regarding the environment.

2.3 Hybrid methods

Hybrid summarisation methods that incorporate both external and internal summarisation techniques can be applied to all stages of the video life cycle. They comprise of any combination of internal and external summarisation methods. They prove to be useful mainly for domain specific techniques.

2.3.1 Summaries based interactivity and personalisation

Video summaries are also distinguished according to their interactivity and personalisation and are classified as:

- *Object based summaries* concentrate on a particular object that occur within the video. In this type of summaries, the objects are characterised during the analysis and are used as the basis for formulating summary. These could be text or graphics to characterise the objects within the summary or to help in generating key-frames.

- *Event based summaries* concentrate on a particular event that occur in the video. For e.g. goals or penalty kicks form very interesting part of a football video and detection of those events become vital for the summary.

- *Perception based summaries* are those that are generated based on how user perceives a particular video. They focus more on users' inference from the video identifying tangible object in the video.

- *Feature based summaries* objectively generate summaries by analysing the low-level features like colour changes, texture changes and/or motion changes in the stream or snippets of silence or speech in audio stream. The key point here is that these type of summaries do not attempt to infer object, event or perception based on the semantics of these features. Summaries that are generated, are shown as a series of key frame images or skims that may or may not be accompanied by any description. These summaries clarify neither the meaning of the video nor the reason as to why the segments have been selected as summary.

Video summaries can also be considered as a tool in the hands of the user for their necessities (interactive or static, personalised or generic). This paper [5] has also given an outline on how to summarise based on these techniques though the finer details are not given. The quality of summaries of each technique are also discussed theoretically. This paper [5] has a good set of references which could be used to develop or improve upon on the techniques.

With regard to our work in this thesis, both the keyframe based techniques described in [1, 2] fall into the category of **internal methods** where only the video is given as input to the algorithm. The algorithm uses only the visual information of the video in the form of *hue* component of HSV histogram. The skims based technique in [3] is an example of **external method** where only users' perception is used. But technique in [4] fall into the category of **hybrid method** where both users' perception and image features are incorporated. Users' perception form the *ground truth* for interestingness factor in

both the techniques [3, 4]. In the case of [4], for extracting the representativeness of the video, the deep features of each frame obtained from 6^{th} hidden layer of AlexNet [6] are used as image features.

Chapter 3

Keyframe based summaries

The main aim of keyframe based techniques is to generate frames that are representative of the whole video. There are many ways of generating key-frames and one straight forward way is to generate them using clustering techniques. Frames are sampled and feature vectors are generated. The generated feature vectors are clustered and the median of each cluster is the keyframe and together all the key-frames form the summary. The summary is just still-images without temporal coherence or motion feature.

3.1 K-means clustering based generation of keyframes

The paper titled *Video Summarization Using Clustering*[1], is a keyframe based summarisation method where k-means is used for clustering feature vectors to get the centroids which are representative of the feature vectors.

K-Means clustering algorithm Inititally k random centroids are selected. The distance of centroid from each of the data point is calculated. Each point is then assigned to the cluster to which it is closest to, minimising the error function. For the clusters that are created new centroids are found. The centroids are updated if there is a change within a tolerance limit. Otherwise the centroids converge to the actual cluser centroids. The Euclidean distance is used as the error function to be minimised in the k-means

algorithm.

$$Error = \sqrt{\sum_{j=1}^{k}\sum_{i=1}^{N}(x_i - c_j)^2} \tag{3.1}$$

where k is the number of clusters specified, c_j is the centroid of the j^{th} cluster and x_i is i^{th} point.

3.1.1 Generating summaries using k-means algorithm

Individual frames are sampled in the ratio 1:10 i.e. 1 frame in every 10 frames. This is a very good sampling rate since videos generally have 30 fps (frames per second) and a lot of frames are redundant. Then the HSV histograms of these are calculated and dimensionality reduction is done by principle component analysis to obtain the feature vectors. Histogram compresses the information in a frame to a vector and the hue part of HSV colour histogram gives a view that a human eye sees. The feature vectors thus obtained are clustered using k-means clustering technique. Frames with similar visual content are clustered together. The median of each cluster is a well-represented frame of each cluster. The summary of the video is all the key-frames put together and is a terse representation of the video.

3.2 Delaunay clustering based generation of keyframes

3.2.1 Motivation

As an improvement over k-means clustering, the paper titled *Key frame based video summarisation using Delaunay clustering*[2] makes use of Delaunay triangulation to create Delaunay diagram. The summary generated is of very good quality with fewer redundant frames (an improvement over the previous technique). This clustering algorithm is automatic i.e. it does not require user intervention in terms of tuning parameter and is suitable for processing batches of videos. Though temporal order is not maintained, redundancy is eliminated. The reason for using delaunay triangulation over other types of triangulation is that it maximises the minimum angles of the triangle, making the triangles tend towards equiangularity which avoids skinny triangles. This also tries to

avoid distortions. Delaunay triangulation captures the spatial information of the feature vectors and the running time is $O(n * logn)$. Integration with PCA reduces the dimensionality and hence the running time of the algorithm.

3.2.2 Clustering algorithm

The process of creating feature vector from the video is same as described in the previous section. The feature vectors are points in the feature space. So the Delaunay triangulation of these points would be the dual of the Voronoi diagram constructed on these points. In other words, two points a and b have an edge connecting them in Delaunay diagram if and only if they share a common boundary in the Voronoi diagram as shown in figure 3.1. Once the diagram is constructed the edges are categorised into two classes: *inter-cluster edge* or separating edge and *intra-cluster edge* or short edge. The separating edges are removed to form the Delaunay clusters. Now for each of these clusters the median will give the keyframe. The main idea behind detecting inter-cluster edges is that these edges show a greater variability in edge lengths or in other words, inter-cluster edges are longer than intra-cluster edges. For each point in the Delaunay triangulation, local mean and local standard deviation are determined after the length of each incident edge is calculated. A list of formulae given below indicate how the inter- and intra-cluster edges are identified.

The mean length of edges incident at each point p_i is given by

$$localMeanLength(p_i) = \frac{\sum_{j=1}^{d(p_i)} \|e_j\|}{d(p_i)}$$

where $d(p_i)$ denotes the number of edges incident at the point p_i and $\|e_j\|$ denotes the length of each edge e_j.

The local standard deviation of length of edges incident at point p_i is given by

$$localStandardDeviation(p_i) = \sqrt{\frac{\sum_{j=1}^{d(p_i)} localMeanLength(p_i) - \|e_j\|^2}{d(p_i)}}$$

The global standard deviation is denoted by

$$globalStandardDeviation = \frac{\sum_{i=1}^{N} localStandardDeviation(p_i)}{N}$$

where N is total number of points.

A short edge or intra-cluster edge is defined as

$$shortEdge(p_i) = \{e_j : \|e_j\| < localMeanLength(p_i) - globalStandardDeviation\}$$

A separating edge or inter-cluster edge is defined as

$$separatingEdge(p_i) = \{e_j : \|e_j\| > localMeanLength(p_i) + globalStandardDeviation\}$$

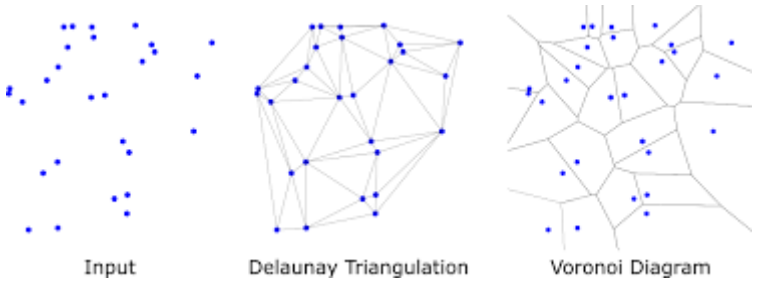

FIGURE 3.1: Example of Delaunay triangulation

3.2.3 Generating summaries using Delaunay clustering algorithm

The pre-sampling and feature extraction methods remain the same as that discussed in the previous section. These features are then fed to the delaunay clustering algorithm to get the key-frames. The user need not specify the number of keyframes beforehand thereby making it suitable for batch processing of videos.

3.2.4 Evaluation of results based on metrics

The metrics for evaluation of the summary which are used by the authors in [2] are:

- *Significance factor* for each keyframe gives a score to it corresponding to the size of the cluster it came from. The greater the significance factor, the bigger is the cluster that the keyframe has come from.

$$significanceFactor(l) = \frac{C_l}{\sum_{j=1}^{k} C_j} \tag{3.2}$$

where C_l is number of frames in cluster l and k is total number of frames in the video.

- *Compression factor* for each video is an indication of the reduction in size with the summarised content as compared to the original set of frames. Larger the compression factor more concise is the summary.

$$compressionFactor = \frac{k}{N} \tag{3.3}$$

where k is number of key-frames and N is the total number of frames processed.

- *Overlap factor* quantifies the extent of overlap between the summary generated by the algorithm and one generated by user. The greater the overlap factor, higher is the coverage of the summary.

$$overlapFactor = \frac{\sum_{k \in commonKeyFrameCluster} C_k}{\sum_{j=1}^{k} C_j} \tag{3.4}$$

The results are tabulated in table 3.1 after evaluating the aforesaid metrics for the following videos:

Video name	Number of clusters	Significance factor	Compression factor	Overlap factor
America's new frontier segment 4	4	0.173 0.3946 0.3054 0.1081	0.1081	100
America's new frontier segment 10	4	0.1432 0.334 0.3963 0.1245	0.083	100
America's new frontier segment 3	6	0.1157 0.088 0.1065 0.1389 0.1528 0.0139	0.3704	100
The voyage of Lee segment 15	3	0.4802 0.4626 0.0573	0.1322	94.27

Video name	Number of clusters	Significance factor	Compression factor	Overlap factor
Future of energy gases segment 3	2	0.785 0.2082	0.0683	100
Drift ice as an geological agent segment 10	3	0.5071 0.2429 0.2143	0.2143	100
Drift ice as an geological agent segment 7	3	0.0667 0.2872 0.3897	0.1538	100
Drift ice as an geological agent segment 5	4	0.055 0.5183 0.1284 0.1789	0.1835	100
A new horizon segment 8	4	0.0994 0.4862 0.1271 0.0387	0.3315	23.98

TABLE 3.1: Metrics judging the quality of the summary produced by delaunay clustering based summarisation

3.3 Discussion

We have summarised some of the videos from **The Open Video Project** using both the techniques from [1, 2]. The hardware configuation of the computer that we have used for this purpose are 8GB RAM and Core 2 Duo processor. We have used computer vision toolbox in MATLAB.

3.3.1 K-means clustering generated summaries

Test videos used were taken from `https://open-video.org`. The parameter k needs to be specified everytime and the user has no idea about it when it comes to an unknown video. There are no closed form formulae for determining k. The summary for a couple videos from open-video database are shown in Figure 3.2 and Figure 3.3.

FIGURE 3.2: Summary generated by k-means algorithm with $k = 6$ for the video "America_ New_ frontier_ Segment 10.mpeg".

FIGURE 3.3: Summary generated by k-means algorithm with $k = 5$ for the video "America_ New_ frontier_ Segment 3.mpeg". Note that there is a background frame and a repetition in the summary

Advantages and drawbacks The **main advantage** is that summaries are generated quite fast and algorithm is simple. The **main drawback** is there is a parameter which needs to be tuned which is the number of clusters k. Other drawbacks include repetition of keyframes in the final summary and in those videos where background appears for a longer time than the content, frame(s) containing background tend(s) to appear many times in the summary. This algorithm is not suitable for batch processing of videos. Also it has been observed that with increase in k there are a lot of redundant keyframes.

3.3.2 Delaunay clustering generated summaries

Same set of test videos from OpenVideo are used here as used in previous subsection. The summaries shown below in figures 3.4 and 3.5 are generated by delaunay clustering:

FIGURE 3.4: Summary generated by delaunay clustering for the video "America_ New_ frontier_ Seg10.mpeg"

FIGURE 3.5: Summary generated by delaunay clustering for the video "America_ New_ frontier_ Segment 3.mpeg".

FIGURE 3.6: Summary generated by delaunay clustering for the video "nasaanni001.mpeg"

Advantages and drawbacks *Advantages* include its usefulness for batch processing of videos, as the user need not specify any parameter for generating summaries and it can generate concise summaries. The main *drawback* is that for some videos the necessary number of keyframes may not be generated. Take the case of "nasaanni01.mpeg" which portrays achievements of NASA on their 25^{th} anniversary like building and repairing of USS, riding rover on the moon, probe sent to Saturn and blast of their rocket. The summary just had only one keyframe shown in the figure 3.6.

3.3.3 Comparison

In order to compare the summaries generated by both the algorithm we give the value for k in k-means as the number of clusters generated by delaunay clustering. The comparison is given below:

FIGURE 3.7: The summaries to the left are from delaunay clustering and to the right are from k-means clustering for the video "America_ New_ Frontier_ Seg10.mpg"

FIGURE 3.8: The summaries to the left are from delaunay clustering and to the right are from k-means clustering for the video "America_ New_ Frontier_ Seg4.mpg"

From figures 3.7 and 3.8 it is clear that the summaries of k-means and of delaunay clustering are quite similar to each other when value of k is equal to number of clusters

generated by delaunay clustering. It also validates the keyframes generated by delaunay clustering. In that way this becomes a proof of correctness of the approach towards generating keyframe summaries using delaunay clustering.

Chapter 4

Skims-based summaries

Creating skims that summarise the content automatically, with the summary still conveying the story of the initial video, is quite challenging. Here two innovative methods [3, 4] are discussed in this section. These methods are more applicable to casually captured videos. Though keyframes provide the underlying content of the video, the "motion part" of the video is lost when only still frames are used to for summarising. The user is deprived of *"video experience"* while viewing the summary. So generating summaries by using snippets of video or short clips also called *video skims* are more preferred.

The summary generated by the computer should be *"interesting"* to the human. This means the computer should know what a human would find interesting in a video and then look for such features while summarising. Hence supervised summarisation techinques are very much encouraged. One such data set used for learning human perception of a good summary is *SumMe* dataset [3]. It consists of short user videos each annotated with scores for video segments. More than 15 users have given these scores to segments that they feel should be summary, with the users coming from different age and gender groups. The videos cover a wide range of themes from events to holidays to sports. The summaries generated by the methods are video clips with the difference being the frames which are chosen by these algorithms [3, 4]. We have chosen 5 videos for testing both the methods and comparing the results. It consists of 3 videos from open-video repository, a drone video and a video in which people play games.

4.1 Creating summaries using interestingness factor

The paper[3] by Gygli et al proposes a new approach for video summarization. The videos of the SumMe dataset[3] contain raw videos with user annotations. This method uses a segmentation technique called *superframe segmentation* (which is discussed in detail in the Appendix C), specifically crafted for raw videos. When a new video is input to the algorithm[3], the video is segmented using *superframe segmentation* and interestingness is calculated for each of the segments and then the most interesting segments are selected using 0-1 knapsack algorithm as given in the Figure 4.1. Visual interestingness for each of the frame is calculated using the set of features described below.

- **Attention:** A score for human attention is calculated based on spatial and temporal saliencies as adapted from [7]. Instead of combining the two saliencies in a non-linear manner, we have used the two saliencies separately as two different features.
- **Aesthetics:** There are different criteria for calculating aesthetics of an image. But the authors have preferred colourfulness [8], contrast [9] and distribution of edges [9] for computing the aesthetics of a frame.
- **Faces and persons:** Faces and persons are detected in a frame using the *Viola-Jones detector* [10] in MATLAB ®. This gives a bounding box around different faces and persons if they are present in a frame. The relative area of the bounding box to that of area of frame gives the score for this particular feature for each frame.

4.1.1 Learning

The features are combined with a linear model, where the weight w is regressed from the annotated SumMe dataset. Suppose we have N frames in a video, the score of interestingness for a frame is calculated by the formula 4.1:

$$i_k = w_0 + \sum_{i=1}^{N} w_i \cdot u_i + \sum_{i=1}^{N} \sum_{j=i+1}^{N} w_{i,j} \cdot u_i u_j \tag{4.1}$$

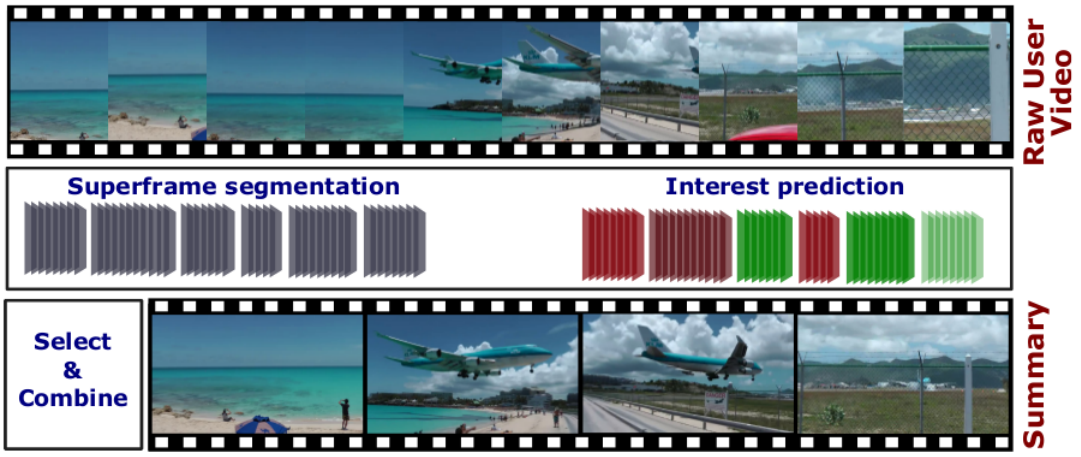

FIGURE 4.1: Methodology for creating summaries

where u_i is the score of feature i and w_i is the weight corresponding to feature i. This model quite is simple and free from overfitting. w is estimated using least-squares from the ground truth in the SumMe dataset. For training, 100 frames are randomly chosen from each of the training video and concatenated to be used for regressing w. This is to ensure that all the training videos are given equal importance. This process is repeated 50 times and then averaged to obtain the resulting weight vectors w.

4.1.2 Generating summary

The total interestingness $I(S_i)$ score of each of the superframe S_i is the sum of interestingness score of each of the frame in it given in formula 4.2:

$$I(S_i) = \sum_{k=n}^{m} i_k \tag{4.2}$$

To select the optimal set of superframes for representing the video, a 0-1 knapsack optimisation is done on the superframes with the value of the superframe to be its interestingness score $I(S_i)$ and its weight to be the number of frames in it i.e. its length $\|S_i\|$. The optimisation is subject to a budget B which is the fraction of frames required in the summary.

$$\max_x \sum_{i=1}^{n} x_i I(S_i) \tag{4.3}$$

subject to

$$\sum_{i=1}^{n} x_i \|S_i\| \leq L_s \tag{4.4}$$

where $x \in \{0, 1\}$ with $x_i = 1$ indicating that the superframe i is selected and L_s is the desired length of the summary. The final summary consists of concatenation of superframes selected by the knapsack algorithm.

4.1.3 Results

The paper [3] originally had proposed 5 interestingness criteria, but we have only chosen 3 aforesaid criteria. Graphs 4.2 and 4.3 below show the closeness of interesting frames in the ground truth and those predicted using the regressed weights for the video "Cooking.mp4". Peaks in the both the graphs 4.2 and 4.3 corresponding to a frame indicates that the frame is interesting. Though expecting the algorithm to match exactly the human evaluation is insane, frames around what human evaluator has shown to be interesting has been picked up by the algorithm.

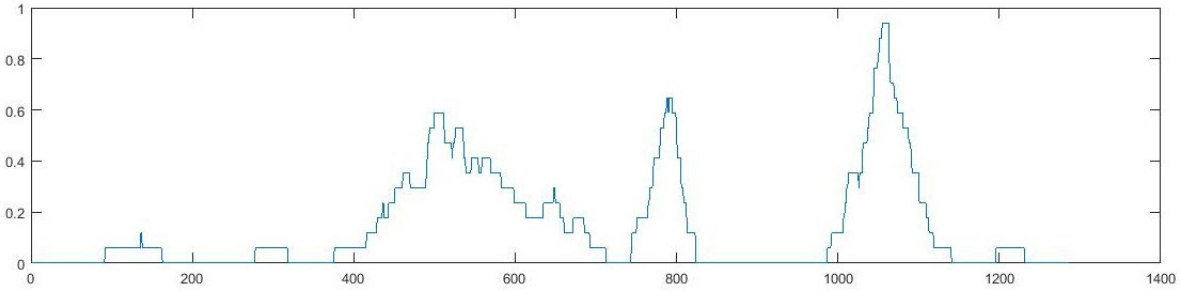

FIGURE 4.2: Plot of interestingness score vs frame number as given in the ground truth of video "Cooking.mp4"

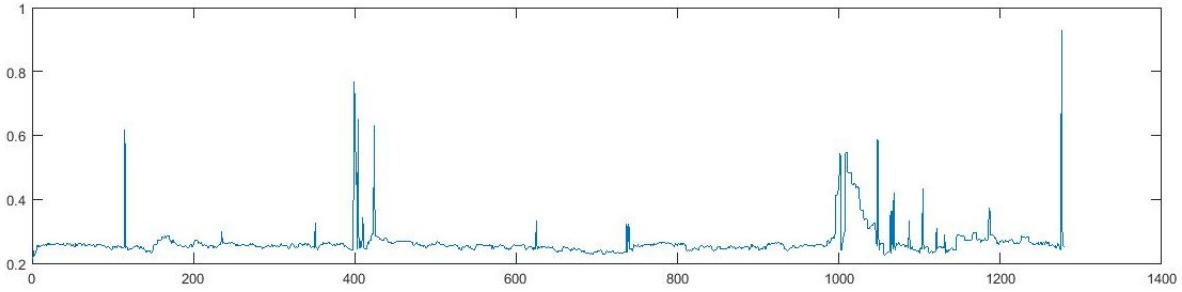

FIGURE 4.3: Plot of interestingness score vs frame number as obtained from learnt weights for the video"Cooking.mp4"

The time of run for the test videos are tabulated in table 4.1:

Video name	Video time (sec)	Summarisation time (Hr)
nasaani.mpg	30	0.39
America_ New_ Frontier_ Seg4.mpg	123	1.62
America_ New_ Frontier_ Seg10.mpg	161	2.15
games.avi	135	2.46
drone.mp4	253	3.75

TABLE 4.1: Summarisation time for videos using only interestingness criteria

4.2 Creating summaries by jointly optimising multiple objectives

4.2.1 Motivation

The participants who annotated the SumMe dataset chose the interesting parts over representative parts which has lead to a situation where the entire video is not represented but only the interesting parts. In the video "games.avi", there are cricket and football being played simultaneously. The summary generated by the previous method only focussed on cricket leaving out football which shared a greater part in the video. Gygli et al have developed a supervised method [4] to learn and jointly optimise multiple objectives of a "good summary". The objectives are submodular functions and they are interestingness, representativeness and uniformity. Submodular functions have *diminishing returns* property and are used in many of computer vision applications. A set function $f : \wp(V) \rightarrow \mathbb{R}$ is said to be **submodular** if given sets $T \subseteq U \subseteq V \setminus \{s\}$ where $s \in V$ then $f(T \cup \{s\}) - f(T) \geq f(U \cup \{s\}) - f(U)$. Submodular functions are extensively used in computer vision. A brief outline of the procedure used in this paper [4] is shown in the Figure 4.4.

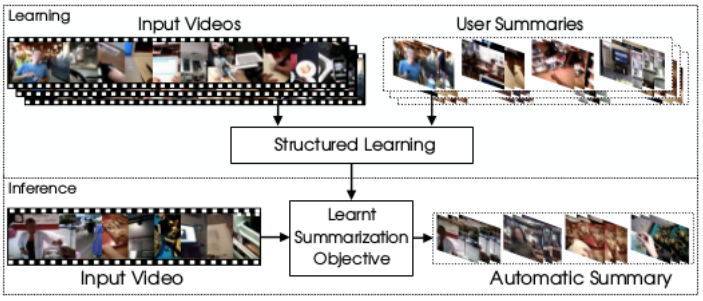

FIGURE 4.4: Generating summaries by learning mixtures of submodular objectives

4.2.2 Formulation

Video summarisation can be modelled as a subset selection problem. Exact solution of a subset selection problem is NP-hard. Hence we look for approximate solutions which are very near to the original solution. This has led to Gygli et al to apply *lazy greedy* optimisation technique for learning and optimising submodular objectives (refer Appendix C for a detailed discussion) i.e the optimisation technique finds greedy solutions with lazy evaluations to speed up the optimisation.

Given a video V and a budget B, let Y_V denote all possible solutions of $y \subseteq V$ given the constraint $|y| \leq B$. We need to find a subset $y^* \in Y_V$ that maximises the objective function o. So the task of video summarisation is to select a summary y^* such that

$$y^* = \arg\max_y o(x_V, y) \tag{4.5}$$

where $y \in Y_V$. Here x_V are all the features extracted from the video. $o(x_V, y)$ is defined as a linear combination of submodular objectives:

$$f(x_V, y) = [f^{int}(x_V, y), f^{rep}(x_V, y), f^{uni}(x_V, y)]^T \tag{4.6}$$

where f^{int}, f^{rep} and f^{uni} represents interestingness, representativeness and uniformity respectively.

The objective function is given by:

$$o(x_V, y) := w^T f(x_V, y) \tag{4.7}$$

where the entries of w are positive.

Linear combinations of submodular functions with positive coefficients are also submodular functions. Y_V grows exponentially or in other words subset selection is a NP-hard problem. Hence solving equation 4.7 quickly is difficult. Therefore as mentioned we resort to lazy greedy optimisation of the same. The method consists of two stages namely supervised learning and summary generation which are discussed in the next two subsections.

4.2.3 Supervised learning

The intuition behind this learning process is that human reference summary should be preferred more than any other summary. Given T pairs of videos and their summaries (V, y_{gt}), the weights in the vector w of equation 4.7 needs to be learnt. So the expression in 4.8 must be optimised:

$$\min_{w \geq 0} \frac{\sum_{t=1}^{T} L_t(w) + \frac{\lambda \|x\|^2}{2}}{T} \tag{4.8}$$

where $L_t(w)$ is the generalized hinge loss of training example t given by

$$L_t(w) = \max_{y \subseteq Y_V^{(t)}} (w^T f(x_V^{(t)}, y) + l_t(y)) w^T f(x_V^{(t)}, y_{gt}^{(t)}) \tag{4.9}$$

Here superscript (t) is used refer to both features and subsets of video t. And for the margin a recall loss similiar to the one used in text summarisation [11] is used here:

$$l_t(y) = \frac{1}{B}.(\|y\| - \|y \cap y^{(t)}\|), \tag{4.10}$$

$l_t(y)$ is a count of how many of the candidate summary y are not represented in the ground truth, normalized by the maximal length of the summary.

This method generates skims and hence summary is considered to be a set of segments. The submodular objectives are defined as:

Interestingness Summaries generated must be interesting. A computer does not know what is interesting to a user. Hence interestingness score is calculated for each frame by learning from annotated summaries in SumMe dataset as mentioned in the previous section. The interestingness score $I(k)$ is to be predicted locally for each frame k given features x_k. Interestingness objective f^{int} is given by:

$$f^{int}(x_V, y) = \sum_{k \in \cup s, s \in y} I(k) \tag{4.11}$$

The defined function is a weighted coverage function and is submodular.

Representativeness This indicates how well a summary represents initial video. Finding the best k segments to represent a dataset is known as the the k-medoids problem and the objective is to select mediods that will minimise the sum of squared error

$$L_r(x^r, y) = \sum_{i \in V} \min_{s \in y} \|x_i^r - x_s^r\|_2^2 \tag{4.12}$$

where x_r are the deep features used to represent a segment. The k-medoid objective can be reformulated as a submodular objective as follows:

$$f^{rep}(x_V, y) = L_r(x^r, \{p'\}) \text{ - } L_r(x^r, y \cup \{p'\}) \tag{4.13}$$

where x_i^r is the deep feature for i^{th} frame and p' is a phantom exemplar, necessary to avoid taking the minimum over an empty set in equation 4.12.

The representativeness of the segments were extracted using deep features taken from layer 6 of AlexNet [6]. Caffe [12] implementation of AlexNet was used since DeCAF[13] has been deprecated (refer to Appendix B for a discussion on AlexNet and Caffe). The AlexNet model used here was already trained on image-net dataset. This is done for all the frames in the video. This feature is row vector of size 4096.

Uniformity A good summary should be uniform without large jumps and must maintain temporal coherence. There is a risk of redundancy with a summary having many temporally adjacent segments. The features x^u are frame numbers and a segment is represented by its mean frame number. The submodular function representing uniformity is

$$f^{uni}(x_V, y) = L_r(x^u, \{p'\}) \text{ - } L_r(x^u, y \cup \{p'\}) \tag{4.14}$$

Given pairs of videos and their user created summaries as training examples, a combined objective was learnt using package "gm_ submodular" package [4] implemented in python, for learning the weights of objective functions in the equation 4.7. This package is provided by the authors of [4].

4.2.4 Generating summary

Once the weights were found, 4.8 was optimised using MATLAB® toolbox for Submodular Function Optimization[14] which is an implementation of lazy-greedy algorithm for submodular functions. When a new video is input, this method creates summaries that are interesting, representative and uniform. MATLAB® has got a good facility for reading and writing videos. With this facility reading video into frames for processing and also for writing the output in the form of video becomes user-friendly.

4.2.5 Results and disscussion

The weights for objectives in equation 4.7 were learnt by optimising equation 4.8 using the ground truth provided for the videos in the SumMe dataset. Python package *gm_submodular* was used for this purpose. These weights are tabluted in table 4.2.

Objective	Weight
Interestingness	0.98619
Representativeness	0.00002
Uniformity	0.01379

TABLE 4.2: Weights obatined for different objectives in 4.7

The time of run for the test videos are tabulated below:

Video name	Video time (sec)	Summarisation time (Hr)
nasaani.mpg	30	0.69
America_ New_ Frontier_ Seg4.mpg	123	1.74
America_ New_ Frontier_ Seg10.mpg	161	2.34
games.avi	135	6.06
drone.mp4	253	10.55

TABLE 4.3: Summarisation time for videos using optimisation of submodular mixtures

It is clear from tables 4.1 and 4.3 that the submodular optimisation takes a lot of time than 0-1 knapsack optimisation. For videos of almost similiar length, say for example "America_ New_ Frontier_ Seg4.mpg" and "games.avi", though the difference in number of frames is around 10, submodular optimisation is twice slower than the knapsack optimization. Hence the frame width and height played a very crucial role in deciding summarisation time. This is because a larger frame size would demand a larger RAM for storage and larger CPU processing time.

We have also observed that superframe segmentation is highly resource-intensive algorithm which requires a very large amount of RAM. For the type of test and training videos that we used 30GB of RAM was required. Hence one of the future work could be to propose a video segmentation technique that uses less RAM and yet gives similiar segments. One workable solution we have explored is to reduce the frame width and height to one-fourth of their original values. The number of superframes does not change but there is slight shift in the boundaries of these superframe by a few frames. This is good trade-off when compared using 30GB of RAM for processing the videos.

We can say that from the weights that are learnt from the ground truth, importance given to *interestingness* is around 98% and that given to uniformity and representativeness is very meagre. Therefore having all this formulation only leads us to the conclusion that humans prefer *interesting* summaries.

Method of generating summaries	f-measure (%)	recall (%)
Random	18.95 ± 0.06	43.72 ± 0.14
Interestingness	20.3 ± 0.06	59.62 ± 0.19
Uniformity	17.96 ± 0.08	38.04 ± 0.22
Representativeness	19.04 ± 0.04	46.58 ± 0.11
Combination of 3 objectives	22.31 ± 0.08	58.43 ± 0.21

TABLE 4.4: f-measure and recall values for different methods of summarisation

The f-measure and recall values for different summarisation objectives that we have obtained are tabulated in Table 4.4. The f-measure and recall values in Table 4.4 match with those obtained by Gygli et al in [4] qualitatively. The magnitude of f-measure values in Table 4.4 are less than those in [4], whereas magnitude of recall values in Table 4.4 are greater than those in [4].

Table 4.4 also emphasizes this point in terms of f-measure and recall only slightly improving with addition of representativeness and uniformity. We wanted to cross check with respect to visual perception in terms of summary produced. Figures 4.5 and 4.6 below show some keyframes across the summaries for the sports video "games.avi" produced by considering only the interestingness (Figure 4.5) and all the three factors (Figure 4.6) respectively. Clearly, interestingness alone does not cover the entire video as most of the frames covering football is dropped (which is more pertinent) as shown in Figure 4.5. The summary generated by the submodular formulation has a greater coverage as seen in Figure 4.6. This elucidates the point that a small numerical improvement in terms

of f-measure and recall achieved when considering all the three factors is visually very significant.

FIGURE 4.5: Some keyframes from summary of "games.avi" generated by considering only interestingness

FIGURE 4.6: Some keyframes from summary of "games.avi" generated by considering interestingness, representativeness and uniformity

Chapter 5

Conclusion and future work

5.1 Conclusion

K-means clustering algorithm requires the parameter k for clustering which is not known beforehand for generation of summary. Nevertheless it is quicker in generating summaries than delaunay clustering. In order to overcome the problem of parameter tuning we moved on to delaunay clustering. This can be used for batch processing of videos. K-means and delaunay clustering select similiar frames for summary when the value of k for k-means clustering equals to the number of clusters generated by delaunay clustering. Keyframe based summaries are just still images without the *motion component*.

So we moved on to skims based summarisation to produce summaries which are themselves video-clips. In general users want summaries that are *interesting* and "interestingness" is highly subjective and supervised approach is necessary for computer to learn what is interesting for the user and summarise accordingly. In such a approach we found that in some videos pertinent parts did not appear in the summary. Hence we switched on to optimising mixture of submodular objectives, with the objectives being interestingness, representativeness and uniformity. The weight for each objective is learnt from the ground truth. In such a formulation we found that representativeness and uniformity along with the most influential interestingness add value to the visual perception of the summary generated.

It is very hard to exactly define what a good summary is. The idea of a good summary varies from person to person. An ideal summarisation algorithm must be very adaptive

to the preference of each individaul. Length of summary also plays a very important role. A long summary tends to focus on representing various content of the video whereas short summaries focus more on interestingness. Also interestingness varies from domain to domain. It is very difficult to develop a framework that can capture interesting events across all the domains. In a particular domain, interesting features can be hand-crafted for generating better summaries. For e.g. in summarising cricket videos events like catches, boundaries, shouts (audio features) etc are indicators of interesting events. It all leads to the conclusion that a summary is a "good summary" only in the eyes of a particular user and cannot be generalised for a larger audience.

5.2 Future work

All our analysis is done considering video to be just a sequence of images without the audio part. One future work could be to analyse audio features for better video summaries. In this regard we could summarise discourses of Bhagawan Sri Sathya Sai Baba by incorporating the audio features as well. Darshan videos of Bhagawan can be summarised by egocentric methods of summarisation. Such work could be a contribution to RadioSai which has a huge repository of Bhagawan's videos. Interestingness is highly subjective, so personalised video summaries can be generated as per the users' viewing habits. This would ensure that each one gets to see what he/she finds interesting. The same algorithm can learn from each viewer's past history of viewing instead of learning from a dataset with limited number of annotated summaries. Another very challenging future work would be summarise videos in their compressed form without decompressing them.

Appendix A

Interestingness criteria

A.1 Attention

Visual attention[7] is given much importance owing to the theories of human attention that have already come up. Though this feature is used quite often in key-frame based summarisation, it is used here because interesting frames tend to have a higher human attention factor. Attention can be broadly classified into two namely, static saliency and temporal saliency. The non-linear combination of them gives the combined attention value. The idea behind this concept comes from the research on neuro-biological system on how the brain and the visual system work in tandem to locate salient regions in an image or a video. The exact mechanisms of the human ability to concentrate on the certain areas, have not yet been understood fully. But nevertheless there are some theories regarding the same. The bottom-up approach towards visual attention is realised in the form of visual saliency. In layman terms visual saliency is a measure of the extent to which an area is different from its neighbourhood. Hence as criteria for interestingness, static and temporal saliencies are used.

A.1.1 Static saliency

or spatial attention value is based on an image descriptor called "image signature". Image signature can be thought as approximately as the foreground of the image. Also the foreground of an image attracts a viewer more than the background. A frame F

from a video is resized to size 64x48 and then the image signature $IS(F_c)$ for a colour channel c is defined as:

$$IS(F_c) = sign(DCT(F_c)) \tag{A.1}$$

The image signature is then transformed back to the spatial domain by taking its inverse deiscrete cosine transform to obtain the reconstructed image F'_c

$$F'_c = IDCT(IS(F_c)) \tag{A.2}$$

The static saliency map $S(F_c)$ for F_c is calculated by

$$S(F_c) = G * \sum_c F'_c \circ F_c \tag{A.3}$$

where $*$ denotes convolution, G is the Gaussian kernel for smoothening and $\circ$ is the Hadamard product operator. The Gaussian kernel for a pixel (i, j) is defined as:

$$G(i, j) = \frac{1}{2\pi\sigma} \exp^{-\frac{i^2+j^2}{2\sigma^2}} \tag{A.4}$$

σ is taken to be 0.045. The saliency map over each colour channel is added to get overall static saliency.

A.1.2 Temporal saliency

Motion in a video captures human attention more quickly and computationally more cheap. Motion attention have become a very important feature in many of video summarisation techniques. Motion gradients are quickly obtained through temporal gradients. Temporal gradients calculate the change in the pixel values across the frames and also serve as estimators of changes in motion component of the video. Suppose we have two frames $F(t)$ and $F(t - \tau)$ which are taken from the video at times t and τ respectively. The temporal contrast of a pixel p with its neighbour q is given by

$$TC_{p.q}(t) = |F_p(t) - F_q(t - \tau)| \tag{A.5}$$

with $F_p(t)$ and $F_q(t-\tau)$ being the intensity values of the pixels p from frame $F(t)$ and q from frame $F(t-\tau)$. The gradient vector at a pixel p is given by

$$T_p(t) = \{TC(t)_{p,r}\}, \forall r \in N_{t-\tau}(p) \tag{A.6}$$

where $N_{t-\tau}(p)$ is a 5x5 neighbourhood of pixel corresponding to p in frame $F(t-\tau)$. Temporal saliency at each pixel p is then computed by using sum of absolute differences between the temporal gradients of 5x5 neighborhood $N_t(p)$ in frame $F(t)$:

$$TS_p = \sum_{s=1}^{N_t(p)} |T_p(t) - T_s(t)| \tag{A.7}$$

After having computed the saliency value at each pixel the temporal saliency map $TS(F)$ is obtained which is then normalised in range of $[0, 1]$.

Gygli et al [3, 4] have combined both the saliencies in a non-linear fashion as mentioned in [7], into a single feature for interestingness. We have used them separately as two different features of interestingness. Human attention score based on spatial saliency is calculated for each frame and for every 5^{th} frame in the case of temporal saliency. The results that are obtained are discussed in the "Implementation and results" section.

A.2 Aesthetics

Choice of aesthetics is highly subjective. So going with the selection of aesthetic features given by Gygli et al in the paper[3] we have the following features for describing the interestingness. The concept of contrast and spatial distribution of edges are adapted from[9] and colourfulness of an image from [8].

A.2.1 Contrast

Contrast is computed as the middle 98% gray level mass of the image's gray level histogram as shown in Figure A.1.

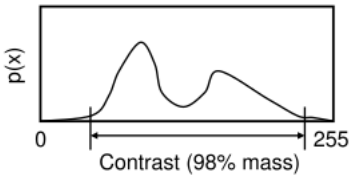

FIGURE A.1: Contrast of an image

A.2.2 Quality

Distribution of edges is a key factor for recognising the whether a photo is professionally shot or not. The subject is well focused in professional photos and a region could be found where the maximum number of edges are present. Whereas in other photographs the edges are distributed across the whole of the image. To implement the edge spatial distribution feature, first a 3x3 laplacian filter with $\alpha = 0.2$ is applied and the absolute value is taken to ignore the direction of the gradients. In case of colour images this is done to each of the components and averaged out to get the laplacian image L . This laplacian image is resized to 100x100 and is normalised such that the image sum is 1. The area of the bounding box enclosing top 96.04% of the edge energy needs to be obtained. The area of the bounding box is calculated by projecting laplacian image L onto x and y axes independently:

$$P_x(i) = \sum_y L(i, y) \tag{A.8}$$

$$P_y(j) = \sum_x L(x, j) \tag{A.9}$$

Define w_x, w_y as the width of 98% mass of projections P_x and P_y respectively. The area of the bounding box is $w_x w_y$ and the quality of image q_a is $1 - w_x w_y$.

Cluttered background tend to produce a large bounding box whereas well defined photos have a smaller bounding box.

A.2.3 Colourfulness

Colourfulness becomes a very important criteria for determining whether a frame is interesting or not. The RGB colourspace is divided into 64 cubic blocks of equal partition. A hypothetical colourful image is generated by sampling points from each of the cubic

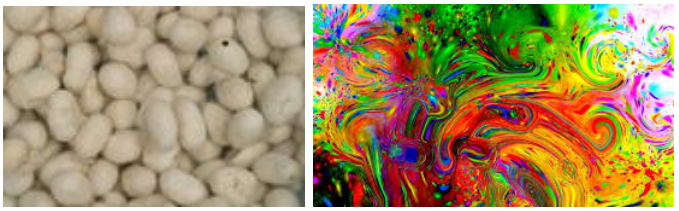

FIGURE A.2: The image to the right is colourful whereas the image on the left is colourless

block with a equal probability of $\frac{1}{64}$. This distribution is named as D_1. The RGB histogram is computed for the input image and is called D_2. The colourfulness measure is the *Earth mover's distance* between the two distributions. The similarity between the color distribution of an arbitrary image is an input image is a rough measure of how colorful that image is.

A.3 Faces and persons

The most popular face and person detection algorithm was proposed by Paul Viola and Micheal Jones, popularly known as *Viola-Jones algorithm*[10]. This algorithm is implemented in Computer vision toolbox of MATLAB®. The algorithm has 4 stages:

- Haar Feature Selection
- Creating an Integral Image
- Adaboost Training
- Cascading Classifiers

This algorithm is quite popular due to its efficient feature selection which are scale and location invariant and, generic detection scheme. This algorithm runs quite fast and is widely available across different programming languages. Figure A.3 shows the faces detected by this algoritm in the image and figure A.4 shows the persons detected by this algorithm and their corresponding score of detection.

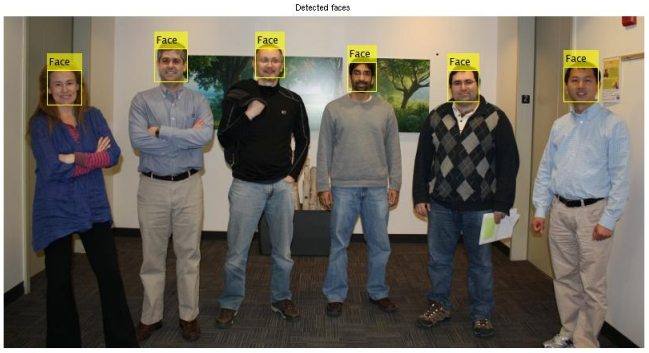

FIGURE A.3: Viola-Jones detector detecting faces in an image

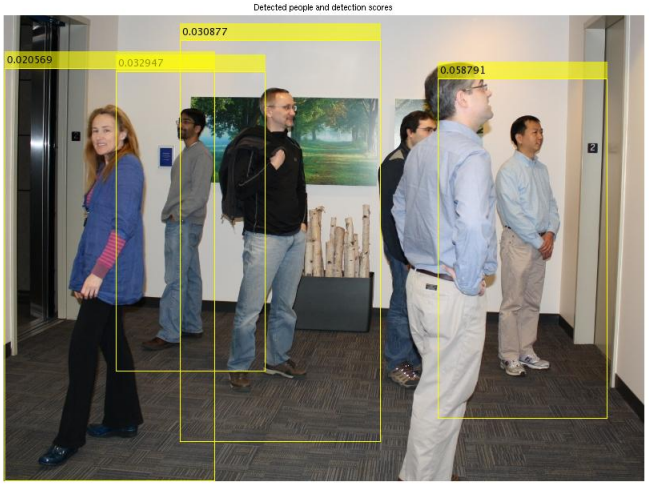

FIGURE A.4: Viola-Jones detector detecting people in an image and the corresponding detection score

Appendix B

Caffe package and AlexNet

B.1 Caffe package

Caffe [12] is a deep learning framework created using C++. It is released under the BSD 2-Clause license. The main motivation benind this framework is expression, speed, and modularity. Berkeley Vision and Learning Center (BVLC) and community contributors have created it. Yangqing Jia created this project during his PhD at UC Berkeley. There are a numerous advantages for prefering this framework over the other.

- *Expressive architecture* is great encouragement for application and innovation. There are no hard-coding involved during configuration of models and optimization. Switching between CPU and GPU is achieved by setting a single flag to train on a GPU machine.
- *Extensible code* aids in active development. Caffe has been forked by a lot of developers and have contributed many significant changes. This framework is state-of-art in both code and model due to these contributors.
- *Speed* makes Caffe perfect for research experiments as well as industry deployment. Caffe can process over 60M images per day with a single NVIDIA K40 GPU*. That is equivalent to 1 ms/image for inference and 4 ms/image for learning. Caffe is the fastest convnet implementation available.

- *Community* Caffe is deployed widely in startup prototypes, academic research projects and even large-scale industrial applications of vision, speech, and multimedia.

B.2 AlexNet

A deep neural network[6] was designed by Alex Krizhevsky et al to classify 1.2 million high-resolution images for the ImageNet LSVRC-2010 contest into 1000 classes. This particular network achieved a top-1 error rates of 37.5% and top-5 error rates of 17.0% which were the least error rates at that point of time. This neural network has 60 million parameters, 650,000 neurons, 5 convolutional layers, some of which are followed by max-pooling layers, and 3 fully-connected layers with a final 1000-way softmax. The AlexNet implementation in Caffe was used. The overall architecture is shown in the figure B.1.

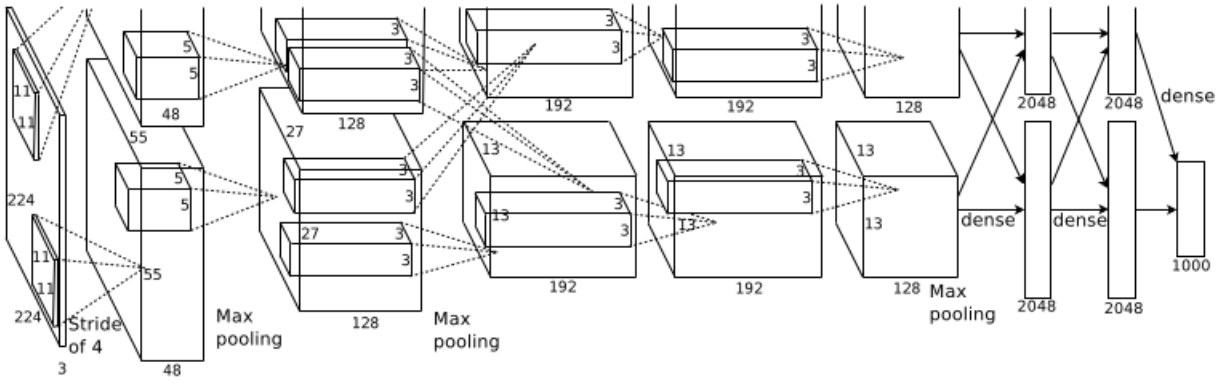

FIGURE B.1: Architecture of AlexNet

Overall architecture

- The network consists of eight layers with weights; the first five layers are convolutional and the rest of three are fully-associated.
- The output from the last fully-associated layer is sent to a "1000-way softmax layer" which produces a prediction over 1000 class labels.
- The kernels of the 2^{nd}, 4^{th}, and 5^{th} convolutional layers are connected only to those kernel maps in the previous layer which reside on the same GPU.

- The kernels of the 3^{rd} convolutional layer are connected to all kernel maps in the second layer.
- The neurons in the fully-connected layers are connected to all neurons in the previous layer.
- The ReLU non-linearity is applied to the output of every convolutional and fully-connected layer.

Appendix C

Optimisation and video segmentation techniques

C.1 Superframe segmentation

Superframe segmentation [3] that has its inspiration from the video editing theory. The video is cut in such a way that the create visualy aesthetic summaries when combined. Superframes are video segments which have their start and end aligned with positions in video which are suitable for a cut. An edited video is assumed to be made up of many small sub-shots. Knowledge from editing theory suggest a cut where there is no motion or there is a match in the motion speed and direction of two neighbouring shots measured. This idea is an extension of *superpixels* for image segmentation. To measure the quality of superframe an *energy function* $E(S_j)$ is defined as

$$E(S_j) = \frac{P_l(\|S_j\|)}{1 + \gamma * C_{cut}(S_j)} \tag{C.1}$$

where C_{cut} is the cut cost and P_l is a length prior for superframes. $\|\cdot\|$ denotes the length of the a superframe. The influence between cut cost and the length prior is controlled by γ, with lower γ leads to more uniform superframes. The cut cost is defined by

$$C_{cut}(S_j) = m_{in}(S_j) + m_{out}(S_j), \tag{C.2}$$

where $m_{in}(S_j)$ and $m_{out}(S_j)$ are motion magnitude estimated in the first and last frame. Tracking points are estimated by KLT which is used for computing mean magnitude of translation to calculate $m_{in}(S_j)$ and $m_{out}(S_j)$. For superframes that have their boundaries aligned with frames containing less or no moiton, the cost is very less. The length prior P_l can be obtained by fitting a log-normal distribution to a histogram of segment lengths. P_l becomes the regularisation term. The energy of C.1 is optimised by hill-climbing optimisation. Inititally superframes are taken to uniformly distributed over the video with segment length $\|S_j\| = \arg\max_l P_l$. The boundaries between the superframes are iteratively updated to optimise C.1. This leads to the segments having their boundaries aligned to position suitable for a cut. The optimisation is performance from coarse to fine way, where a boundary movement by δ is proposed. A movement is accepted if and only if it increases the mean score of C.1 of the two corresponding superframes. The algorithm starts with an intial δ and it is updated iteratively as the algorithm progresses till algorithm converges.

C.2 Knapsack optimisation

Supose given a set of items, each with a weight and a value, it is required to determine which item is to be included in the collection (knapsack) so that the total weight is less than or equal to a given limit and the total value is maximised. This problem derives its name from the dilemma faced by someone who is constrained by a fixed-size knapsack and must fill it with the most valuable items. One can come across this type of problem in any resource allocation situation. Putting it mathematically let us say we have N items one in quantity and each item has a weight w_i and a value v_i with $i = 1, 2, ..., N$. The decision variables are x_i with $x_i = 0$ indicating that the item i is not included and $x_i = 1$ indicating that the item i is included in the knapsack. The capacity that the knapsack can hold is constrained by W.

Maximise

$$z = \sum_{i=1}^{N} x_i * v_i \tag{C.3}$$

subject to

$$\sum_{i=1}^{N} x_i * w_i \leq W \tag{C.4}$$

This is a typical integer linear programming and can be solved using techniques in integer linear programming or by dynamic programming.

C.3 SFO lazy greedy algorithm

Definition A set function $f : \wp(V) \rightarrow \mathbb{R}$ is said to be submodular if given sets $T \subseteq U \subseteq V \setminus \{s\}$ where $s \in V$ then $f(T \cup \{s\}) - f(T) \geq f(U \cup \{s\}) - f(U)$. In other words submodular funtions exhibit the property of diminishing returns of marginal utility.

Definition Let $S \subseteq V$ and $e \in V$, *discrete derivative* for f is defined as, $Delta_f(e|S) := f(S \cup \{e\})$. In other words $\Delta_f(e|S)$ is the *discrete derivative* of f at S with respect to e.

Example Submodular functions are quite popularly used in the applications of computer vision. One real-life example of submodular functions is the problem of placing temperature sensors in a building to form a network of sensors. These sensors detect temperatures and report any abnormal event. The sensors have a limited range of detection and there is a fixed budget for purchsing them. They must be installed in such a way that the total cost is within the budget but they can detect any abnormal activity happening at any corner of the building. Suppose V be the finite set of locations in the building where they can be installed and $f : \wp(V) \rightarrow \mathbb{R}$ is set function quantifying their utility. Now crowding all the sensors near at a particular place decreases the utility because they cannot detect temperatures out of their range. Hence they must be distributed so that their coverage area have minimum overlap with each other but total range of the network should cover the building. When the network's total range covers the full building, adding more sensors is just waste of money. This demonstrates the principle of diminishing returns of marginal utility.

C.3.1 Maximisation

Suppose given a set function which is submodular $f : \wp(V) \rightarrow \mathbb{R}$ and we wish to solve $\max_{S \subseteq V} f(S)$ subject to some constraints on S. Current methods that give exact solutions to this type of problem take exponential time to solve. So there is need for

approximate solutions to this problem. One such method is a *greedy-lazy*[15] method of maximising.

The algorithm starts with the emptyset S_0, and in iteration i, adds the element maximizing the discrete derivative $\Delta(e|S_{i-1})$ with ties broken arbitrarily.

$$S_i = S_{i-1} \cup \{\max_e \Delta(e|S)\} \tag{C.5}$$

This algorithm provides a fairly good approximation for maximisation. The greedy algorithm can be sped by lazy evaluations proposed by Minoux. This is because in some applications evaluating the function f itself can be very costly. In each iteration the algorithm identifies the element e with the maximum marginal gain $\Delta(e|S_{i-1})$, where S_{i-1} is the set of elements selected in the previous iterations. The main idea here is that, as a result of submodularity property of f, the marginal benefits of any fixed element $e \in V$ are monotonically nonincreasing during the iterations of the algorithm. In other words, $\Delta(e|S_i) \geq \Delta(e|S_j)$, whenever $i \leq j$. This algorithm keeps a list of upper bounds $\rho(e)$ (initially set to ∞) on the marginal gains that is sorted in descending order. This is done instead of recomputing $\Delta(e|S_{i1})$ for each element $e \in V$ which requires $O(n)$ computations of f. In each iteration the algorithm extracts the maximum and updates its bounds $\rho(e) \leftarrow \Delta(e|S_{i1})$. Suppose after this update $\rho(e) \geq \rho(e'), \forall e' \neq e$, then submodularity property guarantees that $\Delta(e|S_{i1}) \geq \Delta(e'|S_{i1}), \forall e' \neq e$ and hence the algorithm has found the element of largest gain. We need not have to $\Delta(e'|S_{i1})$ for a very large number of e'. The algorithm then adds $S_i \leftarrow S_{i-1}\{e\}$ and continues till no further element can be added.

Bibliography

[1] Tommy Chheng. Video summarization using clustering. *Department of Computer Science, University of California, Irvine*, pages 1–7.

[2] Yelana Yesha Padmavathi Mundur, Yong Rao. Keyframe-based video summarization using delaunay clustering. *International Journal on Digital Libraries*, 6(2): 219–232, April 2006.

[3] Michael Gygli, Helmut Grabner, Hayko Riemenschneider, and Luc Van Gool. Creating summaries from user videos. *ECCV*, 2014.

[4] Luc Van Gool Michael Gygli, Helmut Grabner. Video summarization by learning submodular mixtures of objectives. *Computer Vision and Pattern recognition*, pages 3090–3098, 2015.

[5] Harry Agius Arthur G. Money. Video summarisation: A conceptual framework and survey of the state of the art. *Elseivier*, pages 121–144, April 2007. URL www.sciencedirect.com.

[6] Geoffrey E. Hinton Alex Krizhevsky, Ilya Sutskever. Imagenet classification with deep convolutional neural networks. *NIPS*.

[7] Mehmood I. Wook Baik S Ejaz, N. Efficient visual attention based framework for extracting key frames from videos. *Signal Processing: Image Communication*, 2013.

[8] Jia Li James Z. Wang Ritendra Datta, Dhiraj Joshi. Studying aesthetics in photographic images using a computational approach. *ECCV*, 2006.

[9] Feng Jing Yan Ke, Xiaoou Tang. The design of high-level features for photo quality assessment. *Computer Vision and Pattern recognition*, 2006.

[10] Jones M Viola P. Robust real-time face detection. *IJCV*, 2004.

[11] H. Lin and J. Bilmes. Learning mixtures of submodular shells with application to document summarization. *In Uncertainty in Artificial Intelligence (UAI)*, 2012.

[12] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[13] Oriol Vinyals Judy Hoffman Ning Zhang Eric Tzeng Trevor Darrell Jeff Donahue, Yangqing Jia. Decaf: A deep convolutional activation feature for generic visual recognition. *arXiv*, pages 1–10, october 2013.

[14] Andreas Krause. Sfo: A toolbox for submodular function optimization. *Journal of Machine Learning Research*, 2010.

[15] Daniel Golovin Andreas Krause. A survey on submodular function maximization. 2012.