LOW COMPLEXITY ON-LINE VIDEO SUMMARIZATION WITH GAUSSIAN MIXTURE MODEL BASED CLUSTERING

Shun-Hsing Ou[1,4], *Chia-Han Lee*[2,4], *V. Srinivasa Somayazulu*[3,4], *Yen-Kuang Chen*[3,4], *Shao-Yi Chien*[1,4]

[1] Media IC and System Lab
Graduate Institute of Electronic Engineering and Department of Electrical Engineering
National Taiwan University, Taipei, Taiwan
[2] Research Center for Information Technology Innovation
Academia Sinica, Taipei, Taiwan
[3] Intel Corporation, USA
[4] Intel-NTU Connected Context Computing Center, Taipei, Taiwan

ABSTRACT

Techniques of video summarization have attracted significant research interests in the past decade due to the rapid progress in video recording, computation, and communication technologies. However, most of the existing methods analyze the video in an off-line manner, which greatly reduces the flexibility of the system. On-line summarization, which can progressively process video during video recording, is then proposed for a wide range of applications. In this paper, an on-line summarization method using Gaussian mixture model is proposed. As shown in the experiments, the proposed method outperforms other on-line methods in both summarization quality and computational efficiency. It can generate summarization with a shorter latency and much lower computation resource requirements.

Index Terms— Video Summarization, Video skimming, On-line video summarization, Gaussian mixture model

1. INTRODUCTION

With the rapid progress of computation, communication, and video recording technologies, today a huge amount of video data are generated every second. How to manage such huge amount of video data has become an important task. To cope with the video data, many researches have focused on exploring efficient solutions for video management.

Techniques of video summarization, or video abstraction, which aim to generate a short representation of the original video for quick indexing and browsing, have thus attracted significant research interests. With video summarization, redundant data in a video stream can be removed while keeping the informative parts. A good summarization can save, for example, the precious time of human operators, the storage space, and the data transmission resource, in many different applications.

However, most of the existing summarization approaches process the video in an off-line manner, which means that the analysis can only start after the entire video is recorded. This limitation reduces the flexibility of the summarization systems. In many applications, such as summarization of data acquired from a video sensor network, it is impractical and inefficient to encode and save the entire video before performing summarization due to the limited storage, computational power, and memory resources. On-line summarization methods [1, 2, 3, 4], in which the summarization is progressively generated during video recoding, are then proposed for a wide range of applications.

On-line summarization means only partial information is available during processing, making it difficult to generate good results. In addition to the summarization quality, a good on-line method should also have the following properties. First, since the on-line methods are usually applied to real-time applications or operated on end devices with limited resources, a good algorithm should be able to operate under low computational complexity and limited memory resources. Second, the method should generate summarization with a very short latency due to the requirements of real-time applications.

In this paper, we present our summarization method using low-complexity on-line Gaussian mixture model (GMM)-based clustering. As shown in the experiments in Section 4, the proposed method generates good summarization results with low computation and memory resources compared to the previous methods [2, 3].

The rest of the paper is arranged as follows. In Section 2, the previous works on video summarization are reviewed. The proposed method is described in Section 3. In Section 4, experimental results are shown. Finally, the conclusion of the paper is given in Section 5.

2. RELATED WORKS

Video summarization has attracted a lot of research attention in the past decade. Comprehensive reviews can be found in [5, 6, 7]. In the TRECVid 2007 and 2008 workshops, video summarization data tracks were provided, and tens of summarization techniques were evaluated and compared [8, 9].

Methods of video summarization can be roughly classified into two categories: keyframe extraction [7, 10, 11, 12, 13, 14, 15] and video skimming [2, 14, 16]. Keyframe extraction selects the most informative frames, while video skimming generates a short video highlight of the original video. Although keyframes can provide a very compact representation of the original video, the timing and motion information are lost. In addition, the summarization results are less enjoyable to watch. On the other hand, video skimming can carry more information, providing more enjoyable video for watching. In this paper, we focus on the video skimming-based method.

Many of the summarization methods apply clustering or graph models with low-level features to group similar contents together. In [12], singular value decomposition is applied to find the most representative frames. Fuzzy-C means clustering is applied in [17]. In [18, 19], graph-based methods are used. Those methods inspire us to employ on-line clustering techniques as a solution to remove redundant frames. Multi-view summarization also attracts many interests recently [10, 13, 16].

Although the previous methods successfully summarize complex videos, most of them operate in an off-line manner. Very limited works focus on the problem of on-line video summarization [1, 2, 3, 4] due to the difficulty of generating summarization with only partial information. Among these methods, only [2] and [3] generate video skimming.

In [3], video summarization is performed directly in compressed video bit stream in an on-line manner to reduce the time and memory requirements for decoding. Feature difference between two continuous frames is used to detect a shot boundary. For each shot, frames are picked at a fix rate. The picked frames are then compared with previously selected summarization frames to reduce redundancy. However, there are two drawbacks of this method. First, the on-line summarization of a single shot can only be done after the entire shot is recorded, which increases the latency and the memory requirements for buffering. Second, each segment has to be compared with all the previously selected frames, which means all the previously selected frames need to be buffered. These drawbacks reduce the advantages of on-line summarization since a large amount of memory is required and the result is generated with an arbitrarily long latency.

In [2], a decision tree is applied for the summarization problem. The input video is divided into constant-length shots first. A binary decision tree is then built-up, where each node represents selecting the shot into summarization or not and each branch represents a result of summarization. The branch with the highest score is selected and the tree is pruned on-line, which acts as a time sliding window. This method still requires large memory buffers and the summarization is also generated with long latency.

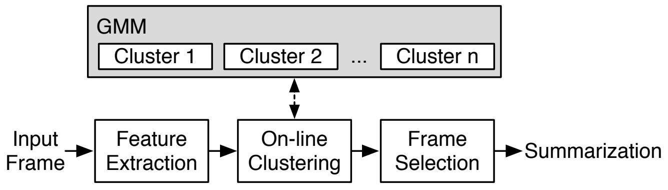

Fig. 1. Block diagram of the proposed method.

Compared with the existing on-line video skimming works[2, 3], our proposed method does not depend on shot detection or temporal sliding window, which means our system can instantly select video skimming frames with minimum latency and buffering. Moreover, the computational cost is low and the memory requirement is light.

3. PROPOSED METHOD

The overview of the proposed method is shown in Fig. 1. As discussed in Section 2, clustering is a common technique used in video summarization to eliminate redundant information. In the proposed system, a feature is extracted first for each input frame. The extracted feature is then clustered by an on-line clustering technique to group similar contents. The importance of the frame is then decided by the clustering result and the parameters of the cluster it belongs to.

There are several choices of on-line clustering. GMM is found to be a suitable solution for two reasons. First, the variance of each cluster can act as a dynamic threshold, providing a better model to the natural video signals. Second, a very efficient on-line parameter estimation method can be applied to perform clustering.

3.1. Feature Extraction

There is no limitation on the features used in the process: any types of features that can successfully represent a frame can be applied. In our implementation, the MPEG-7 color layout descriptor [20] is applied due to its good representation ability and low computational cost for extraction and comparing.

3.2. On-Line Clustering with Gaussian Mixture Model

GMM has been widely applied to many computer vision tasks. One of the most successful applications is background subtraction [21], in which a low-complexity on-line Gaussian mixture model is applied to each pixel for detecting foreground objects. We adopt this idea in our summarization system.

The input video in the feature space is treated a high dimensional "feature process." At time t, GMM regards each feature $\mathbf{x}_t$ as a combination of K Gaussian components, i.e.,

$$\mathbf{x}_t = \sum_{k=1}^{K} \omega_k \eta(\mathbf{x}_t, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{1}$$

where ω_k is the weight, and $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are respectively the mean and covariance matrix of the kth component. η is the probability density function (pdf) of multivariate Gaussian distribution, i.e.,

$$\eta(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^n |\boldsymbol{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}, \tag{2}$$

where n is the dimension of the input feature.

The covariance matrix $\boldsymbol{\Sigma}$ is an $n \times n$ matrix, which requires a large amount of training data to estimate. To simplify the learning process, we assume

$$\boldsymbol{\Sigma} = \sigma^2 \mathbf{I}. \tag{3}$$

This assumes that each dimension of the feature has identical variance. Although this is not true in practice, it is used as an approximation in the high dimensional space and the results are quite satisfactory.

To estimate the parameters ω_k, $\boldsymbol{\mu}_k$, and $\boldsymbol{\Sigma}_k$, EM algorithm is often applied off-line. However, this requires all the features to be buffered, and the computational complexity is usually high due to many iterations before converging to the final result. The on-line parameter estimation method used in [21] is thus applied in our system.

For each input feature $\mathbf{x}_t$ at time t, the distance between the feature $\mathbf{x}_t$ and the mean $\boldsymbol{\mu}_k$ of each component is computed to find a matched component. A component is matched if the distance between $\mathbf{x}_t$ and $\boldsymbol{\mu}_k$ is less than three times of the variance σ_k, i.e.,

$$(\mathbf{x}_t - \boldsymbol{\mu}_k)^T (\mathbf{x}_t - \boldsymbol{\mu}_k) < 3\sigma_k. \tag{4}$$

The weights are then updated as

$$\omega_{k,t} = (1-\alpha)\omega_{k,t-1} + \alpha M_{k,t}, \tag{5}$$

where α is the predefined learning rate, $M_{k,t}$ is 1 for matched component and 0 otherwise. The mean and covariance of the matched component are also updated as

$$\boldsymbol{\mu}_{k,t} = (1-\rho)\boldsymbol{\mu}_{k,t-1} + \rho \mathbf{x}_t, \tag{6}$$

$$\sigma_{k,t}^2 = (1-\rho)\sigma_{k,t-1}^2 + \rho(\mathbf{x}_t - \boldsymbol{\mu}_{k,t})^T(\mathbf{x}_t - \boldsymbol{\mu}_{k,t}), \tag{7}$$

where ρ is the second learning rate and

$$\rho = \alpha \eta(\mathbf{x}_t, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k). \tag{8}$$

If no matched components are found, the component with the smallest probability is updated as $\boldsymbol{\mu}_k = \mathbf{x}_t$ with a small weight and high variance.

Table 1. Summarization Results. Note that due to the limitation of space, only the average results of each dataset, instead of the results of each video, are shown.

Dataset	Method	Precision	Recall	F1 score
BL-7F	[2], D=30	15.4%	63.1%	0.25
(19 videos)	[2], D=90	21.9%	77.2%	0.34
	[3]	30.4%	44.4%	0.36
	GMM	62.6%	74.4%	0.68
Office	[2], D=30	15.3%	77.5%	0.26
(4 videos)	[2], D=90	17.8%	79.8%	0.29
	[3]	15.5%	49.3%	0.23
	GMM	44.4%	88.0%	0.59
Lobby	[2], D=30	77.0%	52.3%	0.62
(3 videos)	[2], D=90	79.9%	42.6%	0.56
	[3]	48.0%	50.3%	0.49
	GMM	72.0%	90.5%	0.80

3.3. Frame Selection

After the clustering, the decision of keeping or dropping the current frame is determined by the parameters ω and σ of the cluster it belongs to. A cluster with larger weight indicates that its content appears more frequently, and thus the content of the cluster is usually more redundant and less informative. A cluster with lower variance indicates a lower activity level in that cluster, and thus it is also less important. Therefore, frames belong to a cluster with large weight and small variance should be filtered out.

The frame selection is performed as follows. At each time t, all clusters are sorted in descending order by the value of $\frac{\omega_{i^*}}{\sigma_{i^*}}$, $1 \leq i^* \leq K$. The clusters in the front of the sorted list are usually less important. For the current frame belonging to the k^*th cluster in the sorted list, we check if

$$\sum_{i^*=1}^{k^*} \omega_{i^*} < T, \tag{9}$$

where i^* is the sorted index and T is the predefined summarization rate. If this inequality holds, the frame is considered non-informative and is then dropped; otherwise the frame is added to the final summarization.

Although the frame selection process is performed in the unit of a single frame, the selected frames usually appear continuously. This is because the selection is based on on-line updating the parameters, which evolves smoothly. The summarization result is thus a video skimming video instead of keyframes.

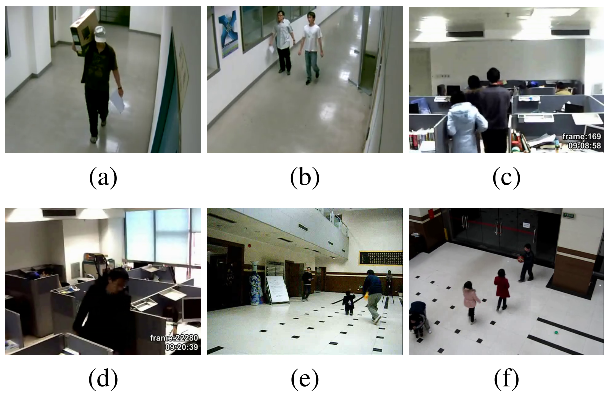

Fig. 2. Example frames of the datasets used in the experiments. (a)(b): *BL-7F*, (c)(d): *Office*, and (e)(f): *Lobby*.

4. EXPERIMENTAL RESULTS

4.1. Dataset and Benchmark

In order to evaluate the proposed summarization method, experiments are conducted using 3 different datasets with 27 videos in total. The description of the dataset is as follows:

- BL-7F: This dataset contains 19 videos in total taken by our surveillance system installed in the 7th floor of the BarryLam Building in National Taiwan University.
- Office: This is the *office1* dataset provided in [16]. There are 4 videos taken with stably-held but non-fixed cameras in an office room. The vibration of the cameras makes it difficult to generate good summarization.
- Lobby: This is the *office lobby* dataset provided in [16]. There are 3 videos taken with stably-held but non-fixed cameras in a large lobby area.

Example frames of those datasets are shown in Fig. 2. Two on-line video skimming methods [2, 3] are implemented as the benchmark methods. Both of these two video skimming methods have been proven to be able to generate comparable results with other off-line methods in the summarization competition [8, 9].

In the experiments, $K = 9$ and $\alpha = 0.004$ is selected for our system. Since in [2] the depth of the tree, D, greatly affects the summarization result and the memory usage, we use both $D = 30$ and $D = 90$ in our experiments.

4.2. Summarization Results

In order to provide an objective comparison, a quantitative metric is applied to evaluate the summarization results. We hire human operators who are left unknown to our work to mark the important segments for each video. The evaluation is then computed by comparing the overlap between the summarization result and the marked ground truth. The precision and recall are calculated in the frame level, where the precision indicates the ability to remove useless parts and the recall indicates the ability to keep information.

As shown in Table 1, the proposed GMM method outperforms the other two methods in both precision and recall. Note that due to the limitation of space, only the average results of videos in each dataset is shown. The F1 score is also computed for comparison.

Table 2. Computation Efficiency of Each Method.

	[2] D=30	[2] D=90	[3]	GMM
FPS (f/s)	21.8	18.8	9.3	34.7
Latency (s)	30	90	$\sim$200	$\sim$0
# Buffered Frames	900	2700	$\sim$5000	1

4.3. Computational Efficiency

We also evaluate the computational efficiency of the on-line video summarization. In our experiments, a low-power Intel ATOM$^{\text{TM}}$ N570 processor with 2GB memory is used for simulating the low power devices. The *Office* dataset with the 640×480 resolution is applied. All the three methods are implemented in C++.

In [2], each tree node has 30 frames. Therefore, the number of frames that need to be buffered is $D \times 30$. In [3], both the frames in the current shot and the quantized DC frames of the previously selected summary are buffered. The latency is determined by the length of the shot.

As shown in Table 2, the proposed method requires much lower computation resources while generating video summary almost without any latency.

5. CONCLUSION

In this paper, we have proposed an on-line video summarization method using the Gaussian mixture model. The experimental results show that the proposed scheme outperforms the previous methods. With much lower computation resource requirements compared to the previous on-line video skimming methods, the proposed GMM-based on-line video summarization can thus be applied to a wide range of applications.

Acknowledgement

This work was supported by National Science Council, National Taiwan University and Intel Corporation under Grants, NSC102-2911-I-002-001 and NTU-ICRP-103R7501.

6. REFERENCES

[1] W. Abd-Almageed, "Online, simultaneous shot boundary detection and key frame extraction for sports videos using rank tracing," in *Proc. IEEE Inc. Conf. Image Process. (ICIP)*, Oct. 2008, pp. 3200–3203.

[2] V. Valdés and J. M. Martínez, "Binary tree based on-line video summarization," in *Proc. ACM TRECVid Video Summarization Workshop*, Oct. 2008, pp. 134–138.

[3] J. Almeida, N. J. Leite, and R. da S. Torres, "Online video summarization on compressed domain," *J. of Visual Commun. and Image Representation*, vol. 24, no. 6, pp. 729 – 738, Aug. 2013.

[4] X. Zeng, X. Xie, and K. Wang, "Instant video summarization during shooting with mobile phone," in *Proc. ACM Int. Conf. Multimedia Retrieval (ICMR)*, April 2011, pp. 40:1–40:8.

[5] B. T. Truong and S. Venkatesh, "Video abstraction: A systematic review and classification," *ACM Trans. Multimedia Comput. Commun. Applicat.*, vol. 3, no. 1, Feb. 2007.

[6] A. G. Money and H. Agius, "Video summarisation: A conceptual framework and survey of the state of the art," *J. Vis. Commun. Image Representation*, vol. 19, no. 2, pp. 121–143, Feb. 2008.

[7] C. Sujatha and U. Mudenagudi, "A study on keyframe extraction methods for video summary," in *Proc. Int. Conf. Computational Intell. and Commun. Networks*, Oct. 2011, pp. 73–77.

[8] P. Over, A. F. Smeaton, and P. Kelly, "The TRECVid 2007 BBC rushes summarization evaluation pilot," in *Proc. Int. Workshop on TRECVid Video Summarization*, Sep. 2007, pp. 1–15.

[9] P. Over, A. F. Smeaton, and G. Awad, "The TRECVid 2008 BBC rushes summarization evaluation," in *Proc. ACM TRECVid Video Summarization Workshop*, Oct. 2008, pp. 1–20.

[10] Y. Li and B. Merialdo, "Multi-video summarization based on Video-MMR," in *Proc. Int. Workshop on Image Anal. for Multimedia Interactive Services*, April 2010, pp. 1–4.

[11] A. Khosla, R. Hamid, C.-J. Lin, and N. Sundaresan, "Large-scale video summarization using web-image priors," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR)*, June 2013, pp. 2698–2705.

[12] Y. Gong and X. Liu, "Video summarization using singular value decomposition," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR)*, vol. 2, June 2000, pp. 174–180.

[13] P. Li, Y. Guo, and H. Sun, "Multi-keyframe abstraction from videos," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2011, pp. 2473–2476.

[14] Y.-F. Ma, X.-S. Hua, L. Lu, and H.-J. Zhang, "A generic framework of user attention model and its application in video summarization," *IEEE Trans. Multimedia*, vol. 7, no. 5, pp. 907–919, Oct. 2005.

[15] Y. J. Lee, J. Ghosh, and K. Grauman, "Discovering important people and objects for egocentric video summarization," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR)*, June 2012, pp. 1346–1353.

[16] Y. Fu, Y. Guo, Y. Zhu, F. Liu, C. Song, and Z.-H. Zhou, "Multi-view video summarization," *IEEE Transactions on Multimedia*, vol. 12, no. 7, pp. 717–729, Nov. 2010.

[17] E. Asadi and N. Charkari, "Video summarization using fuzzy c-means clustering," in *Proc. Iranian Conf. Elect. Eng.*, May 2012, pp. 690–694.

[18] J. Kwon and K.-M. Lee, "A unified framework for event summarization and rare event detection," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR)*, June 2012, pp. 1266–1273.

[19] C.-W. Ngo, Y.-F. Ma, and H.-J. Zhang, "Video summarization and scene detection by graph modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 2, pp. 296–305, Feb. 2005.

[20] E. Kasutani and A. Yamada, "The MPEG-7 color layout descriptor: a compact image feature description for high-speed image/video segment retrieval," in *Proc. Int. Conf. Image Process. (ICIP)*, vol. 1, Oct. 2001, pp. 674–677.

[21] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recognition (CVPR)*, vol. 2, June 1999, pp. 246–252.