ON SUMMARISING THE 'HERE AND NOW' OF SOCIAL VIDEOS FOR SMART MOBILE BROWSING

Zbigniew Zdziarski[1,2], *Cyril Bourgès*[2], *Joe Mitchell*[1], *Pierre Houdyer*[1], *Dave Johnson*[1] & *Rozenn Dahyot*[2]

[1]Tapastreet Ltd, Ireland [2]Trinity College Dublin, Ireland
{zbigniew, joe, pierre, dave}@tapastreet.com {bourgesc, Rozenn.Dahyot}@tcd.ie

ABSTRACT

The amount of media that is being uploaded to social sites (such as Twitter, Facebook and Instagram) is providing a wealth of visual data (images and videos) augmented with additional information such as keywords, timestamps and GPS coordinates. Tapastreet[1] provides access in real-time to this visual content by harvesting social networks for visual media associated with particular locations, time and hashtags [1]. Browsing efficiently through harvested videos requires smart processing to give users a quick overview of their content in particular when using mobile platforms with limited bandwidth. This paper aims at presenting an architecture for testing several strategies for processing summaries of videos collected on social networks to tackle this issue.

Index Terms— Social Media, Web Harvesting, Video Summarisation, Blur Detection, MPEG Codec

1. INTRODUCTION

Video content understanding for indexing, parsing and browsing has been a very popular topic of research in the past decades to help to deal efficiently with very large video repositories. Historically, a large body of work on video content analysis has been mainly focused on processing media created by broadcasters (e.g. TV). Such repositories have video content that is well structured and categorised (e.g. film, news, sport, advertisements, talk shows, etc.), with images of very good quality and with very well defined context-dependent editing rules. On the other hand, very little has been done for videos uploaded on social networks that are often of lesser quality with no or little editing rules and structure. Indeed, an increasing amount of new media content is uploaded from a wide variety of places transforming all users to independent amateur broadcasters. This is providing a new wealth of information about what is happening over time at particular locations on the planet helping rescue efforts, for instance, when facing natural disasters [2]. This opens up new opportunities for designing new search engines dedicated to social media such as the Tapastreet web app. Tapastreet has a location based social media search engine platform that, in its current form, returns geo-located video and image media from major networks for any location and any topic (#hashtags) anywhere in the world. The current platform deals well with images on social media but videos are yet not well tackled. This paper is addressing the problem of displaying and browsing efficiently (i.e. in a smart, informative and fast fashion) video content that has been harvested from social platforms. More specifically we present an early stage platform for testing video summarisation that is under development by Trinity College Dublin and Tapastreet as part of a European project called GRAISearch[2]. We review briefly past work on video summarisation and deblurring (Section 2). We then present (Section 3) and discuss (Sections 4 & 5) our platform with its future directions for improvement.

2. STATE OF THE ART

Truong and Venkatesh proposed to perform video summarisations (a.k.a. video abstractions) by selecting informative and diverse keyframes from video streams [3]. For example, a video can be processed sequentially to mark key frames as ones that are significantly different from previously extracted key frames [4, 5]. A more computationally demanding method has been proposed by Gibson et al. [6] and Yu et al. [7]. They employ a clustering technique where video frames are treated as points in a feature space (e.g. colour histogram) and representative points from each cluster are selected as key frames of the video. A faster approach consists of computing metrics from information theory (e.g. entropy) for each frame in the video stream for detecting editing (e.g. shot changes: cuts, fades and dissolves) in the video stream and then selecting representative key frames from each segmented shot [8]. An even faster possibility is to avail of scene detection information that is inherently embedded in MPEG compressed data. In the MPEG video compression algorithm, frames are grouped into sequences called group of pictures (GOP) starting and ending with I frames (commencement of

[1]http://tapastreet.com/

[2]This work has been supported by the European project GRAISearch FP7-PEOPLE-2013-IAPP (612334), 2014-2018. http://graisearch.scss.tcd.ie

978-1-4799-7971-4/14/$31.00 ©2014 IEEE

a new scene as detected by MPEG) and containing P and B frames (resp. for predictive coded frames and bi-directional predicted frames) [9]. In a similar strategy as [10, 11], we pick the middle frame of a GOP as a key frame for representing a scene.

Blur detection is an important step in our approach because with amateur videos, sometimes every second or third frame is blurred due to mishandling of the camera and hand-shaking. Recently, Pertuz et al. [12] performed a comprehensive analysis of 36 focus measuring operators that themselves were chosen from an extensive review of the state-of-the-art. The purpose of their study was to determine the best of these operators to then perform Shape-from-focus (SFF), which is a depth recovery and 3D reconstruction method. The algorithms that they analysed can be grouped into six classes: gradient-based operators, Laplacian-based operators, wavelet-based operators, statistics-based operators, DCT-based operators and miscellaneous operators (operators that do not belong in any of the previous classes). Pertuz et al. pointed out that the following algorithms have good performance as well as execution time: (1) the gradient-based Tenengrad operator [13], which uses the variance of an image's gradient as a focus measure; (2) Modified laplacian operator [14] and (3) the sum of wavelet coefficients [15, 16].

3. PLATFORM FOR SOCIAL VIDEO SUMMARISATION

3.1. Harvesting social media

All our video summarisation scripts are currently written in Matlab running on our server and are called by a Ruby program. Only performant Matlab scripts will eventually be optimised and re-written in C/C++ (using e.g. OpenCV) once extensive user assessment is performed. The flowchart for the system is presented in Figure 1. Using social network APIs

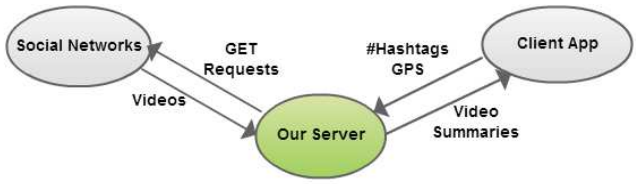

Fig. 1: Harvesting social networks.

(Instagram and Viddy are currently used for now and our system will be extended to other social networks like Twitter and Vine), the Ruby program is used to download all media using a user-defined query (hashtags, GPS location), and links to these images and videos are stored along with their description in JSON format (keywords, GPS location, creation date, etc.) on our server.

3.2. Video processing pipeline

The flowchart of the video summarisation process can be seen in Figure 2. In step 1, we select the middle frame of each GOP

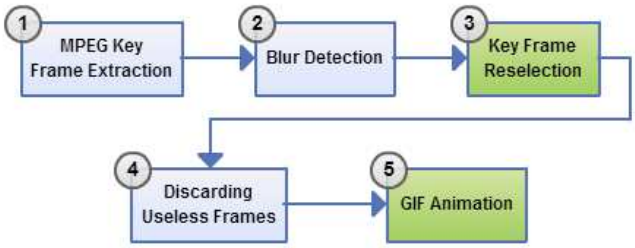

Fig. 2: Video summarisation pipeline.

as the key frame using MPEG encoding of the videos posted online (MPEG-4). In step 2, we process three algorithms (the gradient-based Tenengrad operator [13], modified laplacian operator [14] and sum of wavelet coefficients [15, 16]) on each of the key frames extracted from the MPEG video and also the next $fps/3$ frames for each of these[3]. In step 3, the frame with the lowest aggregate blur result (highest focus result) replaces the key frame selected in step 1. We chose three algorithms from three different groups because if one algorithm failed under the conditions of the video, the other two were there to compensate for it making the system more reliable. In step 4, because social media are in large majority filled with amateur videos with no post-processing or editing, single coloured frames can frequently appear. Black coloured frames can occur when the recording takes place at night or indoors. Sometimes the first few frames of a video are completely black as well. Blue frames can occur, for example, when the user pans his device across the sky. Since frames such as these are significantly different to normal frames, they get flagged as key frames. A simple discarding algorithm for these useless frames has been implemented. If the aggregated standard deviation of R, G, and B values in a frame is smaller than some threshold (which is close to zero), the frame is immediately discarded [17, 18]. In step 5, the selected key frames are stacked together and stored in a GIF file.

4. PLATFORM TESTING

A screenshot of our web user interface on our server is shown in Figure 3. The user provides the URL of the video to be summarised (top), then selects the script 'PiBlurStdDev' (our current version for processing the pipeline - ultimatly several processing pipelines will be implemented and compared) to be used and clicks 'Process'. The results are accessible by clicking on the second button. The 'format' option specifies what format the final animation should be in (see Section 4.2). Results for one video can be seen in Figure 4. Each line of

[3] fps: # frames per second

Fig. 3: Webpage for processing summarisation on our server: `http://graisearch.scss.tcd.ie/summarisation`

images are the key frames generated after each step of our pipeline: I frames at the top, middle GOP below, and our final selected keyframes at the bottom.

4.1. Exemplar scenarios

We ran two experiments associated with sports events. The first was the *Commonwealth Games*: The opening ceremony of the 2014 Commonwealth Games was held in Glasgow on 23/7/2014 at 9pm GMT at Celtic Park. The Ruby program was set up to download videos from social media sites uploaded at the GPS location of the ceremony. In total, 29 videos were downloaded from the beginning of the ceremony until midnight that day. All these videos were amateur videos. In total 137 key frames were extracted from the MPEG files of which 59 were blurry. This shows the importance of having a good blur detection and key frame reselection process. The average sequence length was 13.1 seconds. The biggest number of key frames extracted from the MPEG file was 31 (from a 15 sec long clip)[4]. This video was extremely shaky and, hence, the MPEG compression algorithm detected a lot scenes despite the video showing only a pan across a carpark. One video had only one key frame extracted[5]. It was 6 secs in length and was of the Jamaican team walking in front of the camera. The video was relatively stable. The second event is the *Tour de France*: Stage 18 of the Tour de France began in Pau, went through Trebons and finished in Hautacam in the south of France. The Ruby program was set up to download videos uploaded at the GPS locations of those towns for the entire day of the leg (from 12pm GMT to 7pm GMT). A total of 18 videos were obtained from Instagram and Viddy during this period. All but two were of an amateur nature. The most number of key frames extracted from the MPEG file was 23 (from a 14 sec long clip)[6]. This video was taken by a person filming cyclists riding past them. Since the cyclists all had colourful jerseys, sometimes every 4-5 frames were marked as key frames. 3 videos had only one key frame extracted. They were all videos of short length (6-8 secs) showing a homogenous scene. The total number of key frames extracted from MPEG files was 109 of which 15 were blurry.

4.2. Animation presented to user

We performed some analysis to discern what the best way will be to present the final chosen key frames to the user. We compared 5 techniques: animated GIF, APNG (a non-standard extension to PNG), Webp, Webp-lossy (both are new image formats from Google) and MPEG-4 on 6 videos from the two scenarios mentioned above. Table 1 shows results of our analysis. Webp-lossy fares the best out of the 5 techniques. A close second is MPEG-4. A major advantage of MPEG-4, however, is that it does not suffer from incompatibility issues that APNG and Webp/Webp-lossy does - they are both not compatible on iPhones and some desktop browsers. Although popular, animated GIFs are not efficient memory-wise and alternative representations will be sought.

5. DISCUSSION AND FUTURE WORKS

We presented here a research platform for performing social video summarisation. This architecture provides the user with a central point of access to media from the largest social media sites. We presented three steps here for video summarisation: key frame extraction directly from MPEG compression data, redundant frame removal and blur detection. Initial results look very promising[7]. Sometimes there are too many GOPs in the MPEG videos. This happens when the scene in the video changes very quickly usually when the user moves the camera around a lot. Videos on social networks are so diverse that it is expected that good summarisation capabilities will only be achieved by deploying several processing pipelines each adapted to the specific types of the harvested videos. As an alternative to stacking key frames in an animated GIF, we will also consider the possibility for non-photo realistic image representation of videos [19], panoramic image representation (using image stiching from images in the video), visual saliency [20] for selecting informative key frames and/or efficiently croping selected keyframes, and 3D representation of the scene captured in videos when suitable.

[4] `http://scontent-a.cdninstagram.com/hphotos-xfa1/t50.2886-16/10567199_1447077282228239_170983820_n.mp4`

[5] `http://scontent-a.cdninstagram.com/hphotos-xaf1/t50.2886-16/10567219_1503795999857100_1728234065_n.mp4`

[6] `http://scontent-a.cdninstagram.com/hphotos-xfa1/t50.2886-16/10570745_645836292191332_326672798_n.mp4`

[7] More results available at `http://graisearch.scss.tcd.ie/summarisation`.

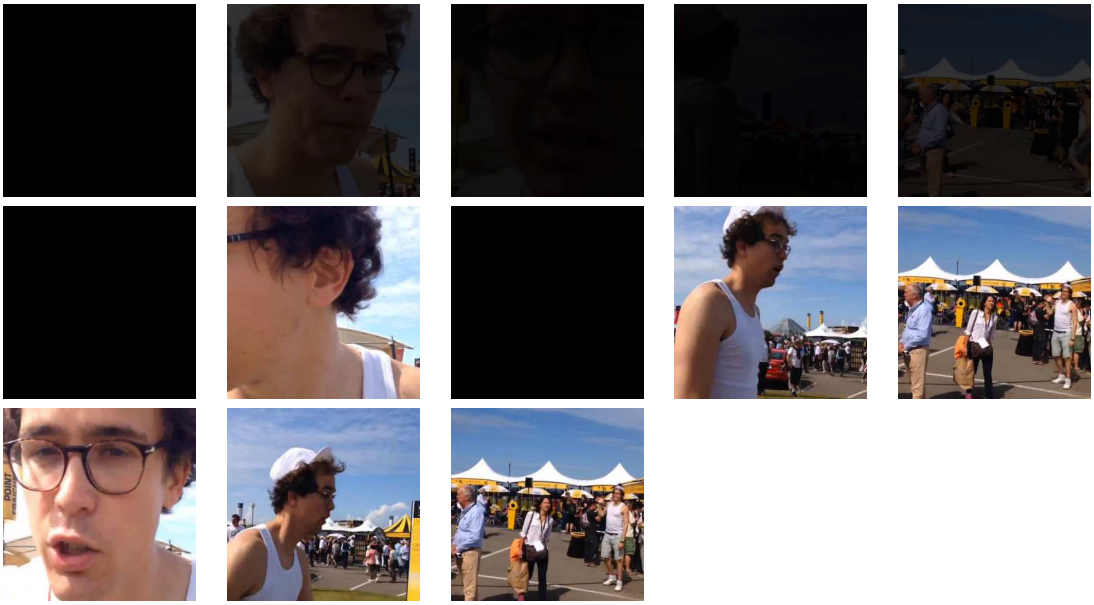

Fig. 4: Results for a video in the Tour de France scenario. First row: I frames from the MPEG codec; Second row: middle GOP frames; Third row: our final selected key frames. Results page (with original video) available here: `http://graisearch.scss.tcd.ie/summarisation/result?url=http://scontent-a.cdninstagram.com/hphotos-xaf1/t50.2886-16/10570755_1513514468861513_773365679_n.mp4`

Video	Orig. size	# of Key Frames	GIF	APNG	Webp	Webp-lossy	MP4
Games1	3.50	2	0.53	1.01	0.64	**0.11**	0.12
Games2	2.25	8	1.80	1.46	1.35	0.29	**0.25**
Games3	2.26	6	1.76	2.62	1.82	**0.21**	0.40
TourDF1	5.56	5	1.31	1.91	1.22	**0.15**	0.27
TourDF2	4.30	7	1.92	3.19	2.21	**0.30**	0.50
TourDF3	2.73	3	0.88	1.31	0.86	**0.09**	0.14

Table 1: Animation analysis results. First column: video source (3 from Commonwealth Games, 3 from Tour de France), second column: original video size, third column: number of key frames generated, remaining columns: size of files for animated GIF, APNG, Webp, Webp-lossy, and MP4 respectively. All file sizes are in MB.

6. REFERENCES

[1] Z. Zdziarski, J. Mitchell, P. Houdyer, D. Johnson, C. Bourges, and R. Dahyot, "An architecture for social media summarisation," in *Irish Machine Vision and Image Processing Conference*, Derry-Londonderry, Northern Ireland, 27-29 August 2014.

[2] Virtual Social Media Working Group and DHS First Responders Group, "Lessons learned: Social media and hurricane sandy," June 2013, Homeland Security, Science and Technology.

[3] Ba Tu Truong and Svetha Venkatesh, "Video abstraction: A systematic review and classification," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 3, no. 1, Feb. 2007.

[4] Xu-Dong Zhang, Tie-Yan Liu, Kwok-Tung Lo, and Jian Feng, "Dynamic selection and effective compression of key frames for video abstraction," *Pattern Recognition Letters*, vol. 24, no. 9-10, pp. 1523 – 1532, 2003.

[5] Changick Kim and Jenq-Neng Hwang, "Object-based video abstraction for video surveillance systems," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 12, no. 12, pp. 1128–1138, Dec 2002.

[6] D. Gibson, N. Campbell, and B. Thomas, "Visual abstraction of wildlife footage using gaussian mixture models and the minimum description length criterion," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, 2002, vol. 2, pp. 814–817 vol.2.

[7] Xiao-Dong Yu, Lei Wang, Qi Tian, and Ping Xue, "Multilevel video representation with application to keyframe extraction," in *Multimedia Modelling Conference, 2004. Proceedings. 10th International*, 2004, pp. 117–123.

[8] Z. Cerneková, C. Nikou, and I. Pitas, "Entropy metrics used for video summarization," in *Proceedings of the 18th Spring Conference on Computer Graphics*, New York, NY, USA, 2002, SCCG '02, pp. 73–82, ACM.

[9] Didier Le Gall, "Mpeg: A video compression standard for multimedia applications," *Commun. ACM*, vol. 34, no. 4, pp. 46–58, Apr. 1991.

[10] Guozhu Liu and Junming Zhao, "Key frame extraction from mpeg video stream," in *Information Processing (ISIP), 2010 Third International Symposium on*, Oct 2010, pp. 423–427.

[11] J. Calic and E. Izuierdo, "Efficient key-frame extraction and video analysis," in *Information Technology: Coding and Computing, 2002. Proceedings. International Conference on*, April 2002, pp. 28–33.

[12] Said Pertuz, Domenec Puig, and Miguel Angel Garcia, "Analysis of focus measure operators for shape-from-focus," *Pattern Recognition*, vol. 46, no. 5, pp. 1415 – 1432, 2013.

[13] E. Krotkov and J.-P. Martin, "Range from focus," in *Robotics and Automation. Proceedings. 1986 IEEE International Conference on*, Apr 1986, vol. 3, pp. 1093–1098.

[14] S.K. Nayar and Y. Nakagawa, "Shape from focus," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 16, no. 8, pp. 824–831, Aug 1994.

[15] Ge Yang and B.J. Nelson, "Wavelet-based autofocusing and unsupervised segmentation of microscopic images," in *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, Oct 2003, vol. 3, pp. 2143–2148 vol.3.

[16] Hui Xie, Weibin Rong, and Lining Sun, "Wavelet-based focus measure and 3-d surface reconstruction method for microscopy images," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, Oct 2006, pp. 229–234.

[17] Naveed Ejaz, Tayyab Bin Tariq, and Sung Wook Baik, "Adaptive key frame extraction for video summarization using an aggregation mechanism," *Journal of Visual Communication and Image Representation*, vol. 23, no. 7, pp. 1031 – 1040, 2012.

[18] Marco Furini, Filippo Geraci, Manuela Montangero, and Marco Pellegrini, "Stimo: Still and moving video storyboard for the web scenario," *Multimedia Tools and Applications*, vol. 46, no. 1, pp. 47–69, 2010.

[19] A. Kokaram, F. Pitie, R. Dahyot, N. Rea, and S. Yeterian, "Content controlled image representation for sports streaming," in *IEEE workshop on Content Based Multimedia Indexing (CBMI'05)*, Riga, Latvia, June 2005.

[20] Z. Zdziarski and R. Dahyot, "Feature selection using visual saliency for content-based image retrieval," *IET Irish Signals and Systems Conf.*, 2012.