VideoSum: A Python Library for Surgical Video Summarization

Luis C. Garcia-Peraza-Herrera, Sébastien Ourselin, and Tom Vercauteren

King's College London
luis_c.garcia_peraza_herrera@kcl.ac.uk

INTRODUCTION

The performance of deep learning (DL) algorithms is heavily influenced by the quantity and quality of the annotated data. However, in Surgical Data Science (SDS) [1], access to it is limited. It is thus unsurprising that substantial research efforts are made to develop methods aiming at mitigating the scarcity of annotated SDS data. In parallel, an increasing number of Computer Assisted Interventions (CAI) datasets are being released [1], although the scale of these remains limited. On these premises, data curation is becoming a key element of many SDS research endeavors.

Surgical video datasets are demanding to curate and would benefit from dedicated support tools. With an average duration of 130.45 minutes [2], surgical interventions pose challenges in terms of visualization, annotation and processing. Simple tasks such as discovering whether videos have frames that have not been correctly anonymized, or annotating the type of a surgical procedure quickly turn into tedious tasks.

In this work, we propose to summarize surgical videos into *storyboards* or *collages* of representative frames to ease visualization, annotation, and processing. Video summarization is well-established for natural images [3]. However, state-of-the-art methods typically rely on models trained on human-made annotations [4], few methods have been evaluated on surgical videos and the availability of software packages for the task is limited. We present `videosum`[2], an easy-to-use and open-source Python library to generate *storyboards* from surgical videos that contains a variety of unsupervised methods.

MATERIALS AND METHODS

Although there are many ways one could opt for to summarize a surgical video, we convert each video into a *storyboard*, i.e. a set of representative video frames, as the one shown in Fig. 1. Formally, we aim to solve for a mapping f such that $f_{\boldsymbol{\theta}}(X) \approx \{Y, Z\}$, where f_θ is a parametric function, $\boldsymbol{\theta}$ is a vector of parameters, $X = \{\boldsymbol{x}_i\}_{i=1}^{N_f}$ is a set of ordered video frames $\boldsymbol{x}_i \in \mathbb{R}^{H \times W \times C}$, $N_f \in \mathbb{N}$ is the number of frames in the video X, $Y = \{y_i\}_{i=1}^{N_c}$ is a set containing the indices $\boldsymbol{y}_i \in \{1, ..., N_f\}$ of the key frames, $N_c \in \mathbb{N}$ is the number of clusters to be included in the storyboard,

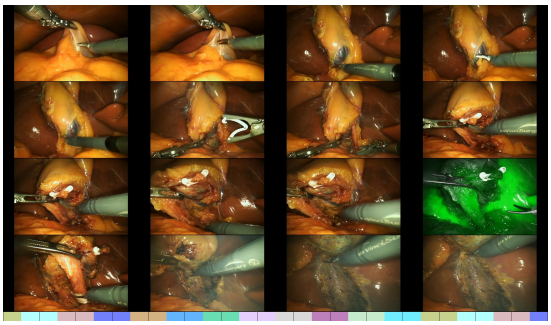

Fig. 1 Baseline storyboard of a surgical video produced by the *time* method of `videosum`. The bar under the collage represents the length of the video. The colors represent the cluster label of the video frames, and the black vertical bars are the key frames. The video is split into uniform temporal segments by the *time* method.

and $Z = \{z_i\}_{i=1}^{N_f}$ is an ordered set of clustering labels $z_i \in \{1, ..., N_c\}$ indicating to which cluster an input frame belongs.

At present, four different methods are available in `videosum`: *time*, *inception*, *uid* [3], and *scda* [5]. The *time* method trivially partitions the video in equal temporal parts and selects the median frame index of each cluster as key frame. In all the other methods, the frames of the video are clustered using k-medoids. The difference between *inception*, *uid*, and *scda* lies on two aspects: 1) how the feature vector is computed for any given frame; and 2) what distance metric is used for clustering. In the *inception* method, the representation of a frame is provided by the 2048-element latent vector from InceptionV3 [6] trained on ImageNet. The distance metric for clustering in this case is the ℓ^2-norm. In the *uid* method, InceptionV3 latent vectors are also used, but the clustering metric is the 2-Wasserstein distance computed between univariate Gaussian models fitted on the elements of the inception feature vector. With μ_i the mean and σ_i the standard deviation of the 2048-element vector computed from frame i, the distance between frame i and j reads $\sqrt{(\mu_i - \mu_j)^2 + (\sigma_i - \sigma_j)^2}$. In the *scda* method, the feature vector of a frame is computed as proposed in [5] (except that we used InceptionV3 instead of VGG-16), and the ℓ_2-norm is used as a distance metric for clustering. To generate the image descriptor, [5] use a low-resolution latent tensor in the

[1] See https://github.com/luiscarlosgph/list-of-surgical-tool-datasets
[2] See https://github.com/luiscarlosgph/videosum
[3] *uid* stands for univariate inception distance.

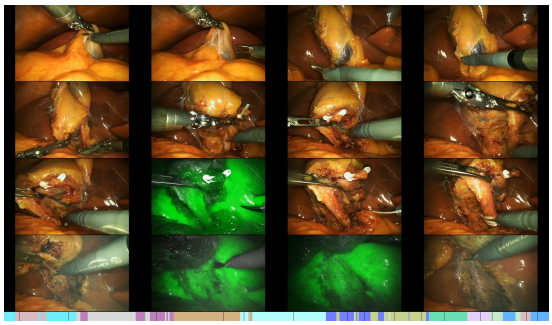

Fig. 2 Storyboard produced by the *inception* method.

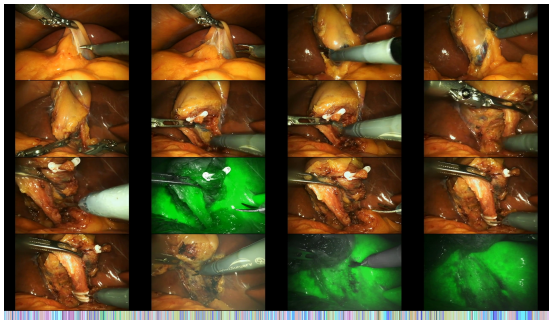

Fig. 3 Storyboard produced by the *uid* method.

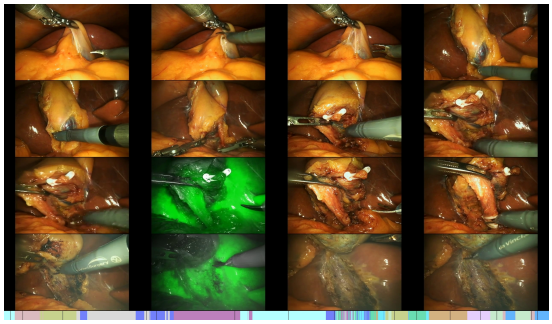

Fig. 4 Storyboard produced by the *scda* method.

network. They sum it over channels and threshold the collapsed tensor to obtain a binary mask. The largest connected component of this binary mask is used to select indices in the low-resolution tensor and create the representation vector for the input image.

One of the risks of selecting visually distinct frames is that storyboards could concentrate on a short part of the surgery. To mitigate it, `videosum` supports time smoothing, allowing the user to select how close the summary should be to the *time* version.

RESULTS

An exemplary storyboard for the methods *inception*, *uid*, and *scda* is shown in Fig. 2, 3, and 4, respectively. These storyboards correspond to the same video[4] whose *time* summary is presented in Fig. 1.

In order to evaluate the different methods, we use the *Fréchet Inception Distance* (FID) [7] to compare how close the collection of images in the *storyboard* is to the full set of frames in the video. In Fig. 5, we show how the FID changes as we increase the number of images in the storyboard for each one of the methods. As shown in Fig. 5, the *inception* method outperforms all the others for collages of different sizes. Without sophisticated optimizations, the current run times for an hour of video sampled at 1 fps are 13 s, 86 s, 216 *s*, and 74 s for *time*, *inception*, *uid*, and *scda*, respectively.

[4]See https://github.com/luiscarlosgph/videosum/blob/main/test/data/video.mp4

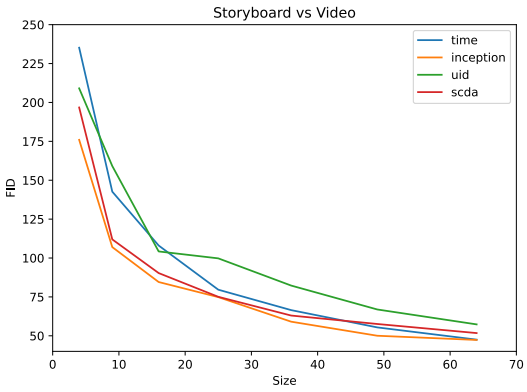

Fig. 5 Fréchet Inception Distance (FID) between the Gaussian distribution estimated from the frames in the storyboard and that obtained using all the frames in the video (lower is better, as it indicates that the storyboard is closer to the original video). The size indicates the number of video frames collected in the storyboard.

DISCUSSION

As shown in Fig. 3, treating the latent vector of a frame as set of samples of a univariate Gaussian distribution (*uid* method) leads to noisy clustering. In terms of running time, *inception* and *scda* are comparable, but the improved performance of *inception* across a wide range of storyboard sizes makes it an ideal candidate to serve as an unsupervised baseline for surgical video summarization.

REFERENCES

[1] L. Maier-Hein *et al.*, "Surgical data science – from concepts toward clinical translation," *Medical Image Analysis*, vol. 76, p. 102306, feb 2022.

[2] A. d. S. Costa, "Assessment of operative times of multiple surgical specialties in a public university hospital." *Einstein (Sao Paulo, Brazil)*, vol. 15, no. 2, pp. 200–205, 2017.

[3] E. Apostolidis, E. Adamantidou, A. I. Metsai, V. Mezaris, and I. Patras, "Video Summarization Using Deep Neural Networks: A Survey," *Proceedings of the IEEE*, vol. 109, no. 11, pp. 1838–1863, nov 2021.

[4] E. Apostolidis, G. Balaouras, V. Mezaris, and I. Patras, "Combining Global and Local Attention with Positional Encoding for Video Summarization," in *2021 IEEE International Symposium on Multimedia (ISM)*. IEEE, nov 2021, pp. 226–234.

[5] X.-S. Wei, J.-H. Luo, J. Wu, and Z.-H. Zhou, "Selective Convolutional Descriptor Aggregation for Fine-Grained Image Retrieval," *IEEE Transactions on Image Processing*, vol. 26, no. 6, pp. 2868–2881, jun 2017.

[6] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *CVPR*, 2016.

[7] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium," jun 2017.