Fast Shot Boundary Detection with Fully Convolutional Neural Networks

Michael Gygli
Google Research, Zurich
gyglim@google.com

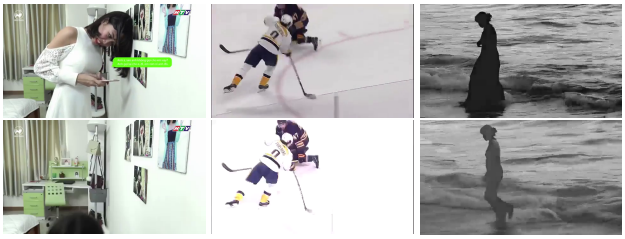
Abstract

Shot boundary detection (SBD) is an important component of many video analysis tasks, such as understanding instructional videos. We propose to learn shot detection end-to-end, from pixels to final shot boundaries, using a Convolutional Neural Network (CNN) which is fully convolutional in time. This allows to use a large temporal context without repeatedly processing frames. With this architecture, our method obtains state-of-the-art results while running at an unprecedented speed of more than 120x real-time. For training our model, we rely on our insight that all shot boundaries are generated and create a dataset with one million frames and automatically generated transitions.

1. Introduction

A shot of a video consists of consecutive frames which show a continuous progression of video and are thus interrelated. The goal of shot boundary detection is to predict when such a shot starts or ends. Representing a video as a set of shots is useful for many video analysis task, such as understanding instructional videos, among others. Due to the broad use of SBD, it has been researched for over 25 years [1] and many methods have been proposed, *e.g.* [1, 2, 3]. But shot detection is not solved yet. While it may seem a simple problem, as humans can easily spot most shot changes, it is challenging for an algorithm. This is due to several reasons, as illustrated in Figure 1. To tackle these challenges we propose to learn shot boundary detection end-to-end, by means of a fully convolutional neural network.

We make the following contributions: (i) A way to generate a large-scale dataset for training shot detection algorithm without the need to manually annotate them and (ii) a novel and highly efficient CNN architecture by making it fully-convolutional in time, inspired by fully convolutional architectures for image segmentation [4]. Our method runs at 121x real-time speed on a Nvidia K80 GPU.



(a) Adjacent frames from different shots. (b) Single shot with strong frame to frame variation. (c) Frames of a dissolve, 0.5 sec apart, but visually similar.

Figure 1: Challenges of shot detection. Understanding if a scene shows strong variation or if a shot change occurs is often difficult.

2. Method

We pose shot boundary detection as a binary classification problem. The objective is to correctly predict if a frame is part of the same shot as the previous frame or not. From this output, it is trivial to obtain the final shot boundaries: We simply assign all frames that are labelled as "same shot" to the same shot as the previous frame.

Network architecture. To solve the classification problem described above, we propose to use a Convolutional Neural Network with spatio-temporal convolutions, which allows the network to analyze changes over time. The network takes 10 frames as input, runs them through four layers of 3D convolutions, each followed by a ReLU, and finally classifies if the two center frames come from the same shot or if there is an ongoing transition. Using 10 frames as input provides context around the frames of interest, something that is important to correctly detect slow transitions such as dissolves. We use a small input resolution of 64x64 RGB frames for efficiency and since such low resolution are often sufficient for scene understanding [6]. Our network consists of only 48698 trainable parameters.

Fully convolutional in time. Our architecture is inspired by C3D [7], but is more compact. More importantly, our model consists of 3D convolutions only, thus making the network fully convolutional in time. The network is given

	F1 score	Speed
Apostolidis et al. [2]	0.84	3.3x (GPU)
Baraldi et al. [3]	0.84	7.7x (CPU)
Song et al. [5]	0.68	33.2x (CPU)
Ours	**0.88**	**121x (GPU)**
w/o fully conv. inference		13.9x (GPU)

Table 1: Performance and speed comparison on the RAI dataset [3]. As can be seen our method significant outperforms previous works, while being significantly faster. We note, however, that [3] uses a single-threaded CPU implementation, while we run our method on a GPU.

an input of 10 frames, and trained to predict if frame 6 is part of the same shot as frame 5. Due to its fully convolutional architecture, however, it also accepts larger temporal inputs. E.g. by providing 20 frames, the network would predict labels for frames 6 to 16, thus making redundant computation unnecessary. This allows to obtain large speedups at inference, as we are showing in our experiments.

Implementation details. We use a cross-entropy loss, which we minimize with vanilla stochastic gradient descent (SGD). At inference, we process the video in snipets of 100 frames, with an overlap of 9 frames. If a frame is part of a transition such as a dissolve, it is labelled as *not* the same shot as the previous, as it is part of a transition, not a shot.

3. Training Data

To obtain a dataset large enough to train an end-to-end model we create a new dataset automatically. The dataset consists of 79 videos with few or no shots transitions and has a total duration of 3.5 hours. From this data we sample snippets of 10 frames, which will serve as the input to our model. To generate training data we combine some of these snippets with a transition. Thus, we have two types of training examples: (i) snippets consisting of frames from a single shot, i.e. non-transitions and (ii) transition snippets, which have a transition from one shot to another.

We generate the following transitions: Hard cuts, Crop cuts (hard cut to a zoomed in version of the same scene), dissolves, fade-ins, fade-outs and wipes. Dissolves linearly interpolate between shots, while fades linearly interpolate from or to a frame of a single color. In wipes a shot is moved out to a side, while the next shot is moved in.

4. Experiments

We evaluate our method on the publicly available RAI dataset [3]. Thereby we follow [3] and report F1 scores. We now discuss these results, which are summarized in Table 1.

Performance. As Table 1 shows, our method outperforms the previous state of the art methods on this dataset. Our method improves the mean F1 score from 84% to 88%, thus reducing the errors by 25%. It obtains an accuracy of more than 90% in recall and precision on most videos. We find that the lower precision on some videos stems from custom transitions such as partial cuts, which are special cases and extremely hard to detect. Furthermore, such transitions were not included into training. On common transitions such as hard cuts, fades and dissolves our method is extremely accurate.

Speed comparison. We measure speed on a machine with an single Nvidia K80 GPU and 32 Intel Xeron CPUs with 2.30GHz. We measured the speed of [5] and our method on this machine, using the RAI dataset. For [2, 3] we used the timings provided in the paper instead. Thus, these numbers don't allow for an exact comparison, but rather give a coarse indication of the relative speed of the different methods.

From Table 1 we can see that our method is much faster than previous methods. We also show that making the architecture fully convolutional is crucial for obtaining fast inference speed. To the best of our knowledge, our model it is the fastest shot detection to date.

5. Conclusion

In this paper we have introduced a novel method for shot boundary detection. Our CNN architecture is fully convolutional in time, thus allowing for shot detection at more than 120x real-time. We have compared our model against the state-of-the art on the RAI dataset [3] and have shown that it outperforms previous works, while requiring a fraction of time. We also note that our model was not using any manually annotated shot transitions for training.

References

[1] A. Akutsu, Y. Tonomura, H. Hashimoto, and Y. Ohba. Video indexing using motion vectors. In *Applications in Optical Science and Engineering*, 1992. 1

[2] E. Apostolidis and V. Mezaris. Fast shot segmentation combining global and local visual descriptors. In *ICASSP*, 2014. 1, 2

[3] L. Baraldi, C. Grana, and R. Cucchiara. Shot and scene detection via hierarchical clustering for re-using broadcast video. In *CAIP*, 2015. 1, 2

[4] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1

[5] Y. Song, M. Redi, J. Vallmitjana, and A. Jaimes. To click or not to click: Automatic selection of beautiful thumbnails from videos. In *CIKM*. ACM, 2016. 2

[6] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *PAMI*, 2008. 1

[7] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015. 1