Scene Separation & Data Selection: Temporal Segmentation Algorithm for Real-Time Video Stream Analysis

Yuelin Xin[1,2,*,†], Zihan Zhou[1,2,†] and Yuxuan Xia[1,2,†]

[1]Southwest Jiaotong University, Chengdu, China
[2]University of Leeds, Leeds, UK

Abstract

We present 2SDS (Scene Separation and Data Selection algorithm), a temporal segmentation algorithm used in real-time video stream interpretation. It complements CNN-based models to make use of temporal information in videos. 2SDS can detect the change between scenes in a video stream by com-paring the image difference between two frames. It separates a video into segments (scenes), and by combining itself with a CNN model, 2SDS can select the optimal result for each scene. In this paper, we will be discussing some basic methods and concepts behind 2SDS, as well as presenting some preliminary experiment results regarding 2SDS. During these experiments, 2SDS has achieved an overall accuracy of over 90

Keywords

scene separation, temporal segmentation, real-time video analysis, dHash

1. Introduction

Image recognition models have gone increasingly accurate in the past few years, yet video semantics tasks are still challenging. A detailed comprehension on video stream could play a significant part in video accessibility [1], surveillance footage auto-interpretation [2, 3], and so on. These technologies have already been proven useful on large video platforms like YouTube, used for real-time video interpretation and video topic analysis.

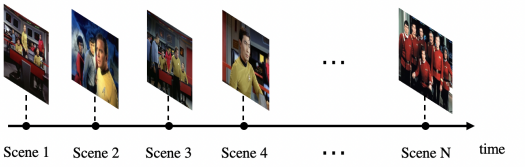

Figure 1: Overall effect of the scene separation procedure. The whole video stream will be separated into scenes, in each of which the images in the video remain relatively stationary.

1.1. The Problem

In the processing of video stream, a 2D CNN can be extended into 3D CNN by adding a temporal dimension [4], but this approach can be hazardous if the video is too long, or it is of indefinite length. However, a 2D CNN is still very usable in a traditional image recognition or image segmentation task.

The problem is that 2D CNNs only recognise a video as discrete images, rather than a continuous stream of images. This poses some issues. For example, a CNN model could not resolve the motion of a person (e.g., walking, dancing) be-cause the person is stationary in every frame, and this will cause the loss of significant information in video analysis. So, we need to devise an implementation that complements the CNN model to solve the continuity issue. This implementation should group the discrete frames (adjacent on the temporal axis) that look similar to each other into *scenes*, this procedure is what we call *temporal segmentation* (also referred as *scene separation* in 2SDS, see Fig. 1 for example).

1.2. Related Work

SlowFast Networks. The SlowFast Networks use a two-pathway architecture for video recognition, the slow pathway (low frame rate) is used to capture spatial semantics, and the fast pathway (high frame rate) is used to capture temporal semantics like motions in a relatively fine temporal resolution [5].

1.3. Our Work

What we have achieved is to devise the temporal segmentation algorithm, 2SDS, which stands for "Scene Separation and Data Selection algorithm". It can slice the video stream into segments on the temporal axis, so it can be interpreted using 2D CNN models while preserving critical

STRL'22: First International Workshop on Spatio-Temporal Reasoning and Learning, July 24, 2022, Vienna, Austria
*Corresponding author.
†These authors contributed equally.
sc20yx2@leeds.ac.uk (Y. Xin); sc20zz2@leeds.ac.uk (Z. Zhou); sc202yx@leeds.ac.uk (Y. Xia)
0000-0002-9732-2414 (Y. Xin); 0000-0003-2613-7569 (Z. Zhou); 0000-0002-1185-2722 (Y. Xia)

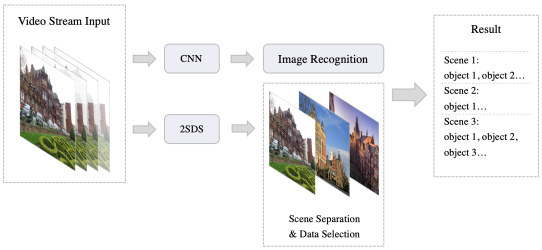

Figure 2: 2SDS used together with a CNN model. The 2SDS algorithm can separate the scenes in a continuous video stream and select the result produced by the CNN model, the two together, can produce a scene-separated recognition result.

information on the temporal dimension. By combining 2SDS with a CNN model (Fig. 2), this implementation is similar to the SlowFast Networks on splitting the input into two pathways, in which the 2SDS is similar to the fast pathway of the SlowFast Networks, except we do not introduce another neural network, but we replace the network with the faster 2SDS, which guarantees even better temporal resolution.

2. Motivation: Why Not Neural Networks

Traditionally, RNN-based models have been quite successful in processing sequential information like time. However, the usage of RNN or even neural networks is not practical in time sensitive tasks like real-time object recognition and live video stream analysis, which requires fast responding algorithms, and RNNs usually cannot meet those requirements.

RNN-based models like LSTM [6] generally have a longer respond time even compared to CNN-based models (although the difference between them vary with different settings of hyperparameters). A CNN + RNN architecture model would mean the doubling of processing time, which is something we would rather avoid when dealing with video stream analysis tasks.

However, RNNs do have the advantage of acting upon temporal information, especially for models like LSTM. So, we need to help the CNN-based models to preserve temporal information, and that is where we introduce our temporal segmentation algorithm, 2SDS.

By adding the 2SDS algorithm, alongside a CNN model, we were able to achieve RNN-like results. In the meantime, by avoiding the introduction of a neural network, this implementation is also faster than the CNN + RNN architecture or the CNN + CNN architecture.

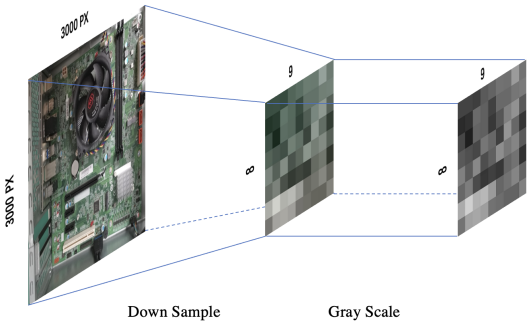

Figure 3: Image processing in 2SDS based on an improved dHash algorithm. The two image processing parts in 2SDS, the first step is down sample, and the second step is gray scale conversion.

3. Method: 2SDS

2SDS stands for "Scene Separation and Data Selection algorithm". It works as a temporal segmentation and result selection algorithm to complement CNN-based models. It contains a two-part procedure of separating the video stream into segments and selecting a representative recognition result from the CNN model for output.

2SDS utilises the difference hash (dHash) method [7] to obtain the rough image difference between two frames, if two frames have a very little difference, they will be grouped into the same scene. This method involves a few simple steps to calculate, and it is the most important method 2SDS uses to achieve scene separation. As the calculation is relatively simple and straight forward, this makes 2SDS extremely fast on scene separation.

Also, 2SDS uses a pooling-layer-like data smoothing and data selection method to pick out the representative recognition result for a particular scene. This method can generally improve the accuracy of the output because it can smooth out the data on undesired frame moving (e.g., camera shaking, broken frames). Alongside the data smoothing mechanism, another data selection mechanism is implemented to select the representative recognition result (referred as *representative* in the following sections) from the whole data segment of a scene (referred as *candidate* in the following sections).

3.1. Scene Separation: based on dHash

The scene separation procedure of 2SDS (Fig. 3) is based on an improved version of the dHash algorithm, which is originally used to judge the similarity of two images. By applying the scene separation procedure, the temporal information can be preserved by the sequencing of the separated scenes. The exact workflow of the scene separation process is discussed extensively below.

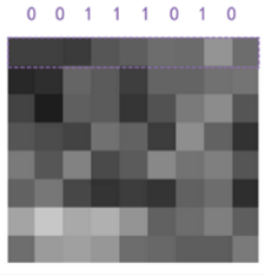

Figure 4: Example on binary sequence conversion. The derived binary sequence of row 1 in this case is 00111010.

Down sampling. To make a rough comparison between two frames in a video, the frames need to be down sampled from their original size to an 8 by 9 (row by column) sub-image. This approach can both simplify the remaining calculation and make the algorithm less sensitive to subtle changes between frames.

Gray scale. We apply gray scale manipulation on the previous sub-image using the Luminosity algorithm, this step is purely for reducing the complexity of calculating the difference on 3 channels. By converting the RGB channels into one gray scale channel, this approach dramatically lessens the complexity of the algorithm.

Calculate Hash value. The derived gray scale image is converted into a single 16-bit hexadecimal hash value. The algorithm looks at all the 8 rows separately, each row has 9 gray scale values from 0 to 255. These 9 values are converted to 8 binary numbers under the following rules:

(a) One binary value stands for the gray scale difference between two adjacent pixels.
(b) If the gray scale value of the pixel on the left is greater than the pixel on the right, the binary value should be 1, otherwise, it should be 0.
(c) Every row should end up with an 8-bit long binary sequence.

An example is given in Fig. 4.

Using this method, we can derive eight 8-bit long binary sequences, each of them can be represented by a 2-bit long hexadecimal value. And by concatenating all the 2-bit hexadecimal values, we can obtain a 16-bit long hexadecimal hash value, and this value will represent the whole image (this is also the reason why the original image is down sampled into an 8 by 9 sub-image rather than an 8 by 8 sub-image, because the 8 by 8 image will face some inconvenience when converting into a hexadecimal hash value).

Calculating the Hamming distance. By calculating the Hamming distance between the hash values of two adjacent frames, we can judge whether the two frames are in the same scene or not. If the Hamming distance is greater than a threshold (usually 5), we consider the two frames to be in two different scenes, and we can separate them accordingly. For the calculation of Hamming distance, we can simply use an Exclusive Or operator on the two hash values, here is an example below (the Hamming distance is 7 in this case):

$$\text{c4e0d8988c989898} \oplus \text{eee6989c8c989898} = 7 \quad (1)$$

3.2. Data Selection and Data Smoothing

When the scene separation process detected a new scene, the data collected on the previous scene is packed into an array. This array contains all the recognition results produced by the CNN model in the previous scene, and the CNN model would have a recognition output on every frame in this scene.

To have a solid output for 2SDS, we need to perform 2 extra steps: data smoothing and data selection. The method that is implemented here is inspired by the pooling layer in a convolutional neural network.

Data smoothing procedure. This step is also called LWAP (Length Weighted Average Pooling). We start by segmenting the array containing all the recognition data into small groups of a defined size. Then, we apply the following formulas:

$$WAL = \frac{\sum_{i=1}^{i\leq\varphi}(L_i \times \omega_i)}{\sum_{i=1}^{i\leq\varphi}\omega_i} \quad (2)$$

$$\begin{cases} f_i(D) = \min_{i=0} |\operatorname{card}(D_i) - \mathrm{WAL}| \\ C_I = [c \in D \mid \operatorname{card}(c) = f_i(D)] \end{cases} \quad (3)$$

Here, L_i stands for the length of each recognition data, for example, in "object 1, object 2, object 1", $L_i = 3$. ω_i stands for the weight of each recognition data which is $0.1 \times L_i$. $\operatorname{card}(D_i)$ stands for the cardinality of set D_i, where D_i is the segments previously obtained by segmenting the original array.

This approach is inspired by the pooling layer in CNN, but instead of a Max Pooling operation, the data smoothing procedure uses a Weighted Average Pooling operation. By using this data smoothing procedure, we can avoid unwanted recognition results like broken frames or camera flashes.

Data selection procedure. We apply a data selection procedure that uses the similar approach that we previously used in the data smoothing procedure, which is also a Weighted Average Pooling operation. This procedure will select the recognition result from a frame whose feature intensity is the closest to the weighted average value of all the candidates (feature intensity refers to the number of different classes of objects in a particular frame).

Finally, we can output the result that we obtained in the previous steps as the representative of the whole

Table 1
Interview video tests results.

Experiment No.	Output - Truth	Accuracy
Interview 1	25 - 25	100.00%
Interview 2	35 - 29	82.86%
Interview 3	31 - 28	90.32%

scene. This particular recognition result will be used to represent the whole scene it is located in, and through the help of NLP and other models, this can even be used to output the natural language interpretation of this video scene.

4. Experiments

Due to the lack of similar algorithms and datasets, we could only provide some preliminary and experimental usage of the 2SDS algorithm[1].

We choose YOLOv5s as our image recognition CNN for this experiment, and we have built an experimental dataset on video object detection using selected YouTube videos in the YouTube-VOS dataset [8]. Although the YOLOv5s algorithm is trained on the COCO dataset, this CNN model is still sufficiently usable in this experiment for it is not the key focus of this experiment.

We are most interested in how 2SDS will perform in scene separation (temporal segmentation) tasks. We classified the testing videos into 3 classes: interviews, vibrant, and hybrid.

The interviews are usually straight forward and easier to undergo scene separation tasks. Vibrant videos are the more difficult ones due to their fast-moving images and transition effects that might seem deceptive to 2SDS. The hybrid video sits in between the first two types, they have some features of the interview videos, as well as features from the vibrant videos, their difficulty should sit in the middle.

4.1. Interview Video Tests

We conducted 3 separate experiments using interview videos (Table 1). The total amount of scenes in these 3 experiments is 82. The overall accuracy of 2SDS during these experiments is 90.10%. There are 2 cases where we find the 2SDS algorithm actually over-judged the transition between two scenes. This is potentially a sensitivity issue posed by the hard coded threshold during scene separation.

[1]We only did some preliminary experiments on the accuracy of 2SDS on scene separation (temporal segmentation) tasks, more detailed experiments are still needed to be conducted.

Table 2
Vibrant video tests results.

Experiment No.	Output - Truth	Accuracy
Vibrant 1	9 - 13	69.23%
Vibrant 2	19 - 38	50.00%

Table 3
Hybrid video test result.

Experiment No.	Output - Truth	Accuracy
Hybrid 1	105 - 106	99.06%

4.2. Vibrant Video Tests

We conducted 2 separate experiments using vibrant videos (Table 2). The total amount of scenes in these 2 experiments is 51. The overall accuracy of 2SDS during the two experiments is 54.90%. The accuracy in vibrant videos is substantially lower than interview videos for the 2SDS is unable to separate two fast-moving scenes effectively. It is important to notice that we used harsh videos like sport videos and dynamic advertisement videos in this experiment, so the performance of the 2SDS is expected to be much lower comparing to the previous experiment. This is the biggest limitation of 2SDS, but this issue is addressable with future improvements of the algorithm.

4.3. Hybrid Video Tests

We conducted one experiment using a long hybrid video (Table 3). The total amount of scenes in this experiment is 106. The overall accuracy of 2SDS is 99.06%. Theoretically, the result of this experiment should sit between the previous two tests, however, an anomaly has arisen most likely due to the lack of samples. A more detailed experiment should be conducted to further determine the accuracy of 2SDS on hybrid videos.

5. Bringing in Spatial Information

Bringing in spatial information and modeling techniques can potentially play a huge role in video interpretation. Previously difficult and untouchable problems like continuous gesture recognition and scene recognition are being cracked using the CNN-based spatio-temporal reasoning model [9] and the 2SDS algorithm as well.

Our work has only utilised the temporal information in video stream, our future work can make use of graphs, and spatially model a frame into a graph, with the objects as the vertices and the spatial relations between the objects as the edges, like the MST-GNN [10] and the VRD-GCN [11]. Doing this, we can extract even more

information out of a video. For example, a person's gesture in a scene can be identified, and the scenes with more significant camera or object movements (e.g., the vibrant and hybrid video tests) will not cause significant problem for the algorithm because the spatial relation of the objects stays the same.

This future work would bring immense potential with the use of spatial information, which will add a whole other dimension of usable information that can benefit video analysis with richer semantics and the ability of grouping fast-moving frames, bringing video interpretation models yet another step closer to how human perceive visual information.

6. Conclusion

Under the context of real-time video stream analysis using temporal segmentation methods, we devised 2SDS, a temporal segmentation algorithm that can be used alongside CNNs to complement for the lack of temporal information handling ability of the CNN-based models. We gave, yet another powerful tool that CNN models can utilise, the ability to take advantage of the inherent temporal aspect of videos. Video stream analysis is a completely different task compared to image recognition, and we are finally seeing some evidence that we can still use 2D CNNs to interpret video information.

The 2SDS algorithm utilise a refined difference hash value method and a novel data smoothing and data selection technique to crack the temporal segmentation problem. Although there are still drawbacks with fast-moving frames in vibrant videos, the 2SDS algorithm has already done a great job at separating relatively simple and stationary scenes in videos, and it gets the job done at a respectful speed, which will allow the 2SDS to get a finer temporal resolution compared with neural networks.

For future work, some improvements on 2SDS (e.g., adding graphs to model spatial relations) can potentially boost the algorithm's performance on fast-moving scenes and smooth transitions.

Acknowledgments

We would like to dedicate our thank to Dr Zhiguo Long, Dr John Stell, and Dr Liu Yang, who have been extremely generous and helpful throughout the course of our work.

References

[1] L. Stappen, A. Baird, E. Cambria, B. W. Schuller, Sentiment analysis and topic recognition in video transcriptions, IEEE Intell. Syst. 36 (2021) 88–95.

[2] A. S. Patel, R. Vyas, O. P. Vyas, M. Ojha, A study on video semantics; overview, challenges, and applications, Multim. Tools Appl. 81 (2022) 6849–6897.

[3] R. Pal, A. A. Sekh, D. P. Dogra, S. Kar, P. P. Roy, D. K. Prasad, Topic-based video analysis: A survey, ACM Comput. Surv. 54 (2021) 118:1–118:34.

[4] A. Diba, M. Fayyaz, V. Sharma, A. H. Karami, M. M. Arzani, R. Yousefzadeh, L. V. Gool, Temporal 3d convnets: New architecture and transfer learning for video classification, CoRR abs/1711.08200 (2017). arXiv:1711.08200.

[5] C. Feichtenhofer, H. Fan, J. Malik, K. He, Slowfast networks for video recognition, in: 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019, IEEE, 2019, pp. 6201–6210.

[6] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (1997) 1735–1780.

[7] N. Krawetz, Kind of like that, 2013. URL: http://www.hackerfactor.com/blog/?/archives/529-Kind-of-Like-That.html.

[8] N. Xu, L. Yang, Y. Fan, D. Yue, Y. Liang, J. Yang, T. S. Huang, YouTube-VOS: A large-scale video object segmentation benchmark, CoRR abs/1809.03327 (2018). arXiv:1809.03327.

[9] O. Köpüklü, F. Herzog, G. Rigoll, Comparative analysis of CNN-based spatiotemporal reasoning in videos, CoRR abs/1909.05165 (2019). arXiv:1909.05165.

[10] M. Li, S. Chen, Y. Zhao, Y. Zhang, Y. Wang, Q. Tian, Multiscale spatio-temporal graph neural networks for 3D skeleton-based motion prediction, IEEE Trans. Image Process. 30 (2021) 7760–7775.

[11] X. Qian, Y. Zhuang, Y. Li, S. Xiao, S. Pu, J. Xiao, Video relation detection with spatio-temporal graph, in: L. Amsaleg, B. Huet, M. A. Larson, G. Gravier, H. Hung, C. Ngo, W. T. Ooi (Eds.), Proceedings of the 27th ACM International Conference on Multimedia, MM 2019, Nice, France, October 21-25, 2019, ACM, 2019, pp. 84–93.