Evolving Automatic Frame Splitters

Andy Song
School of CS & IT
RMIT University, Australia
andy.song@rmit.edu.au

Feng Xie
School of CS & IT
RMIT University, Australia
feng.xie@rmit.edu.au

ABSTRACT

This paper extends the application of Genetic Programming into a new area, automatically splitting video frames based on the content. Compared with human written video splitting programs, GP generated splitters are more accurate. Moreover these video splitting programs have high tolerance to noises. They can still achieve reasonable performance even when the noisy videos are not easily recognizable by human eyes.

Categories and Subject Descriptors

I.2.2 [**Artificial Intelligence**]: Automatic Programming; I.2.10 [**Artificial Intelligence**]: Vision and Scene Understanding—*Video Analysis*; I.4.8 [**Image Processing and Computer Vision**]: Scene Analysis

General Terms

Algorithms

Keywords

Genetic Programming, Pattern Recognition, Video Processing, Frame Splitting

1. INTRODUCTION AND METHODOLOGY

One important task in video processing is splitting a long video stream into several short clips according to the content. This process would facilitate tagging and organizing video data.Content-based frame splitting is a basic operation supported by most video editing tools. However it is usually done manually by users. Automatic splitting is also available which is based on comparing the statistical differences over consecutive frames. However the threshold is often set manually by the developer or user. To address the above issues, we use Genetic Programming (GP) to evolve frame splitters to minimize human input.

GP has also been widely used in image processing and video processing related areas. Poli [1] reported a GP method to identify blood vessels in X-ray coronarograms, and to segment the brain in Magnetic Resonance images. Zhang et al[2] adopted GP to analyse images of the retina for identifying haemorrhages and micro aneurisms. Trujillo and Olague [3] successfully used GP to mark interest points in images, which helped identify edges of objects and changes in colour and illumination. We used GP to evolve motion detection programs to recognize interesting motions from unstable background[4].

GP programs take frame features as input which are extracted based on frame differences. These features are much compact than an entire frame. The first three sets of frame features are frame differences in intensity(f_1), RGB values(f_2) and HSV values(f_3). These features are based on a full-size video frame. In reality it is possible that two different scenes contain similar content and only differ in a small part. Therefore we propose another three sets of features which contains spatial information.

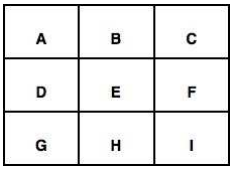

Figure 1: Dividing a Frame into Sub-regions

As shown in Figure 1, a frame is evenly divided into nine sub-regions. Then features are calculated based on comparing one sub-region over two consecutive frames. They are calculated in intensity(f_4), RGB values(f_5) and HSV values(f_6).

The GP function set we used consists of common arithmetic and logic operators: $+, -, *, /, if, >, <, ==$. There are also additional three more complex functions: sAVG, sMAX and sMIN, which all have six arguments. The average, maximum and minimum values of these six values will be returned by these functions. The rationale behind these functions is that they may act like basic feature extraction operators, and be able to find more distinctive differences between normal frames and frames containing scene boundaries.

The terminal set simply contains the feature values and random numbers ranged from 0 to 1. The fitness function is the classification accuracy on training data set.

2. EXPERIMENTS AND RESULTS

A software tool has been developed in this study. It allows users to mark the scene boundaries between frames manually, then generates training data accordingly. Each record in the generated data consists of values calculated based on the aforementioned features over a pair of neighboring

GECCO'11, July 12–16, 2011, Dublin, Ireland.
ACM 978-1-4503-0690-4/11/07.

frames. If that pair contains a boundary, then a class label "1" is added at the end of the record. Otherwise, a "0" is added at the end. In total there were 985 samples generated for the experiments. Two thirds of them were randomly selected for training and the rest were reserved for test.

Table 1: Results of Using All Features

	Training	Test
Accuracy	100%	98.27%
True Positive (TP)	100%	98.79%
True Negative (TN)	100%	98.22%

Table 1 shows the average results of 10 GP runs. Other than the accuracies, the percentage of true positives and true negatives are also presented.

One benefit of using GP is that the evolved programs usually have low complexity. This is the case in this study as well. The frame splitters based on all feature combined have around 50 nodes on the program trees. This is much smaller than the possible max tree size, $2^9 - 1 = 511$ nodes, where 9 is the maximum tree depth defined in our experiments. Only around 10 features were used as the program input. As the result, the evaluation time for these generated programs is relatively trivial, less than 0.001 second for processing each sample.

In reality, noise is often an issue that can not be ignored. To test whether GP has such capability or not, we added different degrees of random noises into all the videos. The test accuracies at different noisy levels are listed in Table 2, which also shows the false positive rate (FP) and false negative rate (FN).

Table 2: Performance on Noisy Videos

Noise Level	Test Accuracy	FP	FN
64	100%	0	0
128	98.07%	0	5.88%
256	98.07%	0	5.88%
512	88.46%	0	35.29%
1024	78.84%	2.85%	58.82%
2048	69.23%	11.42%	70.58%

The results show that the performance of evolved splitters drops gradually with the increase of noise level. However there was little change in accuracy when the noise level was below 512. Even at level 512, GP still achieved a reasonable accuracy of 88.46%. There was no false positives, but with 35.29% false negatives.

Furthermore we compared evolved GP programs with a well-known video editing tool, AVS Video ReMaker [5] developed by Online Media Technologies Ltd., which is a company specialized in digital video and multimedia processing. We used their most recent version, 4.0.2.126 for the comparison.

Table 3: GP vs. Human Written Program

	GP Program	AVS Video ReMaker
Video Set 1	17 out of 17	14 out of 17
Video Set 2	66 out of 66	64 out of 66
Total Accuracy	99.14 %	93.97%

There were two sets of videos for test. The first one has 17 boundaries and the second one contains 66 boundaries. Both of them were applied to the best GP program evolved using only RGB features ($f_2 + f_5$), and tested in AVS Video ReMaker. The result is shown in Table 3.

The splitter evolved by GP recognized all the 83 scene boundaries from the two video sets. However there was a false positive, so the total accuracy was 99.14%. In comparison AVS Video ReMaker did not generate false positives, but false negatives. It missed 5 boundaries in total.

3. CONCLUSIONS AND FUTURE WORK

This paper presents a Genetic Programming methodology for evolving frame splitters. Our investigation shows that based on a set of frame features, GP can build programs of high accuracy in detecting scene boundaries. The evolved programs are small in size. That facilitates real-time processing of video input. Further investigation has shown that the GP method is capable of handling noises. Reasonable performance can be achieved even when the noise level is high even for human eyes. This shows that GP is a robust approach and suitable for real world applications.

The comparison between our GP method and a published video processing tool shows that the evolved program can outperform human written programs. That is a convincing evidence that GP is human competitive and is a good alternative in finding better solutions. Based on this investigation we conclude that GP is suitable for frame splitting. Highly accurate programs can be evolved by this general method.

In the near future we will investigate how to extend the GP method to handle fade-in-fade-out and blackscreen boundaries, so the evolved program can be more flexible in handling different real-world scenarios. Another issue that we would like to explore is handling unbalanced data.

4. REFERENCES

[1] R. Poli, Genetic Programming for Feature Detection and Image Segmentation, *Technical Report, The University of Birmingham, School of Computer Science*, UK, 2000.

[2] M. Zhang, V. B. Ciesielski and P. Andreae, A Domain Independent Window Approach to Multiclass Object Detection Using Genetic Programming, *EURASIP Journal on Applied Signal Processing*, (8): 841-859, 2003.

[3] L. Trujillo and G. Olague, Automated Design of Image Operators that Detect Interest Points, *Evolutionary Computation*, 16(4):483-507, 2008

[4] B. Pinto and A. Song, Detecting Motion from Noisy Scenes using Genetic Programming, *IVCNZ 2009: Proceedings of 24th International Conference on Image and Vision Computing New Zealand*, pages 322-327, Wellington, New Zealand, IEEE, 2009.

[5] AVS Video ReMaker, `http://www.avs4you.com/AVS-Video-ReMaker.aspx`.