Project: SVM-Based Shot Boundary Detection Using Field-Programmable Gate Arrays

Chun F.Hsu
Lab of System-On-A-Chip, Department of Computer Engineering
National Taiwan University
Email: http://r94082@csie.ntu.edu.tw

Mong-Kai Ku
SOC Lab, Dept. of CSIE
National Taiwan Univeristy

***Abstract*—This paper proposed a hardware/software codesign solution to shot boundary detection(SBD) using Support-Vector-Machine(SVM) as the recognition engine. A fully-parallel SVM hardware is implemented on the field-programmable gate array(FPGA). SBD is detected by our SVM module in a live continuous video stream. We used Color Histogram as our primary features and Histograms intersection with temporal kernel filtering in zero-time as our second and the input feature to SVM. With SBD as the primary component in the multimedia content analysis framework, our system is flexible and extensible allowing more high semantic applications attached after our SBD module using duplicated SVM cores. Comparisons to software programas show a reliable detection accuracy. The SVM hardware presented here can run speed up to 255 MHZ on FPGA and a small amount of area, thus it can meet real-time tasks, such as live-TV news or sports game.**

I. Introduction

A. Motivation

Over the past 10 years, with the increasing size of storage products, and the rapid development of video capture and editing technology. More and more digital video data are produced and keeped everyday, including home video, news video, surveillance video, and TV programs. The management to these great amount of data becomes a serious topic. Although the software progress have excellent effort on extracting and analysize the semantic contents these years, there are none of exploration using hardware to accelerate the performance in this field. If the Miltimedia Content Analysis(MCA) can run in real-time, whether in the producing stage or the playing stage. The possibility of binding more MCA challanges in a single system, such as temporal segmentation of video streams into shots, scenes, and classification of shots contents, or even advance and commerce topic, like finding the duplicates of copyright video, is able to run in faster and power-saving on a System-On-Chip(SOC) system.

B. Overview of Shot Boundary Detection

Shot boundary detection is an essential step toward semantic video retrieval and indexing. Video information retrieval is mainly based on a layer hiearchy of shot, scene and story [1] . Among these. shot is the most basic temporal unit in this hiearchy. SBD, also called video temporal segmentation, becomes the foundation stone in video applications. There are commonly two types of shot boundary, abrupt transition(CT) and gradual transition(GT). CT is a rapid change in consecutive frames, indicating a sudden scene change or a stop of recording the video. In gradual transition, much editing techniques have been used, to make this detection work much harder. GT can be further divided into dissolve, fade-out, fade-in, zoom-in, zoom-out, wipe, etc.[2]

Due the various type of GT editing techiniques, and various legnths of GTs. It is difficult and complex to build a general model to detect the type and location of these transitions. If we label each shot boundary frame as different types of shot, it can be reduce to a pattern recognition problem of finding shots. Before year 2000, SBD systems usually extract one or more low-level features, then runs the pair-comparison using statistic similiarity measures, and decide whether a boundary occured or not using threshold, the threshold scheme requires the developers to be familiar with the characteristics of their videos[4]. After year 2000, a general model construction method is introduced and emerged various methods for models construction, such as Hidden Markov Model(HMM), Gaussian Mixture Model(GMM) and other complex machine learning schemes. The models replaced the former threshold decision methods. The ability of solving harder cut and gradual shots has increased in effort of these new methods.However, these complex methods use unlimited resources, such as power, or time to reach the optimal prediction accuracy.

In a general SBD system, it is consisted of three components: low-level time-indexed feature extraction, inter-frame feature similarity computation, and pattern models construction applying various machine learning technique to the segmentation using these technique.[8]

The video features is extracted in a two step. First, global and local color histogram or other low-level features in each frame are extracted. Each frame is represented with a low-level feature vector V_n, n is the frame ID. The inter-frame feature computes the differences between each frames in every possible pairs, with a length of N neighbors for each individual frame in the similarity matrix S, with the numbers of row

and the numbers of column as the total length of the video frames. Thus, both spatial and temporal variations is contain in this matrix. Our method adopt the correlation kernel to filter and acquire the correlation of local neighborhoods similarity only, then combined it into a single vector. We trained the model from training data using support vector machine(SVM), and feed independent test data to detect the boundary after training. In order to improve the speed in order to gain more time to do further process after segmentation, we built the SVM classifier into hardware. It needs to be note that, unlike the unlimited resources in the popular software methods, the accuracy may degrade if we constrained the resources, such as the word-length or dimensions of the input features. The tradeoff beween hardware area and detection precision is need to be considered as the demand of the application.

Many believes that the feature extraction stage may be the most time-consuming as compare to SVM classifier due to the reason of complex and large amount of computations in some system. In our architecture, by the trick of signal routing, this bottleneck could be eliminated in our zero-time signal routing.

In the SVM hardware implemenation, we presented a digital fully-parallel linear kernel SVM. The hardware is flexbile due to the reason of:(1) the word-length of input vector is varied from 8-bits to 32-bits fixed-point number, (2) the input vector dimensions is from 1 32, which can support the length of the temporal kernel filter to 5. Fully-parallel structures guranteed the most optimal performance a SVM hardware engine can achieve. Finally, SVM output the label of shots for each frame in the testing video.

The rest of this paper is organized as follows. Section 2 review the related work of SBD and SVM hardware implementation. Section 3 describes SBD and our enhanced feature extracting methods. Section 4 introduces SVM and architecures of SVM hardware. Section 5 shows the result of shot detection and the cost of hardware, and section 6 concludes this work and list the future work.

II. Related Work

There are abundant literaures of SBD work, some excellent and detailed reviews and surveys include [2] and [4]. Most of the system solve cut detection very well, but remains relatively poor performance in gradual transition. The major challenge is based on the various size of length and diverse types of gradual transition, which makes the annotation and detection becomes inconsistent and complex. Inconsistent object moving velocity, camera motion and flash light effect also influence the system performance.

Some systems prefers using frame histogram differences such as L-1 and L-2 distances[1]. Our papers uses normalized L-1 distance due to the friendly hardware implementation and fast computing speed, :

$$S_{ij} = \sum_{k}^{n} \frac{V_i(k) - V_j(k)}{\sum V_i} \tag{1}$$

A. Temporal Kernel Correlation Systems

Hanjalic[10] suggested that not only pair-wise frame feature needs to be extracted, but contextual information [i.e. variations within the temporal interval nearby] shall be added. It avoid influence of noise disturbance e.g., flashlight. Cooper[8] summarized this idea and proposed a similarity analysis framework. Frames have larger similarity when they are closer. A checkerboard kernel filter the temporal variation of a single frame along the S^H's diagonal, and refer to as the kernel correlation. The maxima of correaltion points out the location of the boundary, either cut or gradual shot. After filtering, the inter-frame feature X_n for the n-th frame is :
$X_n = S^H(n-L, n-L)S^H(n-L, n-L+1)...S^H(n-L, N+L)S^H(n-L+1, n-L).....S^H(n-L+1, N+L).....S^H(n+L, n-L).....S^H(n+L, n+L)$
If all possible pairs of similarity between frames are computed, the computation order is$O(N)^2$, n is the size of total video frames. In practical, due to the size of a shot, too much information will confused the system, e.g., two similar shots in the same scene. Therefore, the size of the kernel would'nt be too large, experiments show that L=5˜10 is enough for a SBD model. This interval L could be obtained by training using cross-validation.

From its characteristics of X^H, much computing effort could be reduced. It is symmetric and it has zero dissimilarity along the diagonal. It doesn't need to compute all the portions in X^H, the complexity of computing the matrix can be reduced to$N * L^2$, which is closed to $O(N)$.

B. Support Vector Machine

Support Vector Machine (SVM) is a statistical learning theory proposed by Vapnik in 1995[5]. It has been refer as a robust and strong data modelling approach, it has been greatly used in video mining applications such as shot detection[4]. It is based on the minimizing structural risk principle to raise the generalization ability of learning machine, mainly on solving optimization problem. SVM classify data by the support vector on hyperplane, and if data is not linearly separable in its original space, SVM transform the data to a higher dimension characteristic space, that is, to map the training data through the kernel function to a higher dimension, and separate the data which is not linearly separable in the lower dimension. SVM training then obtains the support vectors and their corresponding coefficients and a bias value b.

A single SVM classification can only deal with two-classes problem, its decision is based on the sign of the decision function. The testing instance $\vec{Y}$ is classified by the sign of following decision function:

$$f(\vec{Y}) = sign(\sum_{i} \alpha_i Y_i K(\overrightarrow{Y}, \overrightarrow{X_i}) + b) \tag{2}$$

Where $\overrightarrow{Y}$ and $\overrightarrow{X_i}$ are the test vector and the i-th support vector, $K(\overrightarrow{Y}, \overrightarrow{X_i})$ is the kernel function, four common kernels are listed[12].

Linear: $K(\overrightarrow{X}, \overrightarrow{Y}) = (\overrightarrow{X} \cdot \overrightarrow{Y})$

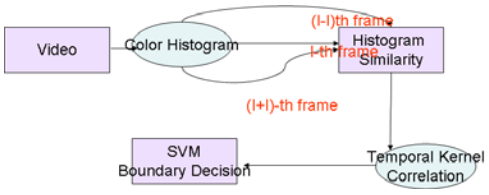

Fig. 1. SBD System Architecture

Polynomial: $K(\overrightarrow{X}, \overrightarrow{Y}) = (\overrightarrow{X} \cdot \overrightarrow{Y})^d$

Radial Basis Function(RBF): $K(\overrightarrow{X}, \overrightarrow{Y}) = (e^{\gamma * (\|\overrightarrow{X} - \overrightarrow{Y}\|)})$

Sigmoid: $K(\overrightarrow{X}, \overrightarrow{Y}) = \tanh((\overrightarrow{X} \cdot \overrightarrow{Y}) + \theta)$

For multi-class problem, Lin has proposed the One-against-one approach. Our system had modified it for hardware design purpose.

C. Hardware-Based SVM

There exist not much literature of hardware-based SVM. There are two fixed-number SVM hardware module, Kerneltron[11], a SVM classification module implementing for video object detection. The massively parallel kernel computation structure has architecture that internally analog and externally digital. Anguita precent a fully digital architecture for SVM training and classification employs linear and RBF kernels. It is a highly optimal general SVM module that can run a speed up to 35.3MHZ. Khan tried another way abandoning fixed number, and proposed a Log-domain number system linear-kernel SVM architecure. It is hardware friendly but may have bottleneck at the real-to-log number conversion back, and may not suitable for high speed multimedia application.

III. Proposed SVM-Based SBD System

A. Feature Extraction

The SBD is detected by quantifying the similarity beween frame and frame.(Figure1) First, we extract the global color histogram from the YUV color space and quantized the Y(luminance) histogram into 64-bins histogram for each frame, this will shrink the dimensions from 256 to 64, and lower the histogram complexity. Denote the histogram for N frames by the matrix H.
$H = [H_1, H_2, ..., H_N]$ The first stage low-level feature is by computing normalized L-1 distance, which is logically equivalent to histogram intersection. The similarity matrix S is obtained by computing all pair-comparison of frames. An intuitive signal routing method with the usage of delays block remove all the redundant computing time and becomes near zero-time computing(Read it and finish it).(Figure2) We adopt the row kernel, and output a vector of size $2 * L$. This routing trick will enhance the feature extraction performance and wouldn't let it left as the bottleneck.

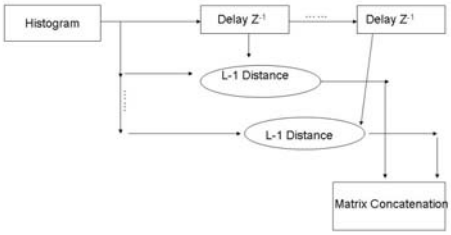

Fig. 2. Zero-time Feature Extraction Routing Algorithm

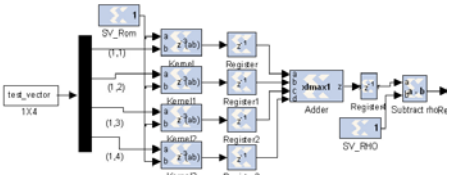

Fig. 3. Hardware SVM Architecture

B. SVM hardware implementation

Our system is design on the Simulink platform in matlab, and the hardware is run, synthesized, and programmed on the Xilinx Video Starter Kit Evaluation board, the FPGA core is Virtex IV XC4VSX35 device. Using the system generator, it is easily to co-simulate the software model on matlab and the hardware on FPGA.Our SVM model is trained by *libsvm* module[6], and selected appropriate parameters with 10-fold cross-validation to avoid the imbalance data curse, all of the software verification and training are run in Matlab environment. The support vectors(SVs) are extracted automatically from the model file generated by *libsvm*. After observation from the special characteristics of linear kernel. The new SVs can be precomputed to compact the process according to(3)

$$f(\vec{Y}) = sign(\sum_i \alpha_i Y_i K(\overrightarrow{Y} \cdot \overrightarrow{X_i}) + b) \quad (3)$$

The new SVs can be precomputed by multiply the α_i with support vector x_i, and sum them up to form a 1-by-m vector, m is the data dimension size. These m vectors will be saved in the rom. It is benefited from the hardware space for rom to store these numbers can be conserved. It also saves the time to multiply and decrease the number of multipliers. Moreover, the fully-parallel structures guaranteed the SVM will meets its most optimal speed performance. The total cycles to run a classifier is stable. It only differs when kernel change. Our linear kernel SVM only needs 6 cycles to finish the task, and increase the system performance as well. The sketch map of our SVM hardware architecures is in figure3. A rom is used to save the support vectors and the alpha coefficients. We had implemented this in fixed number, to reduce the overhead of converting back and forth from log number domain and the real number domain as compared to[3] .

For each instances, the bias b is load to the Register. The kernel block then activates and run the function between the testing data and the precomputed training vectors. The SVM classification results is output to the decision maker. The decision maker applies one-against-one method to give the final label of shot detection type.

IV. Evaluation and Synthesis Result

The SVM software we used for software model verification is *libsvm*. We used *libsvm* for training, then extracted the model data we need and inserted it to the ROM of our SVM module.

A. FPGA Synthesis Result

After synthesizing on FPGA, Table 1. shows the synthesis report among different criterions and different attributes size. It shows that our pipelined design acheive a very high speed clock rate processing the pattern data. It is hard to compared to other's effort since the system design purpose and strategies are far different from each other. But comparisons of synthesis result in certain criterion could still be made. We've compared the result with [9] on their (8,speed) design, 8 means attributes number and speed is their synthesizing criterion, and also compared with [3] on their SONAR data result. Our design still outperfrom them by the speed up up to ten times. This make it easier to insert our SVM in anywhere of the MCA framework to get real-time performance.

B. SBD System Simulation Result

We've train our system using only a 18 secs clip to detect cut shot, and tested our system on some video size more larger, these clips are extracted from internet and includes different topics. It has MTV, sport montage, and home video. The total video length is 47 mins. The *precision*, *recall* and *F1* are the three performance metrics wildly use in evaluation. N_{tp} stands for the numbers(#) of true positive, N_{fp} is the numbers of non-shot frame but detected as a shot, N_{tn} is the numbers of shot that is not detected.

From table 3, it shows that our system has a robust performance and got a high score on recall, but performed only satisfying on precision. This might be due to the data's characteristics. In our test video, flashlight and high object motions had influenced our result. These problems may solve using more training data and adapting better algorithms, since design on Simulink platform enable us to see results exactly same with software and hardware. An interesting observation is that SBD performance wouldn't degrade under different word-length of the input data and the SVs. An explanation of this could be derive from the dissimilarity between a regular frame and the boundary of a shot is significant. It wouldn't have too much decision values closed to zero to misclassify it. It shall be note that if embedded multiplier built in FPGA is used, the overall speed can increased 2 3 MHZ, but makes it more difficult to compare with others.

$$precision = \frac{N_{tp}}{N_{tp} + N_{fp}} * 100\% \tag{4}$$

TABLE I
Place and Route Synthesis Report For Different Criterions

m,dim	Slices	LUTs	Clock(MHZ)
Fix_14,10	1115	1438	255.02

TABLE II
Comparison to Others' Design

	m,dim	Slices	LUTs	Clock(MHZ)
Ours	Fix_14,10	1115	1438	255.02
[9]	Fix , 8	865	1466	35.3
[3]	Log_10,60	629	?	?

TABLE III
SBD System Results

Hardware Description	Precisions	Recalls	F1 (Percentage)
Fix_14,10	0.869	0.974	0.932

$$recall = \frac{N_{tp}}{N_{tp} + N_{tn}} * 100\% \tag{5}$$

$$F1 = \frac{2 * precision * recall}{precision + recall} \tag{6}$$

V. Conclusion and Future Works

A. Conclusions

We had introduced a hardware/software codesign framework of SVM-based shot boundary detection, our system runs at high speed and still keep the accuracy while retaining acceptable precision and recall. The results shows that the speed of overall system has been improved and it is suitable for embedded system designer to adopt our system into their designs. The SW/HW SBD system is the only work so far, and we introduce a extensible framework to make this system not only fit with SBD, but also other semantic content analysis issues. The fully-parallel SVM structure might also help other embedded hardware designers when they encountered an application that is in demand of high speed.

B. Future Work

This paper had described the prototype of a MCA system would like in hardware environment. This design will serve as foundation for low-power and high-speed MCA research.

Our future work includes employ RBF kernel on our SVM to increase the decision rate, to give more choices to users who attempt to embedded SVM in their design. With the singular exception of the recent work in[9], our goal is to build a fully-hardware SVM, including training and classifying, to achieve the goal of on-line learning .

Acknowledgment

The authors would like to thank professor C.J Lin and Winston Hsu for their specialty guidances on SVM and MCA.

REFERENCES

[1] H.J. Zhang, Low, Smoliar. “Automatic partitioning of full-motion video”, *Multimedia Systems 1*, 1993

[2] Koprinska, Carrato. “Temporal video segmentation: A survey”, *Signal Processing: Image Communication*,2001

[3] Faisal M.Khan et.al. “Hardware-Based Support Vector Machine Classification in Logarithmic Number Systems” *Circuits and Systems*, 2005, ISCAS, IEEE Symposium on

[4] J. Yuan, et.al. “ A Formal Study of Shot Boundary Detection”, *Circuit and Systems For Video Technology*, 2007, IEEE Transaction on

[5] V.Vapnik. “The National of Statistical Learning Theory”, 1st ed., New York: Springer-Verlag, 1995

[6] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/˜cjlin/libsvm

[7] M. Cooper and J. Foote. Scene Boundary Detection Via Video Self-Similarity Analysis. *Image Processing*,2001,Proc. IEEE Intl. Conf. on

[8] M.Cooper, et.al. “Video Segmentation via Temporal Pattern Classification”, *Transaction on Multimedia*, 2007, IEEE Transaction on

[9] D.Anguita, A.Boni, et.al.“A Digital Architecture for Support Vector Machines: Theory, Algorithms, and FPGA Implementation,” *Neural Networks*, 2003, IEEE Transaction on

[10] A. Hanjalic. “Shot boundary detection: unraveled and resolved?,” *Circuits Syst. Video Technol*, 2002, IEEE Transaction on

[11] R.Genov, et.al.“Kerneltron: Support Vector Machine in Silicon,”, *Neural Networks*,vol.14, no.5, pp.1426-1434, 2003, IEEE Transactions on

[12] R.Herbich. *Learning Kernel Classifiers:Theory and Algorithms.* Cambridge: The MIT Press, 2002.