Video Shot Boundary Detection using Visual Bag-of-Words

Jukka Lankinen[1] and Joni-Kristian Kämäräinen[2]
[1]*Machine Vision and Pattern Recognition Laboratory, Lappeenranta University of Technology (LUT Kouvola), Kouvola, Finland*
[2]*Department of Signal Processing, Tampere University of Technology, Tampere, Finland*

Keywords: Bag of Words, Bag of Features, Shot Boundary Detection.

Abstract: Recently, convergence of techniques used in image analysis and video processing has occurred. Many computation and memory intensive image analysis methods have become available for per frame processing of videos due to increased computing power of desktop computers and efficient implementations on multiple cores and graphical processing units (GPUs). As our main contribution in this work, we solve the problem of shot boundary detection using a popular image analysis (object detection) approach: visual bag-of-words (BoW). The baseline approach for the shot boundary detection has been colour histogram and it is at the core of many top methods, but our BoW method of similar complexity in the terms of parameters clearly outperforms colour histograms. Interestingly, an "AND-combination" of colour and BoW histogram detection is clearly superior indicating that colour and local features provide complimentary information for video analysis.

1 INTRODUCTION

Problem settings in image and video processing/analysis problems are almost equivalent, but adopted approaches have been divergent due to per frame processing required in many video processing tasks, such as in video shot boundary detection. For example, one hour of video contains approximately 100,000 frames, and the processing time of one second per frame would take 27 hours in total. In this kind of tasks typically "fast-to-compute-features", such as colour histograms, have been used. On the other hand, benchmark databases for image analysis have also become very large. For example, there are nearly 15 million annotated images in the ImageNet[1]. This has set new demands for approaches, and development has not only produced new techniques, but also more efficient implementations of the existing ones.

Thus, in this work, we adopt the state-of-the-art BoW method for processing of massive amounts of images: dense SIFT for feature detection and representation, k-means clustering for codebook generation, L1-normalisation of codebook histograms, and the Euclidean distance for code matching. Our main contribution is to apply this method for shot boundary detection. In addition, we compare video specific codebooks, generated from the local features extracted from an input video, to a "general codebook" generated from the ImageNet descriptors used in (Deng et al., 2010). Moreover, we study the effect of varying the codebook size, which is the most important parameter of BoW. The experiments are performed using the TRECVid 2007 shot boundary detection competition data. We compare our approach to the frame windows method (Tahaghoghi et al., 2005) which was among the top performers in (Smeaton et al., 2010) and can be considered as the baseline method for shot boundary detection. Our main contributions are:

- An efficient bag-of-features method for detecting shot boundaries. In the experiments, our method performed better than the baseline (note that colour histograms are used by many state-of-the-art methods).
- We investigate the effect of the codebook size and whether the codebook should be video specific or general, both being important computational considerations.
- We show how the combination of colour histograms and local feature histograms provides clearly superior results indicating that colour and local features provide complementary information for video processing.

[1]http://www.image-net.org/

2 PREVIOUS WORK

In many applications, such as in video abstraction (Truong and Venkatesh, 2007) and content based retrieval (Sivic and Zisserman, 2003), video shot boundary detection is the first step before higher level processing. For analysis, the shots are usually considered as basic units and thus success of the boundary detection affects the whole processing pipeline. The shot detection has been studied within specific applications and as its own problem and a wide variety of proposed methods exist.

A good introduction to the subject and an analysis of the best approaches with the common benchmark data were provided in a TrecVid survey (Smeaton et al., 2010) which summarised the findings over seven years of the TRECVid shot boundary competition. The vast majority of the best performing methods utilise colour histograms and machine learning algorithms, such as GMM (Gaussian Mixture Models) (Kang and Hua, 2005) or HMM (Hidden Markov Models) (Pruteanu-Malinici and Carin, 2008). It is noteworthy, that the colour histogram difference, which is considered as the baseline method, performs notably well and is virtually parameter free except the difference detection threshold (Gargi et al., 2000). The histogram-based methods utilise some distance-metric between the histograms of two consecutive frames which measures how content has changed. For example, high difference peaks on the time-line may denote hard cuts and sequences of smaller consecutive changes may denote fade-outs. The colour histogram based shot boundary detectors are fast and accurate when accompanied by heuristics for all transitions types (Tahaghoghi et al., 2005; Joyce and Liu, 2006; Mas and Fernandez, 2003). For experiments, we selected the colour histogram variant in (Tahaghoghi et al., 2005).

2.1 Visual Categorisation using BoW

The seminal works of the visual bag-of-features (BoW) are (Sivic and Zisserman, 2003) and (Csurka et al., 2004). In BoW, the salient local image features (interest points) are extracted with a special detector (e.g. SIFT) or fixed size patches are selected using dense sampling on a regular grid. Then, these "keypoints" are described with a descriptor, the SIFT descriptor being the most popular. In the training phase, a codebook is generated by clustering the descriptors into a fixed number of codes. In the matching phase, for each descriptor the best matching code is assigned. An image feature is generated by computing the histogram of codes appearing in the image. Matching can be performed by histogram similarity between two images (frames).

There is a huge number of variants and extensions of the baseline method (e.g., (Lazebnik et al., 2006; Leibe et al., 2008; Cao et al., 2010)), but often the basic method performs the best (Tuytelaars et al., 2010) and for large scale problems the most efficient discriminative methods are not feasible anymore (Deng et al., 2010). For this work, we adopt the recent implementation in (Deng et al., 2010). For feature detection the method uses dense sampling on a regular grid, which has lately replaced the interest point detection methods in the most visual object classification methods (Everingham et al., 2011). The descriptor of choice is SIFT, the codebook is generated using the k-means clustering and the feature histograms are L1-norm normalised.

The shot boundary detection methods using local features are only a few. (Li et al., 2010) computed SIFT regions and descriptors, but did not utilise a codebook. They directly searched for SIFT matches between consecutive frames. A similar approach for content analysis was proposed by Sivic and Zisserman (Sivic and Zisserman, 2003) who used a codebook. The both techniques, however, are extremely slow due to random sampling based matching. Sivic and Zisserman run the matching only for key frames of every shot as their application was content retrieval and Li et al. did not report the computation times for their method. To the authors' best knowledge our work is the first to propose the bag-of-words approach to video shot boundary detection.

3 SHOT BOUNDARY DETECTION

The main parts of our own implementation are similar to the approach presented by Deng et al. This is particularly the case with a general codebook generated from two million features extracted from ImageNet. It is interesting to study how well the general codebook performs as compared to a specific codebook, which is re-generated for every input video. Specific codebooks are generated using features extracted from selected frames (one frame per second in our implementation) and using the k-means clustering method. Next, the shot boundary detection algorithm is given in Alg. 1. It is noteworthy, that the only parameter for our method is the detection threshold τ which is equivalent to the colour histogram detection threshold. The other inputs are video and a pre-computed codebook.

Algorithm 1: Video shot boundary detection (BoW).

```
1: Load codebook $cb$.
2: for all Frames $i$ in video do
3:    Init $\vec{v}(i) \leftarrow 0$.
4:    Extract dense interest points and form their descriptors.
5:    Search the best matches in the codebook $cb$ for every extracted descriptor using the fast KD-tree search.
6:    Form the code histogram $\vec{h}$ using the codes.
7:    L1-norm normalise the histogram and compute the Euclidean distance $d_{curr}$ to the previous frame histogram.
8:    Calculate the distance difference (derivative) $d' = d_{curr} - d_{prev}$.
9:    If $d' \geq \tau$, then mark a shot boundary to the current frame $\vec{v}(i) \leftarrow 1$.
10: end for
11: Return the vector of shot boundaries $\vec{v}$.
```

4 EXPERIMENTS

The experiments were conducted with the TRECVid 2007 Shot Boundary data set which contains almost 7 hours of human annotated videos, 637,805 frames with 2317 transitions. In our evaluation, we used the TRECVid protocol, data and groundtruth, and the provided functionality in the available toolkit. For our method, the operating point is set by the difference threshold τ. Low values result to high recall but low precision, and vice versa. The precision-recall evaluation curves were computed by iteratively testing all possible values of the threshold τ.

4.1 Optimal Codebook Size

The size of the codebook (the number of clusters in K-means) is one of the computational bottlenecks. In object classification, the codebook sizes vary between 1,000 and 100k, but in our case as small as possible is preferred. The precision-recall curves for our method and with varying codebook size are shown in Fig. 1. It is evident that the boundary detection is a low level task which requires only moderate discrimination power from the codebook. Already 100 codes performed very well and increasing the size did not improve the results. The method started to collapse with codebooks smaller than 10.

4.2 General vs. Specific Codebook

Based on the previous experiment, a generated code-

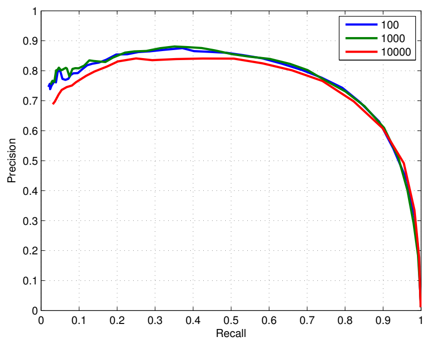

Figure 1: TRECVid precision-recall curves using our BoW method and different codebook sizes.

book of size 100 performs well in shot boundary detection. That does not produce significant computational burden since the local features need to be extracted anyway. However, this could be improved if a general codebook would perform well, since the codebook generation step could be completely omitted. A general codebook was generated using the two million ImageNet features used in (Deng et al., 2010). The shot boundary detection results and comparison to a specific codebook (100) are shown in Fig. 2. The results provide a clear evidence that the general codebook does not perform well in this application and changing the codebook size does not help the situation. This result is quite surprising from object class detection point of view, but for shot boundary detection, video specific codebooks should be used.

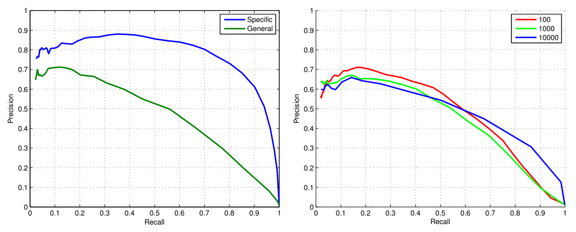

Figure 2: TRECVid results for general and specific codebooks (right: general codebook results with varying size).

4.3 Baseline Comparison

In the last experiment, we compared our method to the tailored colour histogram method in (Tahaghoghi et al., 2005). We also investigated complementarity of the two, colour histograms and BoW histograms. This was achieved by first running the both methods and then combining the output binary vectors (1's denoting a cut and 0's no cut). For combining the logical AND and OR rules were tested. The AND rule should mainly improve the precision and the OR rule the recall. The results are shown in Fig. 3. For the combinations, we tested all possible combinations of the thresholds τ_{BoW} and τ_{RGB} and for a recall point selected the highest precision. Our BoW method clearly outperforms the baseline method using colour histograms. However, it is evident that combining the two still improves detection remarkably.

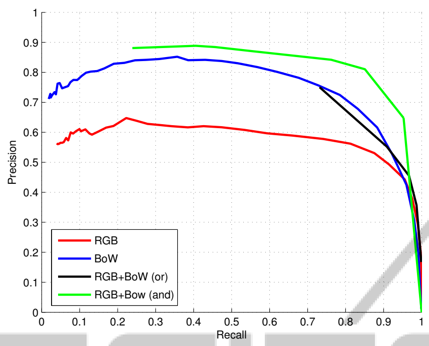

Figure 3: TRECVid comparison with the baseline method and with the hybrid of the two methods.

5 CONCLUSIONS

In this work, we adopted the popular approach for object class detection, visual Bag-of-Words (BoW), to the low level video processing task of video shot boundary detection. To the authors' best knowledge, our work is the first which uses the BoW approach in video shot boundary detection. We utilised the available efficient implementations and our method, which has equal complexity in terms of the number of parameters, achieved clearly superior performance to the baseline. This is an interesting result, since the baseline (colour histogram difference) is at the core of many top performing methods. Our method runs on half frame rate on standard PC hardware and without special optimisation. Moreover, our results showed that the two, BoW feature histograms and colour histograms, provide complementary information, and their combination achieved the best performance. In future work, we will investigate other low level video processing tasks using the BoW approach and optimisation of our implementation to run on at least frame rate.

REFERENCES

Cao, Y., Wang, C., Li, Z., Zhang, L., and Zhang, L. (2010). Spatial bag-of-features. In *CVPR*.

Csurka, G., Dance, C., Willamowski, J., Fan, L., and Bray, C. (2004). Visual categorization with bags of keypoints. In *ECCV Workshop on Statistical Learning in Computer Vision*.

Deng, J., Berg, A., Li, K., and Fei-Fei, L. (2010). What does classifying more than 10,000 image categories tell us? In *ECCV*.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. (2011). The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results. http://www.pascal-network.org/challenges/VOC /voc2011/workshop/index.html.

Gargi, U., Kasturi, R., and Strayer, S. H. (2000). Performance characterization of video-shot-change detection methods. *IEEE Trans. Circuits Syst. Video Techn.*, 10(1):1–13.

Joyce, R. A. and Liu, B. (2006). Temporal segmentation of video using frame and histogram space. *IEEE Transactions on Multimedia*, 8(1):130–140.

Kang, H.-W. and Hua, X.-S. (2005). To learn representativeness of video frames. In *ACM international conference on Multimedia*.

Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*.

Leibe, B., Ettlin, A., and Schiele, B. (2008). Learning semantic object parts for object categorization. *Image and Vision Computing*, 26(1):15–26.

Li, J., Ding, Y., Shi, Y., and Li, W. (2010). A divide-and-rule scheme for shot boundary detection based on SIFT. *Int. J. of Digital Content Technology and Its Applications*, 4(3).

Mas, J. and Fernandez, G. (2003). Video shot boundary detection based on color histogram. In *TRECVid Workshop*.

Pruteanu-Malinici, I. and Carin, L. (2008). Infinite Hidden Markov Models for Unusual-Event Detection in Video. *IEEE Trans. on Image Processing*, 17(5):811–822.

Sivic, J. and Zisserman, A. (2003). Video Google: A text retrieval approach to object matching in videos. In *ICCV*.

Smeaton, A., Over, P., and Doherty, A. (2010). Video shot boundary detection: Seven years of TRECVid activity. *Computer Vision and Image Understanding*, 114:411–418.

Tahaghoghi, S., Williams, H., Thom, J., and Volkmer, T. (2005). Video cut detection using frame windows. In *Australasian Computer Science Conference*.

Truong, B. and Venkatesh, S. (2007). Video abstraction: A systematic review and classification. *ACM Trans. on Multimedia Computing, Communications and Applications (ACM TOMCCAP)*, 3(1).

Tuytelaars, T., Lampert, C., Blaschko, M., and Buntine, W. (2010). Unsupervised object discovery: A comparison. *Int J Comput Vis*, 88(2).