An Improved Algorithm on Viola-Jones Object Detector

Qian Li, Usman Niaz, Bernard Merialdo

Multimedia Department, EURECOM

Sophia Antipolis, France

{Qian.Li, Usman.Niaz, Bernard.Merialdo}@eurecom.fr

Abstract

In image processing, Viola-Jones object detector [1] is one of the most successful and widely used object detectors. A popular implementation used by most image processing researchers and implementers is the one implemented in OpenCV. The detector shows its strong power in detecting faces, but we found it hard to be extended to other kinds of objects. The convergence of the training phase of this algorithm depends a lot on the training data. And the prediction precision stays low. In this paper, we have come up with new ideas to improve its performance for diverse object categories. We incorporated six different types of feature images into the Viola and Jones' framework. The integral image [1] used by the Viola-Jones detector is then computed on these feature images respectively instead of only on the gray image. In addition, we also integrated a key points based SVM [2] predictor into the prediction phase to improve the confidence of the detection result.

1. Introduction

Nowadays the enormous number of videos on the Internet provides us with large visual and audio information. We need efficient tools to annotate these videos automatically based on their content. There exist a large number of applications that depend on video annotation, such as content-based video retrieval, video clustering, video surveillance etc. The importance of individual object detection in videos for retrieval tasks cannot be denied as many real life videos contain significant contribution of local objects.

Most of the state-of-the-art object detection researches focus on face. And one of the most famous detectors is the rapid object detector designed by Viola and Jones. It was initially designed for face detection. Afterwards, it is also applied to detect other kinds of object. It works well on the faces, achieving no worse performance than the best previous systems [18, 19, 20, 21, 22]. Its fast processing speed caters to the need of real-time applications. But through experimentation, we found that it didn't work really well as expected for general-purpose object detection task. In addition, for a certain subset of training data, the training phase of the algorithm doesn't converge.

The Viola-Jones object detector uses Haar-like [16] features, which are reminiscent of Haar basis functions, to train the stage classifier for the cascaded classifier. The Haar-like features are predefined and computed directly on the integral image of the gray image. So the first contribution of this paper is that we have introduced multiple feature images into training the stage classifier instead of only the gray image. For one stage, several stage classifiers are trained on these feature images respectively. The one gives out the biggest discrimination between the object and non-object image patches wins and will then be selected as the stage classifier for the current stage.

The second contribution of this paper is to avoid the case where the training phase can't converge. The Viola-Jones' stage classifier training iteration ends on the false alarm rate reaching a predefined threshold. But for a certain set of training samples, the false alarm rate does not reach the predefined threshold based on our experiments. Here we introduce a new stopping criterion to terminate the training of the stage classifier, the maximum variance ratio between score of positive image patches and score of the negative ones.

The third contribution of this paper is to make the algorithm output a real-value weighted score for each test image, which represents the confidence that the image contains a desired object.

The remainder of the paper is organized as follows. Section 2 details the new elements that we have

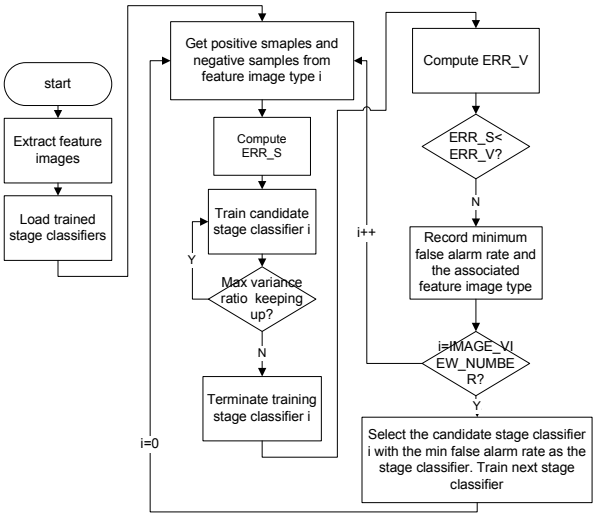

Fig 1. Training phase of the proposed framework

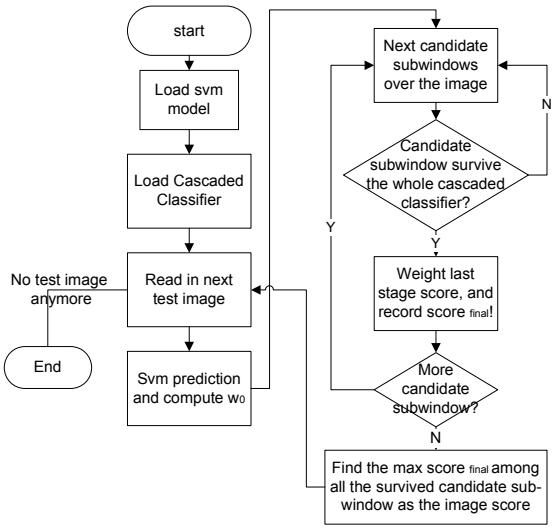

Fig 2. Prediction phase of the proposed framework

introduced into the Viola-Jones object detector implemented in OpenCV [17]. Section 3 would then set up the experiment environment and present the result. Section 4 concludes the paper and introduces some ideas into the future work.

2. THE PROPOSED FRAMEWORK

The flowchart of the proposed object detection framework is shown in Figure 1 (training phase) and Figure 2 (prediction phase).

2.1. Training Phase

The training phase consists of 2 types of classifier: AdaBoost classifier [3] (stage classifier) and cascaded classifier. The cascaded classifier is a degenerated tree of stage classifiers.

Within any image patches, the total number of Haar-like features is very large, far larger than the number of pixels. In order to increase its speed, Viola and Jones have made a simple modification on the AdaBoost procedure. The weak learner is constrained so that each weak classifier returned depends on only one single feature. AdaBoost provides an effective learning algorithm and strong bounds on generalization performance [4,5,6]. And to satisfy the real time demand of the detection algorithm, Viola and Jones computed Haar-like features directly on integral image.

Based on these, first, in this paper we have proposed 6 different feature images for all the objects. For one stage of the cascaded classifier, train 6 stage classifiers based on 6 feature images respectively, as shown in Fig 3. We select stage classifier with minimum false alarm rate as the winner.

The training data is preprocessed and a set of feature images are generated as shown in Figure 4.

These feature images are built based on pixel-wise local image features. We replace the image pixel p with a new value P, which is computed from the local image feature associated with p. In the experiment, the chosen 6 feature images are extracted as follows.

Feature image type I: Gray Image

Feature image type II: LBP [13, 14] Image

LBP image is extracted on the basis of gray image. For each image pixel, compare it with its 8-neighbor pixels respectively. Starting from its upper-left neighbor pixel, visit its 8-neighbor clock-wisely and update the P's bits from left to right correspondingly. If it is bigger than its neighbor, then assign a '*1*' to the corresponding bit position in P, otherwise a '*0*'. This byte P is the new value of this image pixel. As shown in Fig 5, the resulting new value should be:

$$211 = 2^7 + 2^6 + 2^4 + 2^1 + 2^0.$$

Feature image type III: EDGE Image

We use an improved canny edge detector [7], which automatically chooses the high threshold and low threshold value according to the image binarization threshold [11]. The image binarization threshold is then chosen to be the high threshold in canny and its 40% as the low threshold.

Feature image type IV: L-channel Image
L-channel of the Lab [8, 9] color channels.

Feature image type V: A-channel Image
A-channel of the Lab color channels.

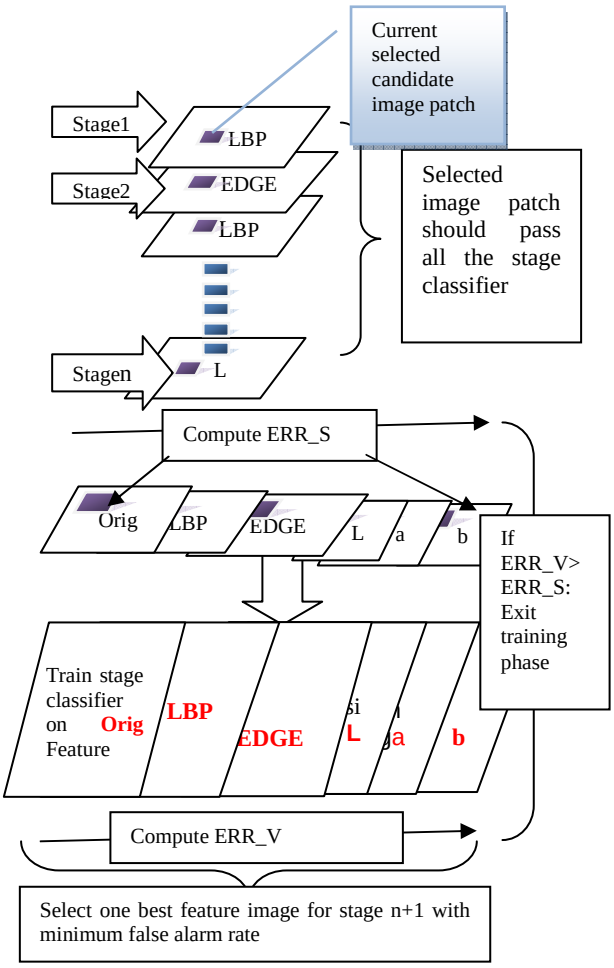

Fig 3. Training sample selecting process and over fitting counter-measure integration

Feature image type VI: B-channel Image
B-channel of the image's Lab color channels.

Second, the state-of-the-art implementation of the stage classifier in OpenCV use false alarm rate reaching a predefined value as its stopping criteria, while this, proved by experimentation, sometimes it doesn't converge even until all the Haar-like features are used to train the weak classifier, the total number of which is far larger than the number of pixels. The total time consumed to find the stage classifier thus is quite long and probably with no result at all in the end.

Here we introduce new stopping criterion: maximum variance ratio (*R*) between score of positive samples and score of the negative ones. The idea is to separate the positives and the negatives as far as possible and meanwhile keep the intra-variance of each class as small as possible (in a consideration of robustness). *R* is defined with equation 1.

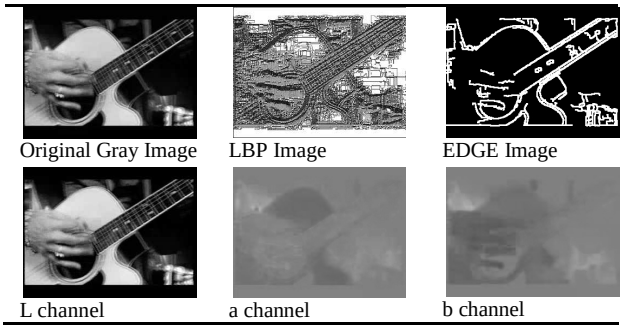

Fig 4. Six feature images for 'Hand' object

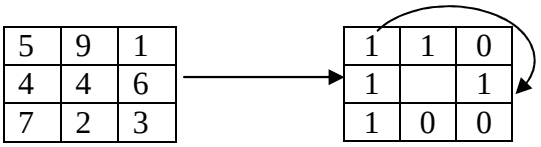

Fig 5. Compute new pixel value for the LBP intensity image, P=11010011=211

$$R = \frac{card(p)*[E(p)-E(p+n)]^2+card(n)*[E(n)-E(p+n)]^2}{card(p)*D(p)+card(n)*D(n)} \quad (1)$$

where p is a set of *scores* of positive samples, n is the set of *scores* of negative samples. *Score* is defined as the stage sum of the last stage classifier of a survived image patch. Stage sum is the cumulative sum of Haar-like features convolved with the image patch. *E (.)* represents the mean of the set, *D (.)* represents the variance of the set and *card (.)* is the number of elements in this set.

We continue the stage training if R keeps increasing. Intuitively, the stage classifier training will finally converge since R is not goanna be very large or keeps increasing all the time.

Third, generally it is the user who defines the total number of stages used by the cascaded classifier. And in most cases, this number could only be found by repeating multiple parallel experiments, which is still a blind process costing a lot of time. Here we have used a small trick to decide the training stages automatically, a set of validation data.

As shown in Figure 3, before starting to train the stage classifier, compute the error rate on training data using the previous trained stages: *ERR_S = FP* (false positive) + *FN* (false negative); After training each candidate stage classifier, compute the error rate on validation data: *ERR_V = FP + FN*; If *ERR_S < ERR_V*, then we could assert that a probable over fitting occurs. The training process is stopped even if the user defined stage number hasn't been arrived yet.

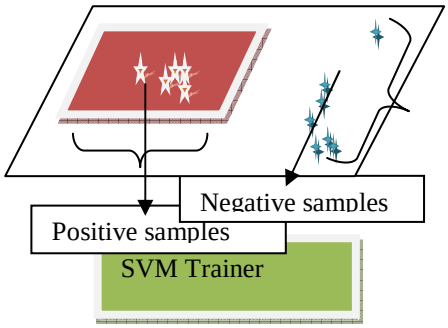

Fig 6. SVM training phase

2.2. Prediction Phase

The key points based SVM predictor is integrated into the prediction phase to weight the output score. We use SIFT [23] here. We compute one global weight for all the candidate image patches in the test image and one local weight for each candidate respectively using this predictor.

Firstly, we train the SVM predictor as in Figure 6. We extract the SIFT key points from the training data set. Those key points falling in the desired object are treated as the positive samples, and those outside the desired object are treated as the negative ones.

Secondly, we extract the key points $\{X\}$ from the test image and predict it using the SVM predictor, then we get $\{p(x)|x \in X\}$ which represents the probability of a key point *x* belonging to the desired object.

The global weight is defined with equation 2:

$$w_o = \frac{card(\{x|x \in object\})}{card(\{x|x \in object\}) + card(\{x|x \notin object\})} \quad (2)$$

To compute the local weight, for each candidate image patch in the test image, find $\{Y|Y \subset X\}$, Y represents the key points which are included in this candidate image patch. For computation convenience, regard$\{y|y \in Y\}$ as independent random variables and so we can compute the entropy of *Y* using equation 3.

$$H(Y) = -\sum p(y)\, log\, p(y)\ , y \in Y \quad (3)$$

H(*Y*) indicates the uncertainty included in these key points, thus*1-H*(*Y*) shows how much we could trust on the information provided by these key points.

If the candidate image patch contains the object, then it should have a higher *difference*, which is defined with equation 4:

$$difference = \frac{\sum p(m)}{card(\{m\})} - 1 + \frac{\sum p(n)}{card(\{n\})} \quad (4)$$

where $m \in object, n \notin object, m \in Y, n \in Y$

So for each candidate image patch, we compute a local confidence weight with equation 5:

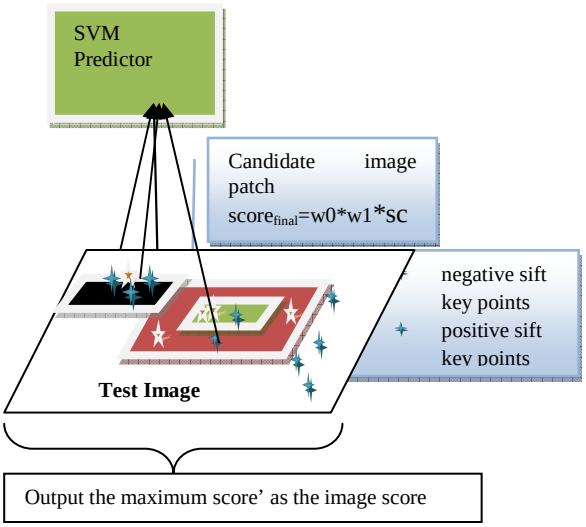

Fig 7. Score computation on test image

$$w_1 = e^{difference*(1-H(Y))} \quad (5)$$

Here we use an exponential function as the transfer function to adjust the influence of *difference*(1-H* (*Y*)) on the final *score* $_{final}$. w_1 is bounded between *1* and *e*.

So the final score for each survived candidate image patch can be computed using equation 6:

$$score_{final} = score * w_1 *(1+w_0) \quad (6)$$

Survived image patches are those who have passed the evaluation of all the stage classifiers defined in the cascaded classifier. For each survived image patch, it is associated with a *score* by definition.

We pick the biggest *score* $_{final}$ among all the survived image patches as the final score for the test image. The computation scheme is briefly shown in Figure 7.

3. Experiments and Results

In our experiments we operate on the TRECVID 2011 development dataset [15]. The number of training samples (positive, negative and validation data) and test images for each concept are listed in Table 1. Test images are the same for all the objects while training images used for extracting samples are independently, separately and randomly chosen from the dataset.

3.1 Experimental Setup

We have chosen 4 objects for the experiment: Scene_Text, Computers, Telephones and Hand, which are part of the list of concepts for TRECVID 2011. For each object, we prepared positive and validation training samples by annotating video frames manually using Object Annotator [10]. And for each object, we

Table 1. Training and testing data number

Objects	Training samples			Test Images
	pos	neg	val	
Computer	233	1185	89	113
Hand	710	1368	53	113
SceneText	516	1191	110	113
Telephone	34	1246	7	113

Table 2. Parameters selected for each object

Objects	width	height	nstages
Computers	24	24	20
Hand	24	20	20
SceneText	54	9	16
Telephone	20	16	14

select training parameters through experiments. The parameters chosen are listed in Table 2.

The parameter pair (width, height) actually defines the size of the candidate image patch. Ways to collect the positive data and negative data to train the SVM predictor are depicted in Figure 6.

For the original Viola and Jones object detector, image patches that have passed the last stage classifier are considered to contain the desired object, choose the maximum *score* without any weighting on these image patches as the score for the test image.

3.2. Results

The feature images that have been chosen for each object are listed in Table 3. Though we have chosen 6 different feature images, not all of them are used. Each object prefers one subset. We can see that LBP is a strong candidate for feature image as most objects select it. It shows strong local image features.

For the proposed algorithm the stage number is evaluated automatically during the training phase. The user doesn't have to do parallel experiments to find the best stage number for each object, which saves a lot of time and efforts at training phase. The algorithm would terminate when detecting the probable over fitting. But meanwhile the multiple feature image processing does bring a reduction on prediction speed.

From table 4, we can see that the stage numbers actually trained are tremendously reduced in the new algorithm compared to the column 'nstages' in Table 2, which are the stage numbers used by the original object detection algorithm.

In terms of the average prediction precision after introducing multiple feature images and SIFT based SVM predictor, the resulting average precision, as shown in Figure 8, is computed on the top 40 images detected with highest scores in all 113 test images. From Figure 8, we can see that the new algorithm

Table 3. Feature Image Chosen for Each Object

stage / object	1	2	3	4
Computers	*B*	*L*	*Orig*	*LBP*
Hand	*LBP*	*B*	*Orig*	*LBP*
Scene_Text	*Orig*	*LBP*	*LBP*	-
Telephone	*LBP*	*Orig*	*Orig*	*Orig*

Table 4. Stage Numbers Chosen for Each Object

object	Computer	Hand	Scene_Text	Telephone
stages	4	4	3	4

shows better performance than the original Viola and Jones' algorithm implemented in OpenCV.

4. Conclusions

The robustness of the detector is still a very trivial problem and also we cannot ignore the impact of the outliers. The detection depends a lot on the selected training data. The new algorithm doesn't address this problem right now. One way would to focus on finding methods to de-couple the dependency of the learning algorithm on the correlation between training data and testing data. Another possibility is to introduce heuristics into the training data selection phase to enhance the robustness of the algorithm.

Currently we have chosen the same 6 feature images for all objects. And thus, the improvement varies on different object. Afterwards, some work should be focused on inspecting suitable feature images particularly for each object.

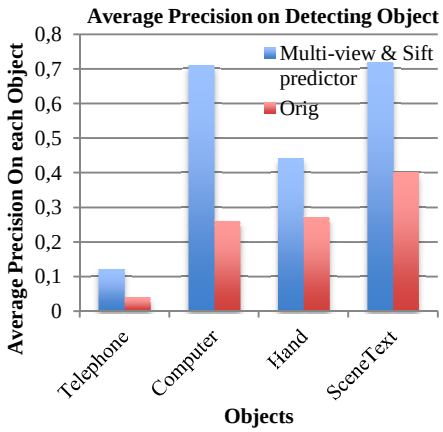

Fig 8. Object detection average precision on selected objects (top 40 in 113 test images)

5. References

[1] P. Viola and M. Jones. “Rapid object detection using a boosted cascade of simple features.” *Computer Vision and Pattern Recognition, IEEE Computer Society Conference* on, 1:511, 2001.

[2] C. Chang and C. Lin. LIBSVM: a library for support vector machines. 2001.

[3] Y. Freund and R. E. Schapire. “A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting.” In *Computational Learning Theory*: Eurocolt ’95, pages 23–37. Springer-Verlag, 1995.

[4] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. “Boosting the margin: a new explanation for the effectiveness of voting methods.” Ann. Stat., 26(5):1651–1686, 1998

[5] E. Osuna, R. Freund, and F. Girosi. “Training support vector machines: an application to face detection.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[6] C. Papageorgiou, M. Oren, and T. Poggio. “A general framework for object detection.” In *International Conference on Computer Vision*, 1998.

[7] J. Canny, “A Computational Approach To Edge Detection”, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6): 679–698, 1986.

[8] R. S. Hunter (July 1948). "Photoelectric Color-Difference Meter". JOSA 38 (7): 661. (Proceedings of the Winter Meeting of the Optical Society of America)

[9] R. S. Hunter (December 1948). "Accuracy, Precision, and Stability of New Photo-electric Color-Difference Meter". JOSA 38 (12): 1094. (Proceedings of the Thirty-Third Annual Meeting of the Optical Society of America)

[10]http://image.ntua.gr/iva/tools/annotator_semi_automatic_image_annotation_program

[11] J. BAI, Y. Q. YANG, R. L. TIAN “Complicated image’s binarization based on method of maximum variance” Proceedings of *the Fifth International Conference on Machine Learning and Cybernetics*, Dalian, 13-16 August 2006

[12] D. G. Lowe (1999). "Object recognition from local scale-invariant features". Proceedings of *the International Conference on Computer Vision*. 2. pp. 1150–1157. doi:10.1109/ICCV.1999.790410.

[13] T. Ojala, M. Pietikäinen, and D. Harwood (1994), "Performance evaluation of texture measures with classification based on Kullback discrimination of distributions", Proceedings of *the 12th IAPR International Conference on Pattern Recognition* (ICPR 1994), vol. 1, pp. 582 - 585.

[14] T. Ojala, M. Pietikäinen, and D. Harwood (1996), "A Comparative Study of Texture Measures with Classification Based on Feature Distributions", *Pattern Recognition*, vol. 29, pp. 51-59.

[15] A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and trecvid. In MIR '06: Proceedings of *the 8th ACM International Workshop on Multimedia Information Retrieval*, pages 321{330, New York, NY, USA, 2006. ACM Press.

[16] C. Papageorgiou, M. Oren and T. Poggio, "A general framework for object detection.", In *International Conference on Computer Vision*, 1998

[17] http://opencv.willowgarage.com/wiki/

[18] K.Sung and T.Poggio. “Example-based learning for view-based face detection”. In *IEEE Patt.Anal.Mach.Intell.*, volume 20, pages 39–51, 1998.

[19] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *International Conference on Computer Vision*, 2000.

[20] D. Roth, M. Yang, and N. Ahuja. “A snowbased face detector”. In *Neural Information Processing 12*, 2000.

[21] H. Rowley, S. Baluja, and T. Kanade. “Neural network-based face detection”. In *IEEE Patt. Anal. Mach. Intell.* , volume 20, pages 22–38, 1998.

[22] Y. Amit, D. Geman, and K. Wilder. “Joint induction of shape features and tree classifiers”, 1997.

[23] David G. Lowe “Distinctive Image Features from Scale-Invariant Keypoins” In *International Journal of Computer Vision*, vol. 60, no 2, 2004, p. 91-110