Object identification in image using global low-level features

Ivo Řezníček
Department of Computer Graphics and Multimedia
Faculty of Information Technology, Brno University of Technology
Brno, Czech Republic, 612 66
ireznice@fit.vutbr.cz

Abstract — *This paper introduces a method of object recognition in image or video sequence using low-level features and powerful machine learning techniques. The goal of the method is to identify an object in unknown images or video. For this purpose optimal svm detector must be created, for which a good dataset is needed with desired images to train this detector and for computing it's accuracy.*

1 INTRODUCTION

Modern technology makes it possible to acquire huge sets of video content e.g. from TV broadcasting, meeting rooms, security systems etc. Such data can be further reused for various purposes. Searching objects in frame within large video libraries is very time consumpting, we introduce the way how to create relatively quick way for detection of the wanted object in video frame, using feature extractors and machine learning techniques. In figure 1 you can see examples of objects we want to detect in a unknown image, signmarker (first column), or two people in image (second column).

Figure 1: Examples.

This paper introduces the way of classifier creation. One and only input to this procedure is large data base of images with the object of interest and images that do not contain the object of interest. When the classifier is selected, we can use it to detect object in unknown images or video frames. We have used sound and vision development and test data from TRECVID 2008 Evaluation [3], where 20 objects in images were annotated.

2 SYSTEM OVERVIEW

Procedure could be splitted to two main part, classifier creation and using this classifier.

The first main part – is illustrated in figure 2. Annotated dataset contains images of interest, where the object is located, these samples are called positive samples. And also contains negative samples, where the object is not located.

Feature extraction must be performed to each image in dataset to perform transformation to feature space, then all following operations are going to be performed in n-dimensional feature space.

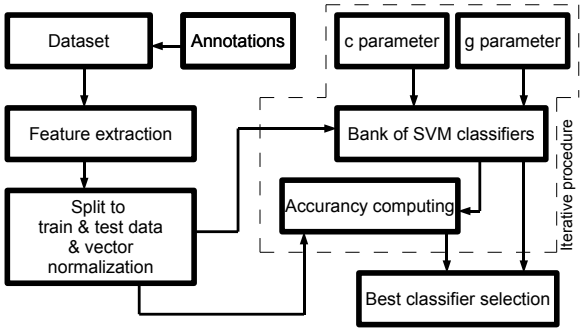

Figure 2: Block diagram.

Classifier creation is a blind process, we need some part of the dataset to create the classifier and another part to compute it's accuracy, then we need to split the dataset to the train and testing parts to do these two actions. Afterwards feature space must be rescaled, using suitable normalization function.

Now there's possibility to create the bankof classifiers for our detection problem. Quality of each classifier is defined of course by the dataset quality

and there are c and g training parameters. These have to be experimentally set, accuracy of the classifiers have to be computed and classifier with the best accuracy is selected as the final classifier.

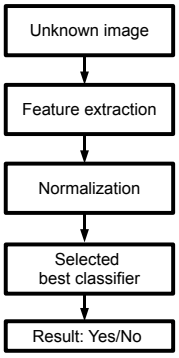

Figure 3: Unknown image detection block diagram.

Second main part – Object detection in Unknown image. Object could be detected using analogous way, all presented in figure 3. Image is transformed to feature space, normalization is performed and our selected best classifier determine if image displays desired object or not, this procedure is much times quicker compared with classifier creation.

2.1 Feature Extraction

The machine learning and classification process is based on feature extraction methods whose result is a feature vector. Feature extraction is performed for every image and/or every frame in video sequences. The features used in the presented approach include:

- color histogram based on HSV color model,
- multi-scale gradient distribution of a frame intensity,
- color layout,
- gabor texture.

Every feature extractor produces it's own feature vector, these all are finally joined together in predefined order to produce one long feature vector. This feature vector represents the image in the feature space, in classifier creation process and in unknown image detection as well. Then the feature fectors need to be joined the same way during detection of unknown image.

2.1.1 Color Histogram Based on HSV Color Model

The color histogram contains statistical information about color distribution in terms of frequency of hues and saturations in the frame (using HSV color model). The better spatial description is achieved by dividing the frame into several patches. The frame division is not adaptive, so all the patches have the same size. Each patch is processed separately. After the histogram is computed, it is normalized.

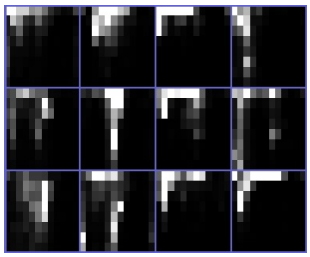

Figure 4: Color histogram descriptor visualization.

2.1.2 Multi-scale Gradient Distribution

The histogram of gradient orientations forms other part of the feature vector. First, the frame gradients are computed. Then each gradient contributes to the histogram bin according to its orientation. The contributions are weighted by the gradient magnitude. The gradients are computed on different frame resolutions so also lower frequency structures can contribute to the final feature vector. The resolution of both color and gradient histogram, the resolution of the frame grid, and the frame scale are the descriptor's parameters.

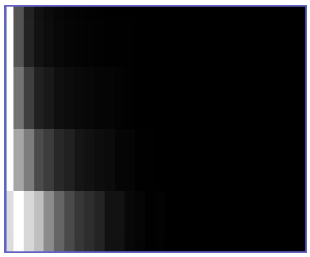

Figure 5: Gradient distribution descriptor visualization.

2.1.3 Color layout

The color layout computation is based on the technique used also in JPEG compression. First, the

image is subdivided into 8x8 pixels blocks in YCbCr color model. Then, the discrete cosine transform (DCT) is applied on each colour channel. The descriptor coefficients are then extracted in the zigzag manner [2], as illustrated in figure 6. We use 20 (Y) + 15 (Cb and Cr) coefficients, so the descriptor has 50 coefficients in total.

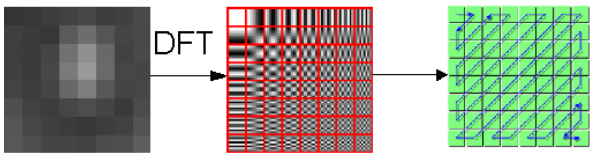

Figure 6: Color layout description process.

2.1.4 Gabor Texture

Using a bank of Gabor filters in the frequency domain, we can divide the space, created using Fourier transform, into bands, as illustrated in figure 7. We use the first moments of energy in the filtered 30 sub-bands (G_P) – 6 angular (30 deg) and 5 radial (in octaves) for construction of the descriptor.

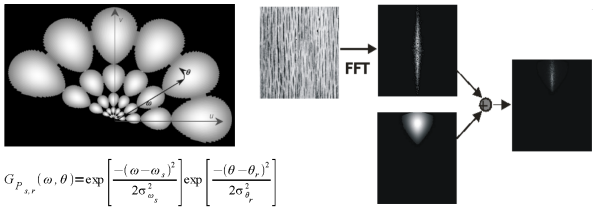

Figure 7: Color histogram descriptor visualization.

2.2 Dataset splitting and normalization

All feature vectors have to be splitted to test and train data, train data are used for training of classifier and test data to compute it's accurancy. Input to splitting function is *splitting ratio* which denotes distribution of data into train and test part.

Splitting ratio 0.6 represents, that 60% of dataset are train data and the rest are test data. For both parts of the whole dataset (positive and negative samples) must be created an one dimensional array with the same length (as count of samples), filled with zeros. These zeros represents that all samples are train data for now. Next step is creating split distribution, it is based on random number generation, using boost library [6]. We generate random numbers, in interval of valid indexes of array, and changing values in array on generated indexes to ones. This procedure is repeated until for example 40% of array is transformed to ones. This is the moment when we have splitting distribution and we are able to split positive and negative samples to testing and train part using splitting ratio. Most frequently ratios are in interval 0.6 to 0.8.

All feature vectors have to be normalized across the whole training subset of the dataset into a closely defined interval (from 0.0 to 1.0). The normalization procedure is as follows:

- for every subitem position in feature vector is calculated minimum and maximum value,
- then whole subset is re-scaled, so that these minima and maxima corresponds to values 0.0 and 1.0 respectively.

The same scaling factors are used also for the test data (in which case the scaled values may exceed the wanted interval but these infrequent cases were not found harmful for evaluation.

2.3 Creation of model

For classification purposes Support Vector Machines [4] were chosen. This approach could be used to modelling of n-dimensional hyperplane for separation between positive and negative samples in the whole feature space. For that purpose only support feature vectors are used and the splitting hyperplane is computed. Support vectors are vectors with minimum distance of intended hyperplane and these are mainly used for it's computing.

Weakness of this approach is that very small number of mis-annotated images and their feature vectors in feature space can dramatically decrease the performance of the output classifier and overfitting happens very easily. Another problem with Support vector machines could be, that the feature space is finally separated only into two parts. If we want to detect for example two kinds of objects, we want to in fact separate feature space into three parts, it is not possible, but two separate models have to be created.

For practical usage of Support vector machine we have chosen the libsvm library [5]. Training process could be controlled by c and g parameters. These parameters are most frequently powers of two in interval $2^{-12} - 2^{+12}$. It creates parameters space with 625 combinations, which have to be searched and the best training combination have to be found. For that purpose we use iterative way of computing, where for example first 40 combinations with higher granularity of intervals is selected, then quality of all classifiers is computed and these information tell us which part of parameters space should be searched with smaller granularity. This could be

repeated until the best part of parameters space is discovered. Selection of the best classifier is now very easy.

The quality of the single classifier is represented by these ratios

- false positive ratio,
- false negative ratio,
- true positive ratio,
- true negative ratio.

These ratios tell to the user four quality measures of the classifier, and the user can consider which ones are important for classifier creation and then the correct part of intervals in the iterative process could be selected.

the iterative process of classifier creation is very time consuming. Thus we tried to speed up this process, some parts of classifers creation could be paralellized using a computing cluster. In time when we have finite set of combinations of c and g parameters in one iteration, we can run all training and following testing procedures in parallel on isolated computational units. We use Sun Grid Engine [7] for controlling execution of jobs on up to 170 computers, with 500 computational units.

3 RESULTS

We have used this approach in TREC Video Retrieval Evaluation 2008 [3] in feature extraction task. 20 situations in image (classes) to detect were described, development and testing dataset were disposal. Development data were used to creating of models for every classes, testing data were used to global evaluation of all participating groups (in TRECVID 2008).

Our results were average, classifiers accuracy were in the interval from 45 % to 65 %. It depends mainly on input dataset quality, in some cases dataset contained too few positive images.

Computational time for creating one classifier depends maily on count of positive images and iterations count. One iteration takes approximately 30 minutes using computer cluster, some time takes manual interval selection for the next iteration. Number of iterations were most often three, so if we iterate three times classifier could be created in time up to 2 hours.

4 CONCLUSIONS

Our solution can be generally described as a brute-force approach, which relies on generic software pieces (several feature extractors, SVM library, training/evaluation framework for distributed computing), that solve the task in an "uninformed" way.

For future implementations of object detection in image we intend to include more object detectors and similar frame-processing engines to provide specialized and "informed" knowledge to the overall classification process. These will be represented as mid-level features entering the per-frame classifier.

References

[1] Yixin Chen and James Z. Wang, *Image Categorization by Learning and Reasoning with Regions*, Journal of Machine Learning Research, vol. 5, 913-939, August 2004.

[2] International Telecommunication Union. 1992. Information Technology – Digital Compression and Coding of Continuous-tone Still Images – Requirements and Guidelines. http://www.w3.org/Graphics/JPEG/itu-t81.pdf. November 2008.

[3] Smeaton, A. F., Over, P., and Kraaij, W. 2006. Evaluation campaigns and TRECVid. In Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval (Santa Barbara, California, USA, October 26 - 27, 2006). MIR '06. ACM Press, New York, NY, 321-330. DOI = http://doi.acm.org/10.1145/1178677.1178722.

[4] J. A. K. Suykens, Support vector machines: A nonlinear modelling and control perspective, Eur. J. Control 2001, 7, 311-327. November 2008

[5] 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, November 2008

[6] Boost C++ library. Available at http://www.boost.org, November 2008.

[7] Sun Grid Engine, http://gridengine.sunsource.net/, November 2008